Universidade de Évora - Escola de Ciências Sociais

Mestrado em Ciências da Educação - Administração, Regulação e Políticas Educativas

Dissertação

Contributos da programação para o desenvolvimento do pensamento computacional em alunos do 1.º ciclo do Ensino Básico. Um Estudo de caso no Agrupamento de Escolas de Borba

Elsa de Fátima Velez Severo Rôlo

Orientador(es) | José Luís Pires Ramos

Évora 2021

Universidade de Évora - Escola de Ciências Sociais

Mestrado em Ciências da Educação - Administração, Regulação e Políticas Educativas

Dissertação

Contributos da programação para o desenvolvimento do pensamento computacional em alunos do 1.º ciclo do Ensino Básico. Um Estudo de caso no Agrupamento de Escolas de Borba

Elsa de Fátima Velez Severo Rôlo

Orientador(es) | José Luís Pires Ramos

Évora 2021

A dissertação foi objeto de apreciação e discussão pública pelo seguinte júri nomeado pelo Diretor da Escola de Ciências Sociais:

Presidente	|	Marília Favinha (Universidade de Évora)
Vogais	|	José Luís Pires Ramos (Universidade de Évora) (Orientador) José Luís Torres Manano Ramo Carvalho (Universidad de Extremadura) (Arguente)

Évora 2021

Agradecimentos

Este espaço é dedicado a todas e a todos que de alguma forma contribuíram para a realização deste projeto. Não sendo possível nomeá-los a todos, deixo aqui os meus sinceros agradecimentos.

Ao meu marido, companheiro e amigo, Paulo Rôlo, pelo apoio, incentivo, cumplicidade e paciência durante esta etapa das nossas vidas.

À minha filha, companheira, determinada e exigente, Sofia Rôlo, pelo apoio e incentivo.

Aos meus pais, Rosa e João Severo, pela vida, pelo apoio, preocupação, incentivo ao estudo, carinho e valores que não se perderam no tempo.

Ao meu Orientador, Professor Doutor José Luís Ramos, pelas sugestões e incentivos, contribuições sem as quais não teria chegado até aqui. Muito obrigada!

A todos os professores do curso pelo incentivo contínuo pela busca do conhecimento.

À direção do Agrupamento de Escolas de Borba, na pessoa do seu Diretor, Professor Agnelo Baltazar, pela sua contribuição neste estudo mostrando-se sempre disponível em colaborar em tudo aquilo que lhe foi solicitado.

Às colegas de turma em especial ao grupo mais próximo, Renata, Merciana e Fátima, companheiras que por meio de conversas e risos soltos contribuíram para aligeirar o peso desta formação.

Ao Miguel Verdugo, pelo empenho, dedicação e excelente trabalho.

Às colegas de trabalho Luísa Carapeta, Maria Luísa Sá e Helena Mouquinho pela partilha, compreensão, apoio e entreajuda.

Ao professor e amigo José Alberto Fateixa pela orientação, disponibilidade e partilha de conhecimentos.

Ao amigo Luís Filipe Pardal, pela paciência, apoio e conhecimentos partilhados.

Aos amigos Helena Verdugo, Maria José Parente, Carla Catarino, Ana Rijo, João Carlos Verdugo, Luís Parente e Pedro Gouveia pela amizade e carinho.

A todos aqueles que se esforçam por manter uma Escola Pública de Qualidade.

Dedicatória

Ao avô José Augusto Velez (In memoriam) pelo carinho eternizado e pela referência que foi e é na minha vida académica.

Resumo

Contributos da programação para o desenvolvimento do pensamento computacional em alunos do 1.o ciclo do Ensino Básico. Um Estudo de caso no Agrupamento de Escolas de Borba

O pensamento computacional é uma abordagem para a resolução de problemas, conceção de sistemas e compreensão do comportamento humano que se baseia em conceitos fundamentais das Ciências da Computação e que se aplica nas ciências, no trabalho e na vida quotidiana. Esta definição de Jeannette Wing, em 2006, foi a propulsora da atual mobilização de educadores, entidades profissionais, empresas e pesquisadores das Ciências da Computação interessados em colocar em prática o desafio de ensinar a pensar computacionalmente. O crescente consenso sobre a importância do tema tem aproximado entidades distintas com o intuito de desenvolver metodologias e ferramentas adequadas para o desenvolvimento do pensamento computacional em contextos de aprendizagem formais e informais.

As diferentes iniciativas partilham um conjunto de softwares de programação, jogos digitais e kits robóticos. Estudos diversos têm sido conduzidos para avaliar as potencialidades pedagógicas dessas ferramentas, onde se insere o software Kodu Game Lab, que permite qualquer criança/jovem construir o seu próprio jogo.

A presente dissertação apresenta um estudo de caso realizado tendo em conta os objetivos transversais do currículo do 1º Ciclo do Ensino Básico e o desenvolvimento do Pensamento Computacional. O software utilizado é o KODU Game Lab, que se caracteriza por ser uma linguagem de programação visual e que conta com um ambiente de programação simples e intuitivo.

A investigação permitiu observar, que através da aprendizagem da programação visual e em

particular com recurso ao software Kodu Game Lab, os alunos reforçaram competências cognitivas, sociais e atitudinais, tais como: dialogar e trabalhar de forma colaborativa, construir um jogo a partir de uma história (storyline), programar os personagens (automação), executar e avaliar o seu jogo (análise e depuração de erros) e raciocínio lógico.

Palavras-chave: Pensamento Computacional, Kodu Game Lab, Ensino da Programação, Aprendizagem, Ensino Básico.

Abstract

Contributions of programming to the development of computational thinking in students of the 1st cycle of Basic Education. A Case Study in the Borba School Grouping

The Computational Thinking is an approach to solving problems, designing systems, and understanding human behavior that is based on the fundamental concepts of computer Science, and applied sciences both at work and in their daily lives. This is the setting for Jeannette Wing in 2006, and has been the driving force of the current, the mobilization of teachers, professional bodies, corporations, and researchers of computer Science who are interested in putting it into practice is the challenge-to teach you to think computationally. The growing consensus on the importance of this issue has been bringing different bodies, with the aim of developing methodologies and tools for the development of Computational Thinking in the context of learning, both formal and informal.

The initiatives share a common set of software programming, computer games, and robotic kits. Various studies have been conducted in order to evaluate the didactic potential of these tools, which includes software, Kodu Game Lab, which allows any child to build his own.

This dissertation presents a Case Study carried out, taking into account the learning objectives across the curriculum in the 1st Cycle of Basic Education and the development of Computational Thinking. The software used was KODU Game Lab which is a programming language that is visual, and that it has a programming environment simple and intuitive to use. We were able to observe through the learning of visual programming (Kodu Game Lab), students will have reinforced cognitive, social and attitudinal skills such as: sharing and working in a collaborative

manner, to build the game around a story (the storyline), and plan out the characters (automation), carry out and evaluate your play (analysis and purification) and logic reasoning.

Keywords: Computational Thinking, Kodu Game Lab, Programming, Learning, Elementary Education

Índice

Índice de Figuras

Introdução

O artigo de 2006 de Jeannette Wing em defesa do ensino do pensamento computacional para todas as pessoas e não apenas para os cientistas (Wing, 2006) veio sustentar a ideia de que todos podem aproveitar as vantagens dos métodos usados em Ciências da Computação para a resolução de problemas, desenvolvimento de sistemas e compreensão do comportamento humano. Para Wing, a habilidade de pensar logicamente, reconhecer padrões, fazer abstrações, decompor problemas e apresentar soluções de forma algorítmica será, para o cidadão do século XXI, tão importante quanto ler, escrever e fazer operações aritméticas básicas (Wing, 2006).

O argumento em defesa do desenvolvimento do pensamento computacional remonta às ideias de Seymour Papert. Mas apesar do pioneirismo de Papert, foi o artigo de Wing que ajudou a popularizar o termo em meados da primeira década do século XXI.

Desde a sua publicação, acumulam-se evidências que reforçam os argumentos educativos, sociais e económicos para o desenvolvimento do pensamento computacional desde a infância. O consenso em torno do assunto tem aproximado educadores, agentes políticos e empresas de tecnologia, como Microsoft e Google, para o desenvolvimento de metodologias e ferramentas adequadas para a promoção do pensamento computacional em ambientes formais e informais de aprendizagem.

Na Europa, a tendência é confirmada pelo relatório Computing our future: Computer Programming and Coding (2015) lançado pela rede europeia dos Ministérios da Educação (European Schoolnet). O relatório apresenta uma visão geral da integração do ensino de programação e pensamento computacional em sistemas educativos de 20 países europeus e Israel. O Reino Unido, por exemplo, inseriu o pensamento computacional nas competências digitais que devem ser desenvolvidas desde os primeiros anos de escolaridade.

Em Portugal, destaca-se o projeto-piloto "Iniciação à Programação no 1.º Ciclo do Ensino Básico" (2015) e atualmente "Probótica - Programação e Robótica no Ensino Básico", promovidos pela Direção-Geral da Educação (2017).

Em comum as diferentes iniciativas utilizam recursos tecnológicos para o desenvolvimento das diferentes habilidades relacionadas com o Pensamento Computacional, por meio de linguagens de programação visual direcionadas para crianças, de jogos ou por meio de kits robóticos.

Sendo o estudo da relação entre a Programação e o desenvolvimento do Pensamento Computacional em alunos do 1.º ciclo do Ensino Básico, um tema recente em Portugal, este tema vai de encontro à nossa preocupação relativamente à utilização do jogo de computador no desenvolvimento de competências cognitivas, sociais e atitudinais, não numa perspetiva de uso de tecnologias de modo passivo pelo aluno, de treino e assimilação de conteúdos, mas sim, na promoção da sua utilização de modo estimulador da forma de pensar e como uma ferramenta ao serviço da resolução de problemas.

Para o contexto deste estudo, realizámos, entre novembro de 2018 e janeiro de 2019, uma revisão da literatura incluindo estudos publicados, com as seguintes palavras-chave: Pensamento Computacional, Kodu Game Lab, Programação, Aprendizagem, Ensino Básico.

Nas pesquisas realizadas com recurso à B-On, ao Researchgate, e ao Google Scholar, baseada nas palavras-chave anteriormente referidas, foram encontradas 23 correspondências em língua Inglesa. Incluímos ainda estudos distintos e privilegiamos aqueles com datas da publicação mais recentes, da última década, nomeadamente entre 2010 e 2017.

Para além das palavras-chave a seleção baseou-se nos seguintes critérios de inclusão:

- nível escolar - 1º ciclo do ensino básico e conteúdo académico - português, matemática, estudo do meio e expressões;

- programação - a criação de jogos que implicaram práticas e conceitos computacionais;
- resultados - os estudos tinham de fornecer os resultados das aprendizagens em forma de dados quantitativos, qualitativos ou ambos;
- indicação dos ambientes de aprendizagem;

Excluímos os artigos que não continham nenhuma informação sobre o desenho e metodologia do estudo ou sobre os resultados obtidos.

Após a leitura e análise dos artigos correspondentes, selecionamos um número total de oito artigos. A listagem dos artigos selecionados e revistos poderá ser encontrada na bibliografia do estudo.

A leitura e análise do conteúdo dos artigos revelou-se fundamental para conhecer o estado do conhecimento neste campo e ainda fornecer-nos uma linha de orientação útil para o ensino da Programação e para o desenvolvimento do Pensamento Computacional em alunos do 1.º ciclo do Ensino Básico.

A investigação teve como objetivos identificar os contributos de uma abordagem narrativa na exploração do ambiente computacional Kodu no desenvolvimento do pensamento computacional e em particular no que diz respeito a competências cognitivas, sociais e atitudinais por alunos do 3.º e 4.º ano de escolaridade que participaram em sessões organizadas no âmbito da Oferta Complementar de um Agrupamentos de Escolas no Alentejo.

A tese está organizada em quatro partes: a introdução, a revisão da literatura – com três secções - o pensamento computacional, crianças e programação e aprender a programar através de jogos digitais; uma segunda parte dedicada à metodologia do estudo, incluindo o desenho de investigação, a caracterização dos participantes, os procedimentos adotados durante a intervenção educativa e os métodos de análise de dados. Finalmente e numa quarta parte são apresentados os resultados obtidos durante a investigação e as discutidas as conclusões do estudo.

Espera-se que este estudo, apesar das limitações metodológicas decorrentes dos estudos de caso, possa contribuir para uma melhor compreensão dos processos de aprendizagem desenvolvidos por crianças e jovens quando envolvidos na criação de jogos digitais – neste caso com recurso ao ambiente Kodu e nos potenciais contributos deste tipo de atividades para o desenvolvimento do pensamento computacional.

O problema e os objetivos do estudo

A formação dos alunos não pode remeter-se somente à transmissão e aquisição de conteúdos, mas, deve também revelar preocupação com a necessidade de que o aluno aprenda a pensar, desenvolvendo competências de adaptação à mudança e à resolução de problemas.

Neste contexto, a Direção-Geral da Educação lançou um projeto-piloto, promovendo a "Iniciação de Programação no 1.º Ciclo do Ensino Básico". Este projeto-piloto surgiu da proposta feita aos estabelecimentos de ensino públicos de Portugal para participarem durante o ano letivo 2015/16, com os alunos dos terceiros e quartos anos de escolaridade, do 1.º Ciclo do Ensino Básico. O referido projeto visou a exploração de várias ferramentas e ambientes computacionais, como por exemplo: o processador de texto e a introdução de duas das mais famosas interfaces gráficas, designadas de: o Kodu e o Scratch, que funcionaram como principais ferramentas no ensino de resolução de problemas baseados na algoritmia e rudimentos de programação.

O projeto "Iniciação de Programação no 1.º Ciclo do Ensino Básico" permitiu aos alunos o contacto com um conjunto de ferramentas e experiências, potenciando ao mesmo tempo a criação

de novos ambientes de aprendizagem e aquisição de competências transversais às diferentes áreas curriculares. A iniciação da programação desde o 1.º Ciclo do Ensino Básico pretendeu promover o desenvolvimento de um conjunto de competências e capacidades nos alunos, nomeadamente: o trabalho em equipa, a estruturação e organização de ideias, a criatividade, o espírito crítico, a resolução de problemas, entre outros.

Decorrente deste projeto surge, o ano letivo 2017/2018, a iniciativa "Programação e Robótica no Ensino Básico", procurando identificar e contextualizar o que se pretende que os alunos desenvolvam, aprendendo programando, em projetos, ao criar histórias, animações e jogos e resolvendo desafios do quotidiano através da programação e da robótica, considerando diferentes cenários de aprendizagem suportados por metodologias ativas de ensino e de aprendizagem.

É, então, no âmbito deste enquadramento que se levanta o problema gerador deste estudo.

"Há um problema quando sentimos a necessidade de preencher o desvio entre uma situação de partida insatisfatória e uma situação de chegada desejável (…)." Gauthier (2003, p. 66)

Nesta perspetiva, o problema constitui-se como o potencial "saber desejável" que o investigador procura alcançar, encontrando a sua pertinência, não só, nas preocupações pessoais do pesquisador em relação à falta de atrativo das práticas pedagógicas ou a habitual orientação que se lhes dá, mas, também, nas do interesse do ensino e da sociedade em geral.

No estudo, pretende-se assim minimizar a distância entre o que sabemos e o que procuramos conhecer, isto é, aprofundar o estudo da utilização do software Kodu, em contexto educativo.

Se através da literatura conhecemos casos, fundamentados em investigações, que referenciam os jogos de computador como instrumentos potenciadores do pensamento computacional e motivadores da aprendizagem, paralelamente também se têm evidenciado estudos no sentido de revelarem uma relação muito estreita entre o desenvolvimento das crianças e os jogos de

computador. Deste modo, partindo destes factos enquanto fundamentos, no âmbito deste trabalho foram definidos os seguintes objetivos específicos:

1) Desenhar, implementar e avaliar uma estratégia de trabalho educativo, em contexto de Oferta Complementar, centrada na criação de jogos com recurso ao Kodu Game Lab.
2) Identificar as competências cognitivas, sociais e atitudinais que um grupo de alunos do ensino básico (8-10 anos) desenvolve através da criação de jogos no ambiente de jogo Kodu Game Lab.

Esta dissertação, em termo gerais pretende por isso, através de um estudo de caso, investigar a utilização do software Kodu, em contexto educativo por alunos do ensino básico, numa turma envolvida em trabalho educativo no âmbito da Oferta Complementar de Escola.

Com o objetivo de fornecer uma direção ao processo investigativo, formulámos a seguinte pergunta de investigação:

Quais as competências cognitivas, sociais e atitudinais que um grupo de alunos do ensino básico (8-10 anos), podem desenvolver (e demonstrar) decorrentes da implementação em contexto de sala de aula, de uma estratégia baseada na criação de jogos no ambiente de jogo Kodu Game Lab?

Pensamento Computacional e Programação para Crianças: breve revisão da literatura

A tecnologia está cada vez mais presente nas nossas vidas. Numa época de grandes transformações sociais, económicas, políticas, culturais, e ideológicas, a noção de "sociedade do conhecimento" parece modificar-se. Surgem novas formas de pensar, de conviver em sociedade e torna-se necessário compreender toda essa mutação contemporânea para então atuar nela. Conforme afirma Prensky (2001) "Os nossos estudantes mudaram radicalmente. Os estudantes da atualidade já não são as pessoas para quem o sistema educativo foi projetado".

No mundo em que vivemos, há espaço para as tecnologias que existem e para as que ainda estão em desenvolvimento. Deparamo-nos a todo o instante com tecnologias por toda a parte. Crianças e adolescentes com dispositivos móveis, smartphones e tablets, escrevendo rapidamente mensagens de texto.

Uma criança de apenas 7 anos consegue facilmente aceder a jogos, ensinar outras pessoas a utilizarem o ambiente virtual, e também manipular os dispositivos de forma mais rápida do que podemos inferir.

Foi para esta geração, a geração tecnológica que Prensky (2001) criou o termo nativo digital. O termo nativo digital foi sugerido para designar os nascidos a partir de 1990 e que apresentam características como, familiaridade com o computador, com os recursos da internet e a capacidade de receber informações rapidamente, processar vários assuntos simultaneamente e desempenhar múltiplas tarefas.

Por tudo isto o conceito de pensamento computacional é importante para a nossa geração, a geração dos alunos e cada vez mais importante para as gerações vindouras.

É neste contexto que surgem três desafios, inspirados no lema do Projeto Minerva: racionalizar, valorizar e atualizar (Ramos, 2016).

Ramos (2016) defende que a introdução do pensamento computacional nas escolas do ensino básico deverá ser assente num rationale que permita aos alunos a construção de um conhecimento baseado não apenas na experiência mas também no conhecimento informado, nomeadamente o conhecimento pedagógico e científico disponível nos domínios relevantes com particular destaque para as áreas da Educação, Sociologia, Psicologia, Filosofia, Ciências da Computação, Inteligência Artificial, Linguística, Neurociências, Antropologia, entre outras áreas e domínios interdisciplinares. O investigador apresenta-nos a valorização como segunda premissa do pensamento computacional enquanto competência da educação do século XXI. Apesar do conceito ser predominantemente cognitivo, o pensamento computacional apresenta ainda a possibilidade de desenvolver competências de aprendizagem sociais, culturais e emocionais, "promovendo a aprendizagem em rede e a colaboração com os outros, através de processos eminentemente sociais e coletivos, tendo como fim último o da educação das novas gerações promovendo os valores universais e humanistas próprios das sociedades democráticas avançadas, onde os indivíduos possam realizar a sua plenitude, de capacidades singulares e únicas no meio social onde crescem e se desenvolvem como seres humanos". (Ramos, 2016).

A terceira e última premissa apresentada por Ramos (2016), prende-se com a atualização dos espaços curriculares de forma a incluírem o pensamento computacional, a programação e as ciências da computação nos 1º e 2º ciclos do ensino básico, em Portugal. É sustentada a teoria de que a conceção, desenho e implementação de iniciativas de introdução ao pensamento

computacional e ou de iniciação à programação nas escolas do ensino básico deverão ter como motivação o contributo positivo, inspirador e educativo que poderão proporcionar aos alunos, quer estas iniciativas se desenvolvam em atividades curriculares ou extracurriculares. Importa salientar que o desenvolvimento do pensamento computacional surge, na perspetiva de Ramos, como componente da literacia digital, tal como a programação e a robótica, e deverão apresentar-se como recursos para atingir os objetivos delineados nas iniciativas educativas desenvolvidas.

A necessidade de se incluir o pensamento computacional na escola e nas iniciativas educativas vem ao encontro de uma escola que possibilite a criatividade e o desenvolvimento de competências que se aproximem da realidade do aluno do Séc. XXI. Dessa demanda emerge a necessidade de se pensar a docência nomeadamente a necessidade de incorporar experiências de aprendizagem adequadas à introdução e ao desenvolvimento do pensamento computacional, no quadro da formação inicial de professores, em todas as áreas curriculares, e não apenas nas áreas do ensino da informática ou computação (Barr, 2011), citado por Ramos (2014).

O pensamento computacional tem sido objeto de alguma investigação em Portugal e no Brasil utilizando principalmente a linguagem de programação Scratch e o site Code.org.

A investigação promovida por Ana Ventura no âmbito da sua Dissertação de Mestrado pretendeu estudar o potencial da utilização da linguagem de programação, Scratch, no ensino da geometria, de forma a intensificar a abstração necessária à compreensão das propriedades das figuras geométricas. A investigação envolveu uma turma de 20 estudantes, do 1º Ciclo do Ensino Básico (CEB), 2º ano de escolaridade, de uma escola de ensino privado do Porto. O estudo assumiu uma abordagem mista, não privilegiando qualquer das componentes qualitativa ou quantitativa. Para a recolha de dados, o uso de notas de campo possibilitou aferir os conhecimentos adquiridos pelos estudantes, e a elaboração de um inquérito por questionário, permitiu uma análise estatística

descritiva do grau de satisfação dos estudantes sobre o recurso usado. Tendo em conta as respostas ao inquérito por questionário, na afirmação "O Scratch ajudou-me na aprendizagem da geometria" 94% dos estudantes respondeu afirmativamente. Na afirmação "Gostava que o Scratch substituise o manual escolar" 61% dos estudantes responderam sim, 17% responderam não sei e 22% responderam não. Na afirmação "Foi fácil trabalhar com os diferentes blocos no Scratch" 89% responderam sim e 11% responderam não saber. Na afirmação "Gostava que o Scratch fosse utilizado mais vezes nas aulas" 89% responderam sim e 11% responderam não saber. Na afirmação "Gostei das aulas em que o Scratch foi utilizado" a resposta foi unânime, 100% dos estudantes responderam afirmativamente. (Ventura, 2017)

Tendo em consideração as respostas dos alunos, a grande maioria confirma que o software Scratch foi uma ferramenta importante de apoio no processo de ensino - aprendizagem. No entanto, é de salientar que, apesar da maioria pretender que o software substitua o manual, alguns afirmam não querer, o que poderá revelar pouco à vontade na utilização das TIC.

É de ter em conta, ainda, a opinião dos estudantes na habilidade de trabalhar com os diferentes blocos, os quais afirmaram ser fácil. Da análise realizada conclui-se também que os alunos que receiam e não se sentem perfeitamente habilitados em manipular os blocos, não têm a certeza de quererem manter o Scratch mais vezes nas aulas, o que poderá traduzir a insegurança e o medo de arriscar. No entanto, todos os alunos gostaram das aulas em que o Scratch foi utilizado. A partir deste estudo pode concluir-se que os conteúdos, de forma geral, foram assimilados. Tal como afirma a professora titular de turma "(…) a aprendizagem é mais significativa quando os estudantes experimentam e constroem o conhecimento, e o Scratch permite isso mesmo". A docente considera ainda que trabalhar conteúdos curriculares com o Scratch inova as práticas docentes e permite que os alunos reflitam, analisem criticamente e desenvolvam o pensamento lógico.

O desenho de interface para o desenvolvimento do pensamento computacional no Ensino Básico: análise do Scratch" foi o tema escolhido para a Dissertação de Mestrado de Sarita Costa. A seleção do Scratch, pela aluna da Universidade Nova, foi motivada pela popularidade do software entre os estudantes e pela boa avaliação entre os educadores. Com este estudo a aluna/investigadora pretende responder à questão norteadora da dissertação: "Como desenhar interfaces de softwares educativos para o desenvolvimento do pensamento computacional no Ensino Básico?".

A análise realizada ao Scratch indica que os seus criadores adotaram a metodologia de desenho centrado no aprendiz para ajudar crianças e jovens a aprender a programar e, por conseguinte, adquirir fluência digital e desenvolver o pensamento computacional. (Costa, 2016)

A linguagem de programação do Scratch permite explorar as três dimensões do pensamento computacional, conceitos, práticas e perspetivas, baseadas no quadro referencial de Brennan e Resnick (2012).

Foi observado, neste estudo, que cada componente da interface do programa pode ser explicado por referências do design, do construcionismo e das ciências da computação. A combinação dessas fontes e o profundo conhecimento da audiência por meio de estudos empíricos resultaram numa plataforma de construção de projetos que oferece as ferramentas necessárias para a criação, a experimentação e a partilha de artefactos digitais.

A aluna/investigadora chega mesmo a comparar o ambiente de jogo Scratch a uma caixa de ferramentas que está sempre aberta para que os aprendizes identifiquem rapidamente os recursos disponíveis. Dessa forma, os autores evitam a carga cognitiva extrínseca ao objetivo de aprendizagem e permitem que os aprendizes concentrem-se na construção de projetos. (Costa, 2016)

A investigação, baseada em vários estudos, confirma que a escolha da ferramenta Scratch é uma opção válida para introdução ao pensamento computacional no ensino básico. A avaliação positiva da comunidade académica e a interação dos alunos, confirmada pelo número de projetos criados e compartilhados (13 milhões), indicam que os autores do Scratch estão a alcançar os objetivos educativos, lúdicos e sociais propostos pela ferramenta (Costa, 2016). Conclui-se que o Scratch permite a aprendizagem de práticas, conceitos e perspetivas do pensamento computacional, está alinhado com a teoria construcionista da aprendizagem e adota uma linguagem de programação visual para estimular a participação de crianças a partir de 8 anos de idade.

A investigadora termina com a esperança de que a partir do seu trabalho novas pesquisas surjam, sobre outras dimensões do desenho de interfaces de aprendizagem que não foram aprofundados no seu trabalho. É importante que os pesquisadores e profissionais também observem ferramentas de introdução ao pensamento computacional que não utilizam o recurso da programação. A análise de boas práticas pode estimular a criação de novos ambientes computacionais com outras estratégias de promoção da aprendizagem (Costa, 2016).

A análise de um artigo de Martins et al. (2016), com o tema, "Inserção da programação no ensino fundamental: Uma análise do jogo Labirinto Clássico da Code.org através de um modelo de avaliação de jogos educacionais", que envolveu 168 alunos do ensino fundamental na experiência do uso do jogo Labirinto Clássico, disponível na plataforma Code.org, permitiu-nos observar um aumento no nível de conhecimento após o uso do Labirinto Clássico. Este estudo conclui que o Labirinto Clássico oferece uma oportunidade para a aprendizagem da lógica de programação, mesmo para os alunos que possuem pouca experiência com os conceitos abordados. Através dos resultados obtidos, observa-se que os alunos tiveram a oportunidade de aprender e praticar conceitos dos quais não tinham um entendimento pleno. Por se tratar de uma ferramenta gratuita

e com um grande número de jogos disponíveis, a plataforma Code.org aparece como uma grande aliada na disseminação do ensino de computação desde o ensino básico. O estudo realizado demonstrou que o jogo foi uma ótima experiência para os alunos, aliando a motivação com a aprendizagem. A linguagem utilizada no jogo é adaptada ao público-alvo e os elementos de controlo deixam a atividade fluida e de fácil entendimento. O impacto alcançado pelos objetivos de aprendizagem do jogo demonstra que a maioria dos alunos não possuía experiência prévia com os conceitos abordados, ainda assim indicaram um aumento no seu conhecimento sobre programação após o uso da ferramenta.

Cavalcante (2016) realizou uma monografia subordinada ao tema "Pensamento computacional e programação introdutória: um estudo de caso sobre competências desenvolvidas na programação em blocos com o code.org", para obtenção do grau de Licenciado em Ciências da Computação pelo Centro de Ciências Aplicadas e Educação (CCAE), Campus IV da Universidade Federal da Paraíba.

O seu trabalho procurou compreender a relação entre a programação e o pensamento computacional, investigando a aprendizagem da programação introdutória com a plataforma Code.org e o desenvolvimento do pensamento computacional em alunos do ensino médio. Para isso foi realizado um estudo de caso na plataforma para identificar as competências desenvolvidas pelos alunos através das atividades de Programação em Blocos ou Programação Visual. Também foi realizado um teste experimental numa oficina de programação para identificar que habilidades foram desenvolvidas pelos alunos participantes. Os resultados do estudo de caso demonstraram que há um conjunto de conceitos e práticas computacionais desenvolvidas em atividades de programação na plataforma Code.org que podem ser compreendidas como competências do pensamento computacional. É demonstrado neste estudo de caso que as competências

estabelecidas na *framework* de Brennan e Resnick (2012) podem ser aplicadas na plataforma Code.org, nomeadamente teste e depuração, reutilização e reformulação, modularização, eficiência e reconhecimento de padrões. Concluiu-se também que as perspetivas computacionais estão relacionadas com as atitudes dos alunos durante a programação. Essas atitudes são relativas ao modo de se expressar, conectar, questionar e ao modo de agir colaborativamente.

Com base nos trabalhos apresentados constámos que o quadro referencial de Brennan e Resnick (2012) se encontra presente tanto no ambiente de jogo Scratch como no site Code.org. Nestes ambientes computacionais identificam-se três dimensões que, segundo os investigadores, estão envolvidas no pensamento computacional: conceitos computacionais (conceitos utilizados na definição de programas, como interação, paralelismo, condicionais), práticas computacionais (práticas de como desenvolver programas, como ser incremental ou interativo, depurar, reusar), e perspetivas computacionais (perspetivas que o programador desenvolve sobre o mundo à sua volta e sobre si mesmo, como capacidade de expressão e de conexão).

O Pensamento Computacional na Escola e na Aprendizagem

Defende-se que, num mundo permeado pela tecnologia computacional, a educação em todos os níveis deveria desenvolver os mecanismos necessários para que os alunos possam não apenas utilizar a tecnologia, mas também compreendê-la e serem capazes de implementar soluções para problemas utilizando recursos computacionais (CSTA, 2011).

O termo pensamento computacional surgiu em março de 2006, através de Jeannette Wing, apesar de Seymour Papert já o ter utilizado em 1980 em Mindstorms. Wing publicou o artigo

Computational Thinking para defender a ideia de que o pensamento computacional deveria ser incluído na formação de todos os alunos por ser uma aptidão fundamental para cidadãos do século XXI, tão importante quanto ler, escrever e fazer operações aritméticas. A autora afirma que o pensamento computacional é uma abordagem para a resolução de problemas, conceção de sistemas e compreensão do comportamento humano que se baseia em conceitos fundamentais das Ciências da Computação. Tem como estratégia o uso de abstrações e decomposição de problemas complexos em partes mais simples. É a forma como as pessoas pensam de modo a resolver problemas com a parceria de um computador (Wing, 2006). O artigo de Jeannette Wing é considerado o propulsor da atual mobilização de educadores, entidades profissionais, empresas e pesquisadores das Ciências da Computação interessados em colocar em prática o desafio de ensinar a pensar computacionalmente.

Noutros artigos de Wing (2008), o pensamento computacional é definido como um processo de pensamento envolvido na formulação de problemas e expressão de suas soluções de tal forma que essas soluções possam ser eficazmente executadas por um agente de processamento de informação. A autora ressalta que o agente de processamento da informação pode ser uma máquina ou um ser humano. Este último ponto é importante. Em primeiro lugar, os seres humanos computam. Em segundo lugar, as pessoas podem aprender pensamento computacional sem recurso a uma máquina. Além disso, o pensamento computacional não é apenas sobre a resolução de problemas, mas também sobre a formulação do problema (Wing, 2014).

Com o objetivo de "desmistificar o pensamento computacional", Shute et al (2017) dedicaram-se a uma investigação com o objetivo de examinar o pensamento computacional na educação. A revisão da literatura relevante realizada apresentou uma diversidade de definições, intervenções, avaliação e modelos. Depois de sintetizar várias definições os investigadores selecionaram a

seguinte definição de trabalho de pensamento computacional: A fundação conceptual necessária para resolver problemas efetivamente e eficientemente (isto é, algoritmicamente, com ou sem a ajuda de computadores) com soluções que são reutilizáveis em contextos diferentes. Esta definição de pensamento computacional é, principalmente, uma forma de pensar e agir, que pode expor-se por meio do uso determinadas habilidades, que poderão tornar-se a base da avaliação baseada na realização de habilidades de pensamento computacional.

Baseados na literatura, os investigadores categorizam o pensamento computacional em seis vertentes principais: decomposição, abstração, desenho de algoritmo, depuração, repetição e generalização.

No entanto, apesar de concordarmos com Shute et al (2017) quando afirmam que a definição de pensamento computacional se desenvolve à medida que os pesquisadores começam a agregar conhecimento sobre o tema, tendo em consideração os objetivos desta dissertação, apoiamo-nos na definição operacional adotada pela International Society for Technology in Education (ISTE) e pela Computer Science Teachers Association (CSTA) que oferecem um quadro de referência útil para a incorporação do Pensamento Computacional em programas curriculares do ensino básico. Segundo esta definição, o pensamento computacional é um processo de resolução de problemas que consiste (mas não se limita) nas seguintes características:

- Formular problemas de forma que permita usar um computador e outras ferramentas para solucioná-los;
- Organizar e analisar dados de forma lógica;
- Representar dados através de abstrações, como modelos e simulações;
- Automatizar soluções através de pensamento algorítmico (uma série de passos ordenados);

- Identificar, analisar e implementar soluções possíveis com o objetivo de alcançar a mais eficiente combinação de passos e recursos;
- Generalizar e transferir esse processo de resolução de problemas a uma grande variedade de um mesmo tipo de problema. (CSTA, 2011).

ISTE e CSTA (2011) também apresentam as atitudes e predisposições necessárias para o desenvolvimento do conjunto de habilidades inerentes ao Pensamento Computacional:

- A confiança em lidar com a complexidade;
- Persistência em trabalhar com problemas difíceis;
- Tolerância à ambiguidade;
- A capacidade de lidar com problemas abertos e finitos;
- A capacidade de comunicar e trabalhar com outros para atingir um objetivo comum ou solução. (CSTA, 2011).

A CSTA (2011) elaborou diretrizes de currículo para o ensino de computação do Ensino Básico ao secundário que se transformaram numa das principais referências pedagógicas nessa área. O documento K–12 Computer Science Standards apresenta objetivos de aprendizagens organizados em três níveis, agrupados por anos de escolaridade do sistema educativo americano. De acordo com este modelo, o ensino da computação deve abordar: o pensamento computacional; a colaboração entre os estudantes; a prática da computação e programação; os computadores e dispositivos de comunicação; e os impactos éticos, globais e na comunidade.

Partindo das diretrizes curriculares que orientam a planificação de experiências de aprendizagem do pensamento computacional, os professores podem planificar aulas a partir de recursos

pedagógicos oferecidos por pesquisadores da área de educação em Ciências da Computação. Conhecer esses recursos é, também, imprescindível para o desenho de interfaces que ajudam os alunos a alcançar os objetivos de aprendizagem. Diferentes ferramentas tais como o Scratch, MIT App Inventor, Alice, Kodu, Blocky entre outras, já são utilizadas com essa finalidade. Embora a maioria tenha como foco o ensino da programação, oferecem funcionalidades que facilitam o ensino da aprendizagem de conceitos básicos do pensamento computacional, como abstração, pensamento algorítmico, decomposição, generalização, paralelismo e deteção sistemática de erros. Adicionalmente, habilidades como pensamento algorítmico, e a fluência digital podem apoiar-se. (Garneli et al, 2015)

Estes investigadores defendem que a educação poderá apoiar os alunos das mais variadas formas numa aprendizagem cuidadosamente projetada. Esta perspetiva não significa que todos os alunos se tornarão, necessariamente, programadores profissionais, mas que ganharão práticas úteis e habilidades da era digital.

O pensamento computacional dará assim a oportunidade ao aluno de se expressar e ser criativo. Assim como se ensina música, pintura e expressão escrita nas escolas, a programação também necessita de experimentação, de aprender fazendo. Da mesma forma que não imaginamos que todos os alunos serão músicos, pintores ou escritores, não podemos supor que todos venham a ser programadores profissionais, mas estes princípios básicos dão a oportunidade aos alunos para serem criativos e apreciarem a música, a pintura, a escrita e a programação como alternativas de expressão ao longo da vida.

A perspetiva de promover o pensamento computacional em crianças e jovens, possibilita também a proposta e o desenho de projetos multidisciplinares e a resolução de problemas colaborativamente nas mais diversas áreas de conhecimento. Artistas, filósofos, designers e

cientistas de todas as áreas podem colaborar na atividade intensamente criativa de resolução de problemas. O aluno de programação trabalha com outras áreas de conhecimento de modo a representar o problema de forma clara e não ambígua sendo assim possível ser representado e tratado por um computador. Esta capacidade de expressar o problema para que o computador possa resolver, caracteriza o pensamento computacional (Teixeira, 2017).

A revisão da literatura realizada no âmbito deste estudo, demonstrou duas tendências, nomeadamente: a definição e a caracterização de pensamento computacional, muito desenvolvido depois da definição inicial de Wing mas que conserva ainda a natureza inicial do conceito; a descrição de atividades bem-sucedidas que foram projetadas e executadas para introduzir e desenvolver o PC nas escolas e nos alunos.

Vários documentos demonstram, também, o potencial e as vantagens de introduzir PC na educação, já que permite a crianças e jovens, pensar de um modo diferente, resolvendo problemas, analisar questões diárias com uma perspetiva diferente para desenvolver a capacidade de descobrir, criar e inovar, para compreender o que a tecnologia tem para oferecer.

Várias ferramentas de programação foram utilizadas nas aulas de programação dos alunos, nos estudos analisados. No entanto, para o estudo, foram selecionados os contextos educativos utilizados para motivar e realçar a aprendizagem da programação de forma atrativa para os alunos. Contextos educativos com uma perspetiva construtivista facilitando a construção e aquisição de conhecimentos que utilizaram o ambiente de jogo Kodu Game Lab.

O principal mentor do construcionismo foi Seymour Papert que após estudar o desenvolvimento cognitivista com Piaget, ajudou a fundar o Laboratório de Inteligência Artificial do Instituto Tecnológico de Massachusetts – MIT. Foi nesse ambiente que desenvolveu, em 1967, a primeira versão da linguagem de programação Logo, desenhada especialmente para crianças.

A experiência com o Logo foi fundamental para o desenvolvimento da teoria construcionista nas décadas seguintes. Para explicá-la, é necessário fazer uma distinção entre a teoria de Papert e o construtivismo. Ambas compartilham a noção de aprendizagem como construção de estruturas de conhecimento, independente das circunstâncias de aprendizagem. Porém, o construcionismo acrescenta a ideia de que a aprendizagem será facilitada num contexto em que o aluno esteja conscientemente envolvido na construção de algo, "seja um castelo de areia na praia ou uma teoria do universo" (Papert, 1991). A valorização do envolvimento dos alunos em atividades com as quais se identificam é, portanto, um princípio fundamental nesta teoria.

Segundo Papert o conceito de construcionismo é compreendido como "aprender-fazendo" ("learning-by-making").

A abordagem construcionista vê a aprendizagem como a construção de relações entre o velho e o novo conhecimento por meio da criação de artefactos socialmente relevantes (Kafai, 2006).

O construcionismo privilegia os caminhos individuais ao invés do melhor caminho. Na fase de formação de conceito é melhor que se mantenha a mente aberta até que o aluno formule e reformule o seu próprio conceito pois desta forma as diferenças são respeitadas (Papert, 1991).

Nesta perspetiva, as novas tecnologias têm a vantagem de aumentar a variedade de atividades que podem ser realizadas pelos alunos. Os softwares educativos podem ser "objects-to-think-with" que ajudam os alunos a refletir sobre o seu desempenho.

A criação de jogos é o exemplo principal da atividade construcionista, especialmente na era dos jogos digitais.

De acordo com Kafai (1995), a aprendizagem, durante o processo de construção ativo, torna-se mais pessoal e atrativa, movendo-se além da aprendizagem mecânica, tornando-se mais significativa, unida e eficaz.

Kafai (2015) defende três dimensões da aprendizagem construcionista, a dimensão pessoal, social e cultural.

Analisando a primeira categoria de aprendizagem, a dimensão pessoal, esta encontra-se focada na apropriação do conhecimento e transformação que está representada no processo de criação de jogos. Os benefícios concentram-se na aprendizagem da programação, mas também incluem outros conteúdos académicos e habilidades como vários assuntos, resolução de problemas e habilidades (Grover & Pea, 2013). Em segundo lugar, os benefícios sociais. Concentram-se em várias formas da colaboração implicadas na construção de jogos de aprendizagem, se trabalha com outros no design ou é sobre a partilha e troca de projetos (Grimes & Fields, 2015). Finalmente, a terceira dimensão examina os benefícios e limites culturais que circunscrevem a participação em atividades de criação de jogos digitais. (Margolis et al., 2008).

Segundo o autor e de acordo com a visão de Piaget, o foco primário do construcionismo examina a aprendizagem a partir de uma perspetiva pessoal. Papert viu o compromisso com a programação Logo como um modo de facilitar a construção de estruturas de conhecimento, o que denominou de «apropriação» para que os alunos pudessem construir o seu próprio conhecimento e começar a identificar-se pessoalmente com ele.

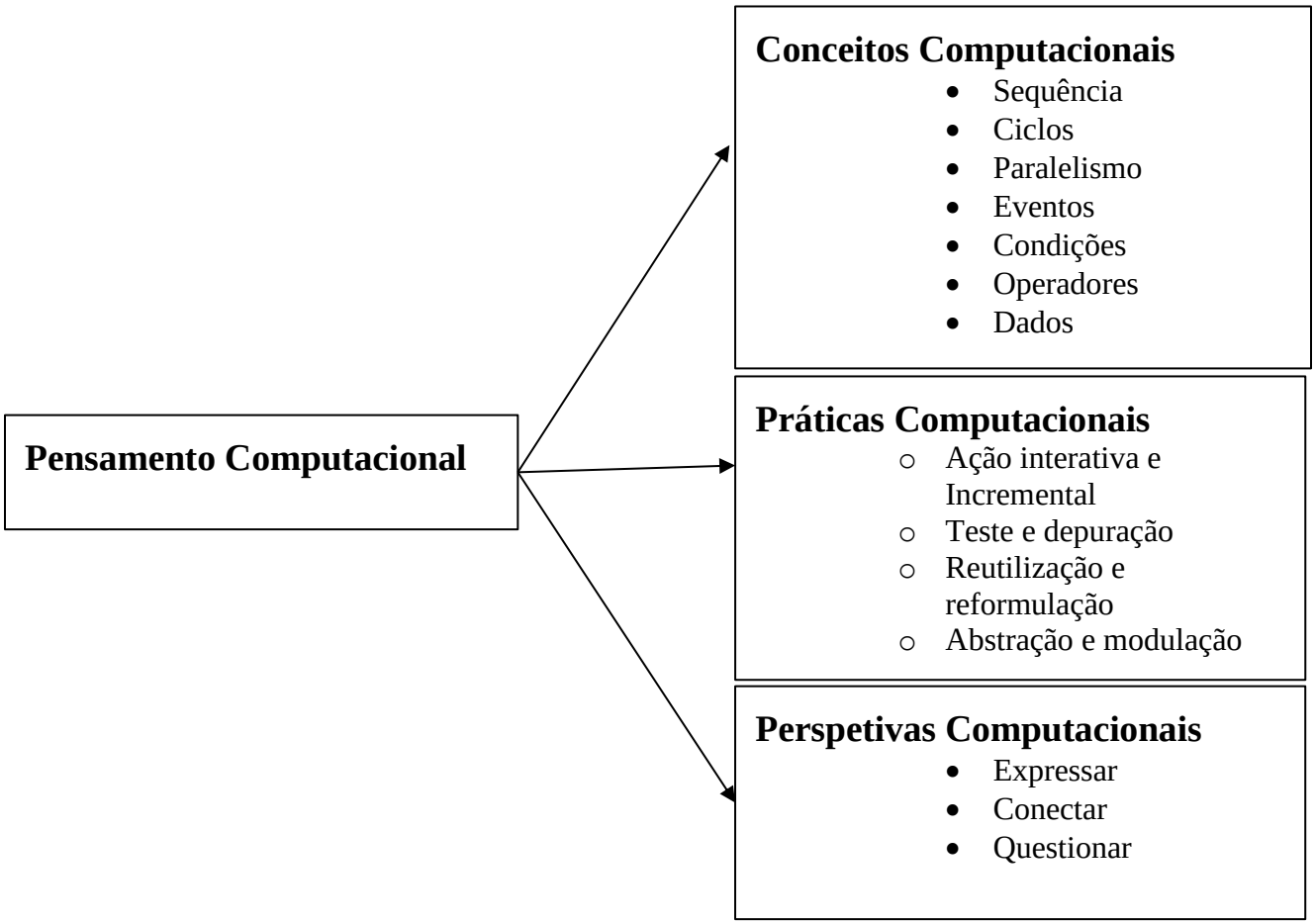
Na sua aproximação ao ato de jogar Kafai e Burke (2014) basearam-se nos fundamentos da teoria construcionista para entender como os alunos podem aprender no processo do desenho e construção de jogos através da programação. Foi criado o termo "participação computacional" para capturar estas diferentes dimensões, em que desenho e implementação de sistemas não são exclusivos à função de pensamento algorítmico, mas fundamentalmente representativo das práticas e perspetivas necessárias para contribuir dentro das redes sociais e compreender a natureza cultural e social de uma sociedade em rede.

Embora a codificação tenha recebido a maioria da atenção por incluir desenho de software, práticas de programação, depuração, e de código, estas práticas capturam o que se descreveu como "pensamento computacional", que Wing (2006) definiu como desenho de sistemas para uma solução eficaz de problemas com computadores. Apesar do pensamento computacional não ser apenas código, o código representa uma das avenidas-chave para envolver os jovens numa primeira compreensão sobre como os sistemas eficazes são concebidos e mantidos, um jogo de habilidades que pode aplicar-se a campos tão diversos como mecânica industrial, biologia computacional, e *marketing* analítico. (Kafai & Burke, 2015)

Um caminho útil para operacionalizar o pensamento computacional envolvido nas atividades de programação foi proposto por Brennan e Resnick (2012), que distinguiu conceitos computacionais, práticas e perspetivas.

A Figura 1 apresenta um esquema resumo das dimensões do pensamento computacional proposto por (Brennan e Resnick, 2012)

Figura 1 Dimensões do Pensamento Computacional



Os conceitos computacionais referem-se a elementos como sequências, ciclos, paralelismo, eventos, condicionais, operadores e a estrutura de dados que estão presentes em muitas linguagens de programação.

Para realizar estas conceções, os estudantes tiveram que dedicar-se a práticas computacionais que são atividades adicionais, reutilizando e misturando, testando e depurando, modularizando e abstraindo.

Finalmente, perspetivas computacionais como expressão, conexão e interrogação referem-se a conceções do mundo que se desenvolvem à medida que se envolvem com meios digitais.

Kafai e Burke (2015) conduziram um estudo/pesquisa acerca dos benefícios da construção de jogos construcionistas. Este estudo/pesquisa foi realizado entre 2014 e 2015 com recurso ao ERIC, ao Google scholar e a pesquisas realizadas por Kafai para o seu livro "Minds in Play" (1995). Foram consultados os arquivos dos jornais Computers & Education, Games & Culture e International Journal of Learning and Media. Foram consultados ainda os arquivos das conferências Digital Games Research Association e Games, Learning and Society. A ACM Digital Library também serviu para pesquisa. Foram incluídos apenas os estudos mais recentes (da última década).

Alguns estudos enquadraram os seus resultados no construcionismo, ao passo que outros utilizaram a criação de jogos para aprender, como um contexto para estudar a resolução de problemas, matérias académicas, ou outras habilidades.

Dentro da avaliação da aprendizagem dos alunos de programação, observou-se uma variedade de conceitos computacionais ou práticas tais como depuração e remistura, ao passo que outros examinaram a natureza da resolução de problemas ou planeamento implicado na criação jogos.

Os resultados do estudo de criação de jogos para aprendizagem, embora maioritariamente positivos, também levantaram várias inquietações:

a) o envolvimento no desenho de jogo;
b) o esboço da aprendizagem com e sobre computação;
c) a recolha de dados;
d) a ausência de conclusões negativas;
e) a escassez de estudos focados na aprendizagem colaborativa;
f) a ausência de oportunidades online.

Um dos desafios óbvios em unir as conclusões de tal conjunto diversificado de estudos é a grande variedade de contextos, ferramentas, idade dos alunos e períodos de tempo usados para fazer jogos. Uma consequência surpreendente de tal diversidade é que os jogos gerados e os benefícios de aprendizagem variam consideravelmente em forma e substância.

Obviamente que um jogo projetado em 2 horas em comparação com um que levou semanas de desenvolvimento terá resultados de aprendizagem diferentes e terá como base projetos de pesquisa diferentes.

Apesar dos prazos mais curtos oferecerem oportunidades para projetos experimentais que avaliam resultados particulares (Vos et al., 2011), os períodos mais longos oferecem oportunidades para a compreensão do desenvolvimento de interesses, conhecimentos e habilidades individuais das crianças (Peppler & Kafai, 2007)

Espera-se que os dois formatos continuem a ser utilizados em futuras pesquisas para aprofundar a compreensão da aprendizagem na criação de jogo.

Embora tenha sido empregue um meta-quadro desenvolvido por Brennan e Resnick (2012) para capturar a aprendizagem de conceitos computacionais, práticas e perspetivas na criação de jogo, há ainda um trabalho considerável e necessário em como se conceituar e avaliar o pensamento computacional ou se se pretende adotar uma visão mais expansiva da participação computacional (Kafai & Burke, 2014), que não se concentre apenas na área pessoal mas que também inclua dimensões sociais e culturais da aprendizagem — dois aspetos que foram subestimados na pesquisa mas que são essenciais para a compreensão e desenho de oportunidades de aprendizagem produtivas e sustentadoras na criação de jogo.

Outra área adequada para mais documentação e investigação será como os planos colaborativos podem realçar novas oportunidades de aprendizagem no jogo construcionista. Sabe-se, a partir da

pesquisa realizada nos jogos, como as colaborações devem motivar e sustentar esforços dos alunos para avançar no jogo.

À exceção da pesquisa colaborativa de programação na construção de jogos (Denner & Werner, 2007), poucos estudos incluíram a aprendizagem colaborativa nos seus projetos.

Foi também observada uma ausência de estudos específicos para identificar oportunidades online, de fazer e partilhar jogos. Programas como Scratch, Alice e Kodu — considerados como «os instrumentos» — redefiniram-se como «comunidades». Migraram do software autónomo para aplicações online em tempo real e, no processo, reformularam práticas de literacia atuais para as comunidades "faça você mesmo - do it yourself", *aumentando* de pensamento computacional para participação computacional (Kafai & Burke, 2014).

Estimular colaborações e comunidades de designers de jogo, online e offline, nos primeiros anos de escolaridade, onde as crianças projetam os seus primeiros jogos individuais, às enormes comunidades online, onde os jogos são alguns dos desenhos mais populares compartilhados é outro desafio-chave que motiva as crianças a fazer partilhas.

Ganhar acesso a uma ampla e apreciada comunidade significa que as crianças têm a oportunidade de alavancar aquela comunidade como uma extensão do próprio instrumento, com um feedback significativo para ajudar os jovens desenhistas a ganhar um apoio nos trabalhos de desenho de jogo, ao passo que os desenhistas mais experientes podem crescer na proficiência e criar jogos cada vez mais complexos. Mas antes de esta comunidade poder alavancar-se, os jovens devem ter primeiro a confiança para compartilhar o seu próprio trabalho, bem como aprender a natureza e respeitar o papel de feedback construtivo.

Em última análise, o objetivo deste estudo é promover ambientes que sejam estimulantes para a aprendizagem, e é aqui onde abordagens construcionistas juntam esforços instrucionistas e onde

se pode construir um caso para "jogos conectados", uma abordagem que não traça limites entre os jogadores e designers enquanto participantes dos meios digitais mas vê-os como complementares uns aos outros, como Papert (1995) já havia imaginado:

"Se uma pessoa pertence a uma cultura em que os jogos de vídeo são importantes, transformando-se de um consumidor para um produtor de jogos pode muito bem ser uma forma ainda mais poderosa para algumas crianças encontrarem a importância do que estão a fazer" Papert (1995).

Crianças, Jogos e Programação

A utilização de jogos digitais no desenvolvimento de competências relacionadas com o raciocínio lógico e a Lógica de Programação é uma estratégia que vem sendo bastante discutida, devido ao forte apelo desse tipo de sistema interativo junto à geração que hoje frequenta o ensino, caracterizada por Prensky (2001) como nativos digitais.

Neste estudo, procuramos destacar, sob a ótica do pensamento computacional e do Construcionismo, uma abordagem relativa ao uso de jogos digitais na Educação onde os alunos criam seus próprios jogos e aprendem através desta experiência.

Destacamos ainda a possibilidade de uma aprendizagem significativa através do desenvolvimento de um artefacto por parte dos alunos, favorecendo a construção do conhecimento, segundo a teoria do Construcionismo (Papert, 1994), ao invés da simples memorização de informações. Destacamos também o desenvolvimento do conhecimento em áreas específicas, como a Matemática e a Literatura, entre outras. Desta forma, defendemos, como outros pesquisadores (Kafai, 2006), que o grande potencial dos jogos digitais na Educação está nesta criação de jogos

por parte dos alunos, favorecendo uma aprendizagem contextualizada e crítica. Assim, permite-se aos alunos, tanto a construção de conhecimentos em campos específicos quanto o desenvolvimento de competências importantes para a vida na sociedade atual.

Para que possamos explorar o potencial educacional dos jogos digitais, é necessário conhecer a natureza destes artefactos, as suas especificidades, assim como as especificidades do processo educacional. Da mesma forma, é importante compreender o contexto no qual a integração entre jogos e ensino será proposta – quem são os alunos, os professores, como é o ambiente – assim como quais são os resultados esperados – espera-se construir conhecimento relativo à uma disciplina específica, desenvolver competências globais ou ambos?

Kafai (2006) argumenta que o grande potencial educacional dos jogos digitais não está no ato de jogá-los, mas sim no processo de criação, defendendo, que esse tipo de iniciativa exploraria mais aprofundadamente a relação entre jogos digitais e aprendizagem.

Nesta abordagem, os próprios alunos são responsáveis pela criação dos seus jogos e, durante este processo, podem desenvolver tanto conhecimentos em áreas específicas como diferentes competências – como autonomia, capacidade de solucionar problemas e domínio das tecnologias digitais.

É possível argumentar que a construção de jogos digitais em contexto escolar possui, na sua essência, uma grande influência da Teoria Construcionista de Papert (1985, 1991, 1994). Nela, o autor defende que a aprendizagem deve ser ativa e que o aluno deve ser responsável pelo seu próprio processo de construção do conhecimento, através da elaboração e consequente teste de teorias para as suas perceções e ideias, num processo cíclico.

Papert (1985, 1991, 1993) defende ainda, como vimos anteriormente, que esta construção do conhecimento pode ocorrer de forma mais significativa através da construção de um artefacto por

parte do educando. O autor argumenta que, através da criação, os alunos desenvolvem autonomia em relação aos seus processos de aprendizagem, que ocorre assim de forma mais significativa, em detrimento à mera memorização de conteúdos descontextualizados.

Neste sentido algumas iniciativas foram realizadas recentemente e com resultados relevantes, nomeadamente, o Projeto de Programação a partir de Jogos, implementado em cinco escolas primárias gregas, envolvendo 138 alunos, o projeto "Computational Thinking For Children Education", implementado nas 4 turmas de escola primária da escola de Mazara del Vallo em Itália, envolvendo 81 alunos e o "Game Design and Learning" implementado como um programa depois das aulas, em 7 sessões de 3 horas cada ou num Campo de Férias, em 8 sessões de 5 horas cada, dirigido a crianças dos 9 aos 15 anos com o objetivo do ensino do "game-design", programação e resolução de problemas.

Sendo os jogos de computador, provavelmente, o meio de entretenimento mais comum dos jovens; crianças entre os oito e os dez anos e adolescentes (idades 5-15) jogam jogos usando vários dispositivos eletrónicos, (Britain, 2013), o Kodu Game Lab foi o software de programação selecionado em todos os projetos. Com um download gratuito, uma das mais-valias do Kodu é que permite aos alunos praticar conceitos de programação básicos, como as afirmações "if-then", variáveis e condicionais (Stolee & Fristoe 2011). Sendo o principal objetivo da utilização do software a criação de jogos, o processo de criar jogos digitais necessita, naturalmente, que os alunos entendam e trabalhem com conceitos das Ciências da Computação. Entre outras opções disponíveis (Scratch, Alice, GameMaker), o software Kodu destaca-se por várias razões. Em primeiro lugar, o ambiente de Kodu é tridimensional (3D), ao contrário de qualquer outro software de jogo (Scratch). Em comparação com os ambientes 2D, a capacidade de criar jogos 3D do Kodu torna-o visualmente mais apelativo para as crianças. Graças ao seu mecanismo de jogo em tempo

real 3D, a partir do seu primeiro encontro com o Kodu, até os alunos mais jovens podem criar jogos parecidos aos jogos que jogam regularmente (MacLaurin, 2011). O mecanismo de jogo 3D em tempo real ajuda os alunos a criar jogos que podem competir com os modernos jogos de consola, enquanto cumprem suposições de jogo comuns (física) sem qualquer esforço da parte do aluno. O Kodu também permite a criação de simulações. Finalmente, até com a sua interface de utilizador simplificada, o Kodu introduz os alunos a fundamentos de conceitos computacionais (Stolee & Fristoe, 2011) num contexto de desenho do jogo. Baseado na linguagem de programação visual, o Kodu alivia os alunos dos níveis de abstração necessários noutros softwares ou instrumentos de programação (MacLaurin, 2011).

Implementado e desenvolvido a partir da perspetiva construcionista de Papert, o Projeto que envolveu 138 alunos gregos abarcou a noção de que os alunos deveriam ser produtores bem como consumidores dos meios de comunicação digitais e que deveria ser-lhes dada a oportunidade de usar os computadores como um meio de expressão criativa para fazer um produto da relevância pessoal para eles.

O projeto teve a duração de praticamente um ano letivo (de início de setembro de 2016 a meados de maio de 2017), foi desenvolvido em trabalho a pares e tinha como principal objetivo examinar se o desenho de jogos digitais, por alunos, teve impacto na compreensão e capacidade de utilizar conceitos de programação. Foram então formuladas as seguintes hipóteses:

- Os alunos podem entender e efetivamente utilizar conceitos de programação básicos, como lógica Booleana, funções, condições, loops, valores e variáveis, quando projetam os seus próprios jogos digitais. Por conseguinte, os seus jogos são funcionais e jogáveis;

- Os resultados da aprendizagem, em termos de como correta e eficientemente os conceitos de programação se usam, e quão completos e funcionais os jogos são, dependem da abordagem de ensino que se utiliza;
- Os estudantes formam atitudes e perceções positivas quanto ao seu envolvimento nas atividades do jogo;
- A sua visão depende da abordagem docente utilizada.

Durante a implementação do Projeto os alunos projetaram os seus jogos, elaboraram o desenho de jogo e construíram o jogo com a programação dos personagens.

Este Projeto contribuiu para a literatura relevante porque classificou os construtos presentes na programação dos jogos dos alunos e verificou se usaram corretamente. Por outro lado, os resultados foram examinados com prudência, devido às características do Kodu.

Os conceitos de programação mais comumente utilizados foram condições, variáveis e sequências. A lógica booleana e as funções foram os conceitos menos usados.

Os conceitos mais problemáticos e onde os alunos cometeram erros de programação foram Lógica Booleana, sequências e funções.

Os conceitos menos problemáticos foram condições e variáveis.

Em conjunto, os resultados na programação de erros e o desenho de jogo, sugerem que, fazendo os seus jogos, os alunos participantes, em todos os grupos, conseguiram aprender e utilizar os conceitos de programação e práticas de desenho de jogo.

Um aspeto importante da autoria do jogo é "da aprendizagem pelo desenho do jogo" (Ke, 2014).

Para Kafai (2015), aprender pelo desenho do jogo é importante porque ajuda os estudantes a aprender como aprender; já que não existe uma solução única para os problemas de design de jogo, os estudantes podem escolher as suas próprias estratégias e soluções de tratar com a complexidade

da atividade de criação de jogo. A ausência de programação de assuntos técnicos significa que os estudantes não se vinculam a problemas relacionados com a aprendizagem de outras linguagens de programação (sintaxe e uso de símbolos), e podem concentrar-se na lógica da programação e nas habilidades necessárias para a planear e implementar os seus jogos. Além disso, é a simplicidade do Kodu que o torna um instrumento apropriado para principiantes. A escolha do trabalho em pares trouxe vários benefícios porque:

a) os parceiros são capazes de compartilhar ideias e tarefas completas colaborativamente, um aspeto importante da teoria construtivista (Kafai & Harel, 1991);
b) fomenta e sustenta o compromisso dos estudantes, importante para a realização, complexo e aberto às atividades (Kafai & Resnick, 1996);
c) permite aos estudantes construir o conhecimento entre e por si mesmos e proporciona uma fonte de suporte intelectual (Vygotsky, 1978);
d) considera-se que as explicações dos pares se combinam melhor às compreensões dos estudantes em comparação com outros recursos (Lewis, 2011);
e) os grupos maiores são menos flexíveis; os estudantes provavelmente terão sucesso em tarefas cognitivas quando trabalham em pares (Kutnick et al., 2005).

A criação de jogos de computador é uma forma da expressão criativa, que se relaciona com o divertimento (Buckingham & Burn, 2007). Um dos resultados positivos deste projeto foi que os alunos se sentiram motivados e gostaram de fazer jogos de computador. Nas palavras de Papert (1996), a experiência dos alunos foi “um grande divertimento”; os alunos preferem atividades desafiantes sobre não-desafiantes, com a condição de que sejam interessantes e pessoalmente relevantes.

Relativamente ao projeto “Computational Thinking For Children Education”, implementado nas 4 turmas de escola primária da escola de Mazara del Vallo em Itália, envolvendo 81 alunos, utilizaram-se diferentes instrumentos para recolha de dados. Um pré-teste e um pós-teste foram utilizados para descobrir diferenças na atitude em relação à programação de computador entre os alunos. A pesquisa foi extraída do "Conjunto de sala de aula - Kodu Game Lab”, traduzida e adaptada aos alunos em italiano. Outras informações acerca do desempenho dos alunos em língua italiana e matemática, informação socioeconómica e contexto familiar foram fornecidas diretamente pelas escolas.

Os resultados apresentados neste estudo fornecem as perceções preliminares em como a programação de computador de alunos de escola primária - considerado como um facilitador importante de habilidades de pensamento computacional - pode melhorar-se através de sessões de codificação práticas.

A utilização de um instrumento de programação visual e a adoção de uma abordagem narrativa estimularam uma perceção positiva sobre a programação de computador nos alunos; especificamente, como destacado pela comparação entre o pré e o pós teste, os alunos entenderam a programação como algo engraçado, fresco, excitante, criativo e “algo onde eu seria bom”.

No que concerne a alunos de escola primária, a literatura acentua a relevância de estratégias de narrativa para estimular a aprendizagem de processos (Szurmak & Mindy, 2013).

Relativamente ao “Game Design and Learning” implementado como um programa depois das aulas em 7 sessões de 3 horas cada ou campo de férias em 8 sessões de 5 horas cada, dirigido a crianças dos 9 aos 15 anos com o objetivo do ensino do “game-design”, programação e resolução de problemas apresentou como objetivos:

- Ensinar e desenvolver o desenho do jogo;

- Programação;
- Resolução de problemas.
- Foram desenvolvidas as seguintes atividades:
- Desenho do jogo;
- Abordagem à resolução de problemas;
- Resolução de problemas;
- Desenho livre.

Tendo em conta que, na última década, as tarefas digitais de desenho do jogo atraíram a atenção como tarefas atraentes de ensinar às crianças habilidades complexas de pensamento (Akcaoglu 2014; Denner, 2012; Hwang, 2013), a atratividade de tarefas de desenho do jogo como atividades educativas pode explicar-se por vários fatores:

- Os jogos são inerentemente atraentes para crianças. (Gee, 2003)
- Os jogos são sistemas complexos, compostos de muitas variáveis relacionadas. (Fullerton, 2008; Robertson, 2012)
- O processo de design do jogo, como uma tarefa de desenho, requer a ativação de habilidades metacognitivas importantes, tornando-o num contexto ideal para a prática de resolução de problemas. (Ke, 2014)
- Durante o processo de design as crianças adquirem a possibilidade de criar representações externas (visuais) das suas ideias abstratas de outra forma. (Baytak, 2010).
- A exteriorização das representações mentais é um aspeto importante do processo de design e da resolução de problemas. (Bonnardel, 2010)

- Durante a criação de jogos os alunos têm a oportunidade de se envolver na programação (Denner et al. 2012).
- Especialmente com os novos interfaces de programação, projetos de software do jogo de nova geração ("Microsoft Kodu" 2012; "Scratch", 2012) são intuitivos até ao ponto de serem utilizados facilmente por crianças de nove ou dez anos. (Fowler, 2011)

Por estas razões, as tarefas de desenho do jogo são divertidas, autênticas, e apropriadas para ensinar desenho e habilidades de resolução de problemas a jovens crianças.

A partir da implementação deste projeto concluímos que o processo de design detalhado aqui pode servir de exemplo noutros contextos onde as diferentes tarefas se utilizam para ensinar tipos diferentes de habilidades de pensamento (pensamento computacional).

Neste sentido, o contexto "game design and learning" (GDL) pode considerar-se como um caso eficaz de integração da tecnologia, onde a tecnologia se utilizou para resolver um problema educativo (isto é, o ensino da resolução de problemas num contexto atraente).

Deve observar-se, que os alunos que participaram no programa GDL mostraram uma melhoria significativa nas suas habilidades de resolução de problemas (Akcaoglu, 2014; Akcaoglu e Koehler 2014).

Os dados recolhidos a partir dos projetos analisados e da revisão da literatura realizada permitem-nos concluir que os jogos educativos digitais, nomeadamente o software Kodu Game Lab, são altamente eficazes em crianças e jovens, os "nativos digitais" (Prensky, 2001).

Aprender a programar através de jogos digitais: o Kodu Game Lab

O ambiente de jogo Kodu Game Lab, apresenta-se como um programa que implementa uma linguagem de programação visual desenvolvida por Microsoft especificamente para o ensino da programação (MacLaurin, 2011).

Ao contrário do Scratch e Alice, o Kodu parece tornar a aprendizagem mais fácil para o utilizador, as imagens são mais atraentes e a criação jogos tridimensionais é inerente ao trabalho no programa. Stolee e Fristoe (2011), conduziram um estudo em 346 ambientes de jogo Kodu Game Lab com o objetivo de responder a três questões de pesquisa:

1) Que conceitos das Ciências da Computação podem ser ensinados ou expressos no Kodu Game Lab?

2) Quanto tempo podem os utilizadores passar a programar os seus jogos no Kodu Game Lab?

3) Com que frequência cada conceito das Ciências da Computação aparece nestes jogos?

Quanto ao ensino, os autores sustentam que o Kodu tem o potencial para ensinar uma vasta variedade de conceitos das Ciências da Computação incluindo variáveis, visão estratégica, lógica booleana, objetos, e fluxo de controlo. Enquanto algumas ações se repetem por predefinição, uma falha, percetível, existente no Kodu Game Lab, é a falta de suporte para definir estruturas de controlo básicas como ciclos e funções. Uma vez que não consegue desenvolver estes conceitos, poderia sustentar-se que o Kodu Game Lab omite a instrução fundamental que poderia gerar a compreensão baseada na resolução de problemas dos estudantes e como trabalham o programa. O

estudo, contudo, produziu alguns resultados positivos. Embora os autores admitam a necessidade de testes para confirmar que a aprendizagem ocorreu, muitas das experimentações foram realizadas em mundos grandes e complexos que expuseram os alunos a vários conceitos de programação. O estudo mostrou que quase três quartos dos jogos estudados continham personagens programados ou, por outras palavras, personagens que tinham código criado para torná-los mais interativos do que o seu estado original (com um número médio de regras por personagem de 5,14). Consequentemente, conclui-se que os utilizadores não só criam estes personagens, mas também demoram algum tempo para os tornar mais complexos. Um dos resultados interessantes deste estudo centrou-se no tempo passado no próprio ambiente de jogo Kodu. O estudo mostrou que, em média, os utilizadores passaram aproximadamente 25 minutos, por sessão, utilizando o Kodu. Durante esse tempo, os utilizadores editaram e codificaram numa média de 12,7 vezes e jogaram o seu jogo numa média de 13,7 vezes. Os autores acreditam que estes dados indicam que os utilizadores rapidamente passam da edição ao jogo para testar o seu novo código à medida que o desenvolvem. Esta é uma técnica comum dos estudantes que aprendem como programar (Hanks & Brandt, 2009).

Um estudo suportado pela Scientific and Technological Research Council of Turkey (TUBITAK) foi também implementado como uma atividade de verão intitulado de “Programming my own game”. 25 alunos dos 6º e 7º anos integraram a atividade. Durante a atividade o Kodu Game Lab foi ensinado aos alunos e posteriormente foram autónomos para criar o seu próprio jogo com o Kodu Game Lab. Utilizou-se um desenho de estudo de pesquisa de natureza qualitativo, mais especificamente, o método de estudo de caso.

Neste estudo específico o Kodu Game Lab foi escolhido como ambiente de programação porque permite aos utilizadores criarem jogos 3D semelhantes ao jogo que utilizam na sua vida e porque utiliza uma linguagem simplificada de programação (Fowler, 2012).

Durante as entrevistas, os alunos foram questionados acerca da melhor coisa na atividade de verão. 8 alunos afirmaram ter sido a criação do seu próprio jogo. Um dos alunos afirmou que entendeu e criou muito bem o jogo na parte final e que algo muito bom havia emergido, tendo ficado bastante satisfeito. 4 alunos exprimiram a sua felicidade sobre as atitudes dos instrutores em relação a eles e que esta sensação havia sido a melhor coisa no momento da atividade. 3 alunos falaram de atividades sociais, 1 aluno falou acerca da certificação e 1 aluno afirmou que a partilha de jogos foi a melhor coisa da atividade.

Os alunos também foram questionados acerca da pior coisa no momento da atividade. 6 alunos afirmaram que a observação tinha sido aquilo que poderia chamar-se de mau. No primeiro dia da atividade, as crianças estiveram presentes numa conferência sobre programação. 5 dos alunos mencionaram a conferência como a pior coisa da atividade. 2 alunos queixaram-se de permanecer demasiado tempo parados ao computador e aos almoços. Por fim, 1 aluno afirmou que estar atrás durante as sessões foi a pior coisa.

Para 5 alunos nada foi desafiante durante a atividade de verão enquanto 5 deles afirmaram que o início da atividade foi desafiante. Para 3 crianças que passaram ao nível seguinte e para 2 crianças que pontuaram no Kodu Game Lab foi um desafio. A programação, o término do jogo, as faltas do Kodu e levantar-se cedo foram outros desafios mencionados por 1 aluno.

Para investigar o efeito profissional da atividade de verão, os alunos foram questionados, no final da atividade, sobre se escolheriam uma carreira ligada à programação, como engenharia

informática. 11 dos alunos deixaram subentender que a programação pode ser uma opção de carreira no futuro e 10 estudantes afirmaram que não modificaram a sua decisão profissional.

Finalmente os alunos foram questionados acerca das alterações que gostariam de fazer na atividade. 14 alunos sugeriram que a atividade durasse pelo menos duas semanas. 4 alunos pretendiam modificar o tempo da atividade como atividade letiva. 3 dos alunos afirmaram que algumas características devem acrescentar-se ao Kodu Game Lab. Além disso foi ainda mencionado por um aluno, os almoços, microfones e auscultadores para cada participante, acrescentando algumas atividades desportivas.

Os resultados deste estudo mostram que a programação de jogo com Kodu Game Lab atrai o interesse dos alunos e que estes gostaram bastante de programar o seu próprio jogo. Baytak e Land (2010), Hwang (2013) e Robertson (2012) também realçaram o efeito motivacional da programação de jogo.

Os alunos podem usar Kodu para a programação de jogos e não tiveram dificuldades na utilização do programa quando se habituam a ele.

Este estudo mostrou-nos ainda que, no início da atividade, os estudantes devem apoiar-se mais.

Conclui-se também que atividades de programação com ambientes de jogo semelhantes devem ter a duração de, pelo menos, duas semanas.

Recorrendo ainda ao estudo elaborado pelo Department of Education and Early Childhood Development - Melbourne /Austrália que envolveu 25 turmas de 20 escolas diferentes, desenvolvido por Allan Fowler e Brian Cusack, com crianças entre os 10 e os 13 anos, este pretendia responder à questão: «os resultados da aprendizagem dos alunos melhoraram com a integração de tecnologias da Web 2.0? Se sim, em que medida, de que forma e em que circunstâncias?». Os professores inquiridos foram unânimes acerca da utilização pedagógica do

ambiente de jogo Kodu Game Lab. Segundo eles a utilização do Kodu Game Lab em sala de aula é relevante para a aprendizagem dos alunos e desenvolvimento do currículo, fornece um ambiente propício à aprendizagem colaborativa, promove a autonomia e auto motivação, permite a utilização de pensamentos de nível superior, a criatividade e a resolução de problemas, permite a comunicação e colaboração numa comunidade mais ampla que o contexto turma através do Wiki Planeta Kodu e promove o desenvolvimento de práticas de avaliação, incluindo a avaliação interpares.

Relativamente à opinião dos professores acerca do impacto que a utilização do Kodu teve na aprendizagem dos alunos, destacam-se a motivação e o envolvimento na aprendizagem, a criatividade, o desenvolvimento do pensamento crítico, a colaboração e a transferência de habilidades e conhecimentos para outros aspetos da aprendizagem.

Dos alunos inquiridos, em comparação com experiências anteriores, 61 % afirmaram que adquiriram mais conhecimento, 58% afirmaram que ganharam maior confiança na capacidade em realizar as tarefas e 54% dos alunos consideraram ter desenvolvido um pensamento mais profundo sobre as tarefas, foram mais criativos e tiveram consideração pela opinião dos outros. Além disso, mais de 50% dos alunos afirmou ter gostado mais de trabalhar com o Kodu do que noutros projetos em que esteve envolvido, assumiu maiores responsabilidades pela sua própria aprendizagem e colaborou mais com os outros.

Tendo em consideração os estudos referidos e a perspetiva de que o ambiente de jogo Kodu Game Lab, através da construção de jogos, permite a resolução de problemas, a programação num contexto visual e iconizado e permite aos alunos discutir e trabalhar colaborativamente conseguindo elaborar uma história, construir um jogo e apresentar o seu projeto; dado que o Kodu é um instrumento que ajuda a compreender a lógica da programação (sequências, ciclos, estrutura

de repetição, condições e ações, identificação e correção de erros), o Kodu torna a programação interessante e criativa para os alunos, que ao invés de ter que escrever códigos, utiliza e cria objetos. Cada personagem e objeto tem um conjunto específico de ações, comportamento e propriedade, reagindo a um conjunto de diferentes eventos (ver, bater, ouvir, mover, lançar, entre outros).

Metodologia

Neste capítulo são apresentados uma breve introdução sobre a investigação qualitativa e descritos o desenho de investigação, a caracterização dos participantes, instrumentos de recolha de dados e os procedimentos.

A natureza qualitativa desta investigação justifica uma primeira abordagem a esta temática. A investigação qualitativa tem na sua essência, segundo (Bogdan & Biklen, 1994), cinco características:

1) a fonte direta dos dados é o ambiente natural e o investigador é o principal agente na recolha desses mesmos dados;

2) os dados que o investigador recolhe são essencialmente de carácter descritivo;

3) os investigadores que utilizam metodologias qualitativas interessam-se mais pelo processo em si do que propriamente pelos resultados;

4) a análise dos dados é feita de forma indutiva;

5) o investigador interessa-se, acima de tudo, por tentar compreender o significado que os participantes atribuem às suas experiências.

A investigação qualitativa utiliza principalmente metodologias que possam criar dados descritivos que lhe permitirá observar o modo de pensar dos participantes numa investigação.

Segundo (Coutinho & Chaves, 2002), o facto de o investigador estar pessoalmente implicado na investigação confere aos planos qualitativos um forte cariz descritivo, daí que a grande maioria dos investigadores considere o estudo de caso como uma modalidade de plano qualitativo.

Segundo Gauthier (2003), a investigação científica visa conhecer melhor uma realidade, compreender melhor este universo do qual fazemos parte.

Desenho de investigação

O desenho de investigação adotado corresponde ao estudo de caso cuja unidade de análise é uma turma de alunos do 3.º e 4.º ano de escolaridade, inscritos na Oferta Complementar do respetivo Agrupamento de Escolas. A adoção do estudo de caso pode ser justificada pela singularidade da situação considerando que as iniciativas de aprendizagem da programação são ainda escassas e com poucas oportunidades de trabalho no âmbito dom currículo formal. Também o voluntarismo dos alunos deve ser sublinhado uma vez que a disponibilidade e o gosto por este tipo de atividades não está muito disseminado entre a população escolar.

O estudo de caso recorre a técnicas de observação – observação das aulas ou sessões em que decorreram as atividades, com registo vídeo – vinhetas – e com registo dos dados nas notas da investigadora. Também recorre à análise do código de programação criados pelos alunos durante a construção do seu projeto – um jogo no ambiente Kodu a partir de uma história, introduzindo uma abordagem narrativa à aprendizagem da programação para crianças no ambiente computacional Kodu.

Esta investigação teve como ponto de partida uma análise e reflexão de natureza profissional do contexto onde a investigadora desenvolve a sua atividade docente. Nesta fase, a análise realizada teve como finalidade caracterizar e avaliar alguns parâmeros relevantes para o conhecimento dos alunos bem como averiguar sobre algum possível problema ou necessidade. Nesta análise foram igualmente considerados os conteúdos do projeto Curricular de Agrupamento, o Projeto Educativo,

o Projeto Curricular de Turma, a legislação enquadradora, critérios de avaliação, as Metas Curriculares para o 1ºciclo e as linhas orientadoras da iniciativa "Programação e Robótica no Ensino Básico".

Desta reflexão resultou a perceção da professora e investigadora que a maioria dos alunos eram desatentos e que a maioria não demonstrava interesse pelas atividades letivas (facto registado no Projeto Curricular de Turma) e consequentemente, apresentava um comportamento agitado dentro da sala de aula. Tendo em conta o perfil dos alunos, concluímos que seria necessário adotar uma estratégia que os motivasse, os desafiasse e os envolvesse na conceção de algo significativo (Papert,1993).

O facto de existir, em Oferta Complementar para os 3º e 4º anos, a opção de Programação no 1ºCiclo, e procurando atribuir uma dimensão holística ao projeto, foi apresentada aos alunos a ideia de desenvolverem jogos baseados numa história, com o objetivo de desenvolver e articular várias competências e saberes. A maioria demonstrou agrado, apesar de alguns preferirem jogos com mais ação. Esta possibilidade não foi inviabilizada desde que se mantivessem dentro do mesmo tema.

Estavam, desta forma, reunidas as condições para implementar uma opção pedagógica intrinsecamente ligada a uma visão da educação numa perspetiva construcionista: uma abordagem em que os alunos aprendem através da participação em projetos, estabelecendo ligações entre diferentes ideias e áreas do conhecimento e o/a professor/a assume um papel de orientador das atividades de aprendizagem. Seria um processo de co construção (trabalho em colaboração com a Biblioteca Escolar; em colaboração entre colegas; e em parceria interdisciplinar com a disciplina de Português); e seriam produzidos artefactos (jogos) (Papert, 1993). Desta forma, e conciliando a informação recolhida com o disposto nas Metas Curriculares e nas linhas orientadoras da

iniciativa “Programação e Robótica no Ensino Básico”, a nossa intervenção centrou-se no subdomínio: exploração e criação de jogos no Kodu Game Lab.

Dos planos qualitativos estudados, o estudo de caso pareceu-nos o mais indicado para o desenvolvimento desta investigação pois corresponde ao estudo intensivo de um processo que envolve uma estratégia educativa inovadora, envolvendo um conjunto de alunos do ensino básico, um caso com interesse científico e pedagógico para a comunidade educativa, um caso intrínseco, ou nas palavras de Creswell, sendo que a investigadora procurou uma compreensão aprofundada (2005), das aprendizagens adquiridas pelos alunos quando imersos nestes ambientes de aprendizagem computacionais, recolhendo dados por múltiplas formas (Creswell, 2005)

Podemos afirmar que é a estratégia utilizada quando se pretende conhecer o “como?” e o “porquê?” (Yin, 1994) e quando o investigador detém escasso controlo dos acontecimentos reais ou mesmo quando é inexistente, e quando o campo de investigação se concentra num fenómeno natural dentro de um contexto da vida real.

Yin (2005) define o estudo de caso como estratégia de pesquisa que possui na sua essência esclarecer uma decisão ou um conjunto de decisões, assim como o motivo pelo qual foram tomadas, como foram implantadas e com quais resultados obtidos dentro de uma situação específica. Para Bell (2002) o método de estudo de caso é o mais adequado para investigadores isolados, visto que possibilita o estudo de determinado aspeto num espaço de tempo não muito alargado. Ainda, segundo este autor, num estudo de caso, o investigador observa, questiona e estuda, recolhendo dados que incluem documentos, observação direta, entrevistas, registos e artefactos físicos (Bell, 2002). A opção por um estudo de caso único, o qual enfatiza o conhecimento do particular (André, 1995), poderia colocar em causa a validade externa da investigação (Coutinho, 2002; Yin, 2003). Porém, face ao objeto de estudo definido, não se visava

a generalização dos resultados. No entanto, e como considera Yin (2003), a questão da fiabilidade relacionada com a replicabilidade das conclusões a que se chega (Vieira, 1999) não pode deixar de ser colocada se queremos que ao estudo de caso seja reconhecida pertinência e valor. Para isso exorta o investigador a fazer uma descrição tão pormenorizada quanto possível de todos os passos operacionais do estudo, e a conduzir a investigação como se alguém estivesse sempre a espreitar por cima do seu ombro (Yin, 2003), possibilitando que outros autores independentes possam repetir os mesmos procedimentos em contextos comparáveis (Vieira, 1999).

Caracterização dos Participantes

O projeto foi concebido e implementado com o intuito de compreender quais as competências cognitivas, sociais e atitudinais que os alunos desenvolvem através da criação de jogos no ambiente de jogo Kodu Game Lab, tendo como alvo os 20 alunos, do Agrupamento de Escolas de Borba (8-10 anos). O grupo/turma é constituído e caracterizado da seguinte forma:

- um aluno de etnia cigana matriculado pela terceira vez no 2º ano;
- um aluno de etnia cigana, matriculado no 3º ano, que beneficia das medidas universais de suporte à aprendizagem e à inclusão (D.L. nº54/2018), (aluno A);
- dezoito alunos matriculados no 4º ano, beneficiando quatro deles das medidas universais de suporte à aprendizagem e à inclusão (D.L. nº54/2018), (alunos B, C, D e E).

A turma, em geral, apresenta um aproveitamento que se situa entre o Suficiente e o Bom. Destacam-se dois alunos pelos bons resultados alcançados em todas as áreas curriculares.

Relativamente ao comportamento, o grupo/turma é em geral muito falador e desatento, sendo esse fator, de acordo com a professora titular de turma, um forte obstáculo ao seu sucesso escolar.

A professora titular de turma considera ainda que os alunos da turma, de uma forma geral, são pouco esforçados, desistindo facilmente das tarefas quando estas os obrigam a maior empenho e dedicação, recorrendo imediatamente à sua ajuda.

As metodologias e as estratégias utilizadas, tanto com a professora titular de turma como com a docente/investigadora foram as mais diversificadas possíveis, recorrendo a aulas práticas, trabalho de grupo e trabalho de projeto. O recurso ao trabalho a pares surgiu em primeira instância na sala de aula para que os alunos com mais dificuldades fossem apoiados por alunos com maior segurança nas aprendizagens.

Aos alunos com dificuldades que se distanciam muito do resto do grupo, foram-lhe aplicados, trabalho e apoio mais individualizados.

Os alunos B, C e D por apresentarem dificuldades face às competências gerais e específicas em Português, Matemática e Estudo do Meio, beneficiam das seguintes medidas de diferenciação pedagógica:

Apoio prestado pelo professor e/ou pelos pares em sala de aula;

- ✓ Trabalho em grupo na sala de aula;
- ✓ Solicitação mais frequente na sala de aula;
- ✓ Realização de tarefas faseadas e orientadas.

Estes alunos beneficiam das Acomodações Curriculares a aplicar em Conselho de Turma:

- ✓ Trabalhos de casa que envolvam a família;
- ✓ Ensino de métodos de estudo;
- ✓ Desenvolvimento de competências de comunicação/escrita;
- ✓ Apresentação de situações e materiais reais/concretos;
- ✓ Reforço positivo;

- ✓ Diversificação de estratégias;
- ✓ Beneficiar de um professor coadjuvante na sala de aula;

Os alunos B, C e D beneficiam de Adaptações ao Processo de Avaliação, nomeadamente:

Diversificação dos instrumentos de recolha de informação;

- ✓ Tempo suplementar para a realização da prova;
- ✓ A leitura de enunciados;
- ✓ Despenalização do erro;
- ✓ Valorização da oralidade;
- ✓ Teste de escolha múltipla/ respostas curtas;
- ✓ Testes adaptados/ nível de escola.

O aluno E por apresentar dificuldades face às competências gerais e específicas em Português, Matemática, Estudo do Meio e Inglês, beneficia das seguintes medidas de diferenciação pedagógica:

- ✓ Apoio prestado pelo professor e/ou pelos pares em sala de aula;
- ✓ Trabalho em grupo na sala de aula;
- ✓ Solicitação mais frequente na sala de aula;
- ✓ Realização de tarefas faseadas e orientadas.

Este aluno beneficia das seguintes Acomodações Curriculares a aplicar em Conselho de Turma:

- ✓ Trabalhos de casa que envolvam a família;
- ✓ Ensino de métodos de estudo;

- ✓ Desenvolvimento de competências de comunicação/escrita;
- ✓ Apresentação de situações e materiais reais/concretos;
- ✓ Reforço positivo;
- ✓ Diversificação de estratégias;
- ✓ Beneficiar de um professor coadjuvante na sala de aula;

O aluno E beneficia de Adaptações ao Processo de Avaliação, nomeadamente:

- ✓ Diversificação dos instrumentos de recolha de informação;
- ✓ Tempo suplementar para a realização da prova;
- ✓ A leitura de enunciados;
- ✓ Despenalização do erro;
- ✓ Valorização da oralidade;
- ✓ Teste de escolha múltipla/ respostas curtas;
- ✓ Testes adaptados/ nível de escola.

Há ainda a salientar que o aluno E não foi autorizado, pelo seu Encarregado de Educação, a participar no estudo e que o aluno matriculado no 2º ano de escolaridade também não integrou o estudo por não frequentar as aulas de Oferta Complementar (apenas dirigidas para os 3º e 4º anos de escolaridade).

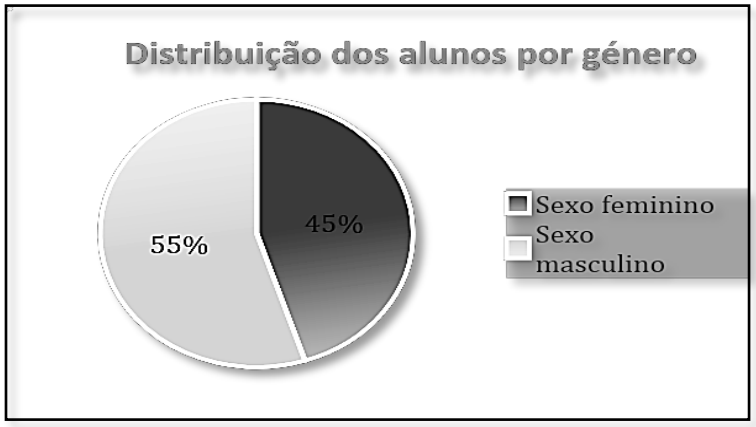
Contexto pessoal e escolar

O grupo/turma é constituído por 20 alunos matriculados em 3 anos de escolaridade distintos, distribuídos da seguinte forma: 1 aluno matriculado no 2ºano de escolaridade, 1 aluno matriculado no 3ºano de escolaridade e 18 alunos matriculados no 4ºano de escolaridade.

Figura 2 Distribuição dos alunos por ano de escolaridade

Quanto à distribuição dos alunos por género, 45% (9) são do sexo feminino e 55% (11) do sexo masculino.

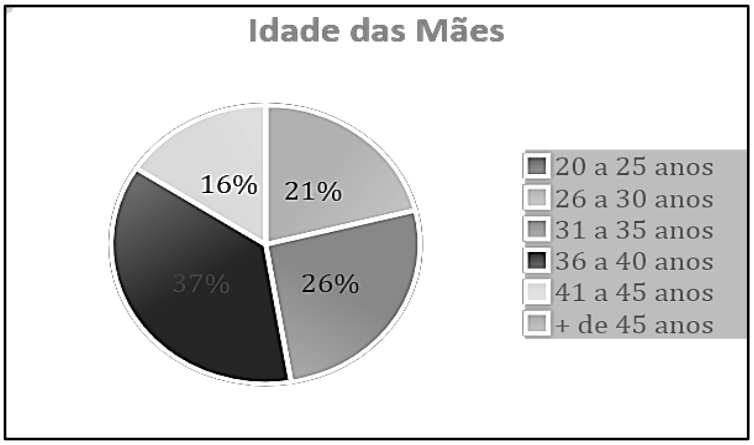
Figura 3 Distribuição de alunos por género



Contexto social e económico

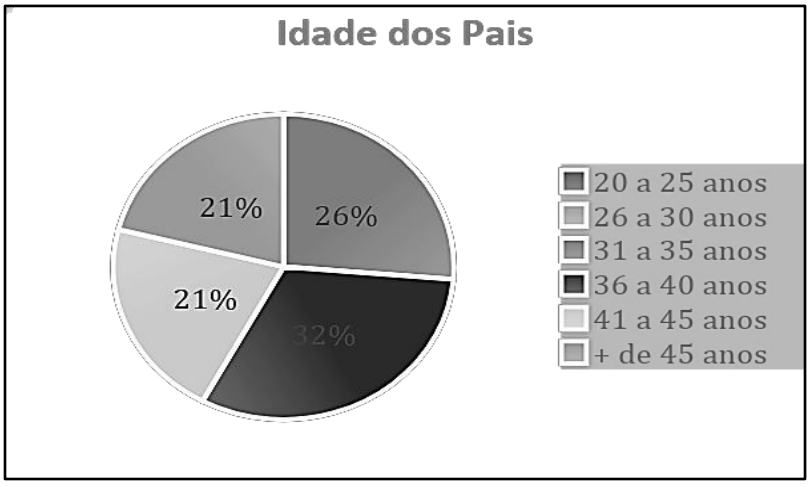
Relativamente à idade das mães, 37% (7) têm entre 36 e 40 anos, 26% (5) têm entre 31 e 35 anos, 21% (4) têm entre 26 e 30 anos e 16 % (3) têm entre 41 e 45 anos. (Não possuímos os dados relativos a uma das mães)

Figura 4 Idade das mães



No que diz respeito à idade dos pais, 32% (6) têm entre 36 e 40 anos, 26% (5) têm entre 31 e 35 anos, 21% (4) têm entre 41 e 45 anos e 21% (4) têm mais de 45 anos. (Não possuímos os dados relativos a um dos pais).

Figura 5 Idade dos pais



Procedimentos

Nesta seção descrevemos os procedimentos adotados ao longo de um determinado período de tempo (12 de novembro a 10 de dezembro de 2018, incluindo a estratégia pedagógica adotada e as atividades desenvolvidas com os alunos durante as sessões de trabalho educativo.

No que diz respeito à estratégia, considerámos uma *abordagem narrativa* destinada a promover o pensamento computacional em alunos do 1.º ciclo em que as atividades do projeto foram estruturadas e subdivididas em quatro fases principais: Storyline, Formulação de Problemas (Abstração), Formulação da Solução (Automação), Execução e Avaliação (Análise e Depuração).

No seguimento da descrição da abordagem narrativa, descrevemos agora os eventos conduzidos durante as sessões dedicadas à construção dos projetos de programação.

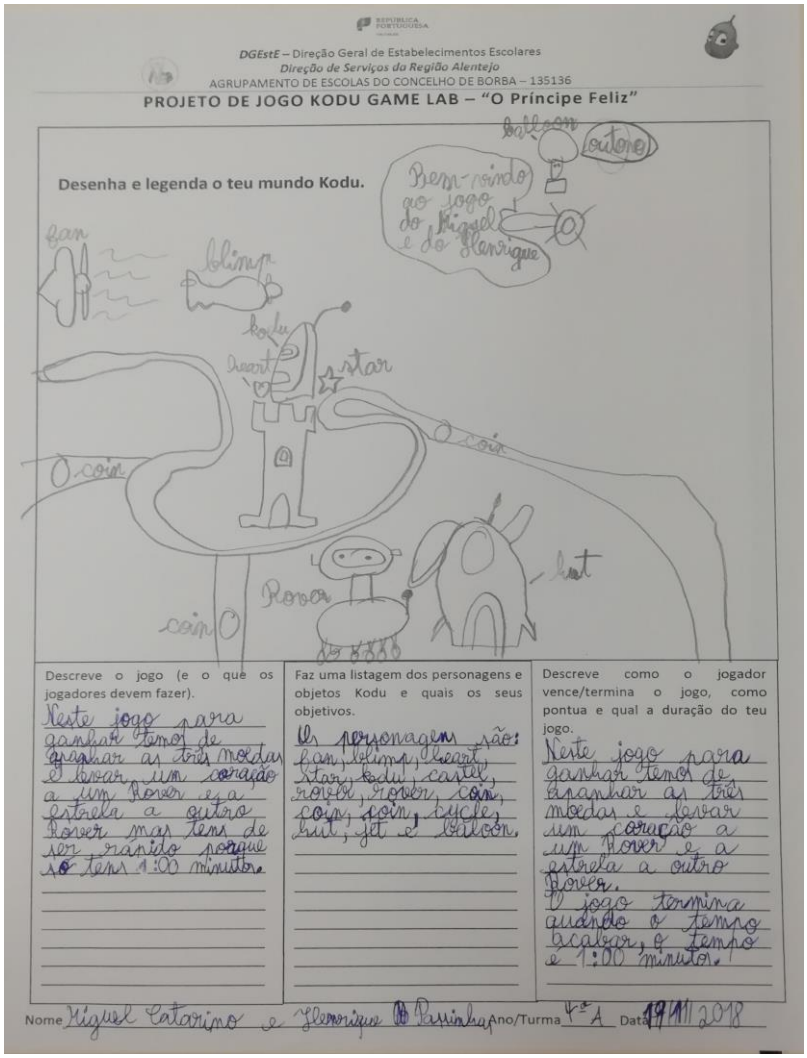
A primeira aula observada decorreu no dia 19 de novembro de 2018 na sala de informática do Agrupamento de Escolas de Borba, entre as 16:30h e as 17:30h.

Um problema foi apresentado aos alunos. Após a apresentação da história "O Príncipe Feliz", os alunos teriam que construir um jogo, baseado na história, no ambiente de jogo Kodu Game Lab. Os alunos foram divididos em grupos de trabalho o que deu origem a 8 grupos, 7 com 2 alunos cada e 1 com 3 alunos, privilegiando desta forma o Pair Programming (programação a pares) como estratégia de aprendizagem da programação ativa e colaborativa e que ajuda os alunos a ultrapassar frustrações quando estão perante problemas desafiadores, aumentando a autoconfiança e o interesse pela programação.

Foi entregue uma folha de projeto a cada grupo de trabalho na qual os alunos construíram os seus projetos de jogo, nomeadamente as correspondências entre os personagens da história e os

personagens do jogo, os objetivos de cada personagem, o desenho e descrição do jogo e a forma como venciam ou terminavam o jogo, como se pode observar na figura 6.

Figura 6 Projeto de jogo Kodu Game Lab – "O Príncipe Feliz"



DGEstE – Direção Geral de Estabelecimentos Escolares
Direção de Serviços da Região Alentejo
AGRUPAMENTO DE ESCOLAS DO CONCELHO DE BORBA – 135136

PROJETO DE JOGO KODU GAME LAB – "O Príncipe Feliz"

Desenha e legenda o teu mundo Kodu.

Descreve o jogo (e o que os jogadores devem fazer).

Neste jogo para ganhar tenot de apanhar as três moedas e levar um coração a um Rover e a estrela a outro Rover mas tens de ser rápido porque só tens 1:00 minutos.

Faz uma listagem dos personagens e objetos Kodu e quais os seus objetivos.

Os personagens são: Ban, blimp, Heart, star, Kodu, castle, robot, rover, coin, coin, coin, bycle, bat, jet e balloon.

Descreve como o jogador vence/termina o jogo, como pontua e qual a duração do teu jogo.

Neste jogo para ganhar tenot de apanhar as três moedas e levar um coração a um Rover e a estrela a outro Rover.
O jogo termina quando o tempo acabar, o tempo é 1:00 minutos.

Nome Miguel Catarino e Henrique Pereira Ano/Turma 4º A Data 19/11/2018

No que diz respeito à segunda aula os grupos de trabalho mantiveram-se em funcionamento na segunda aula observada que decorreu no dia 26 de novembro de 2018, com recurso à estratégia do Pair Programming.

Nesta aula os grupos dedicaram-se à construção dos seus jogos no ambiente de jogo Kodu Game Lab.

Seis dos oito grupos trabalharam nos computadores de secretária existentes na sala de informática e dois grupos trabalharam em computadores portáteis com webcam para que o trabalho e a interação fossem registados como forma de observação de aulas durante as sessões.

A investigadora solicitou aos alunos que criassem o seu próprio jogo, baseado no projeto construído na aula anterior, dizendo em voz alta tudo o que lhes viesse à mente.

As gravações resultantes encontram-se em formato MP4 e guardadas com data e hora específicas.

O Pair Programming revelou-se uma excelente estratégia de trabalho uma vez que permitiu aos pares a articulação do trabalho e no caso do Miguel e do Henrique permitiu também a partilha de conhecimentos como podemos constatar neste excerto de diálogo:

"Miguel: *Faz "Weak*".

Henrique: *O que é "Weak*"?

Miguel: *Fraquinho. Vamos meter um vento fraquinho. Já não dá mais...*

Henrique: *Ah, já está!*"

No caso do Gabriel, da Débora e do Guilherme o Pair Programming criou a possibilidade para que todos opinassem e partilhassem conhecimentos como podemos constatar neste excerto de diálogo:

"Débora: *Ah, mas a torre tem que ser maior. Senão é um Kodu gigante em cima de uma torre minúscula.*

Gabriel: *Agora temos é que virar.*

Débora: *Está aqui, rodar. Roda mais.*

Gabriel: *Vamos adicionar objeto.*

Débora: *Muda de cor primeiro*."

É de referir que, apesar de todos os alunos se terem mostrados participativos e interessados, o Pair Programming apresentou um constrangimento no que diz respeito ao "saber esperar". Os alunos apresentaram-se impacientes sem saber esperar pela sua vez de programar como podemos constatar neste excerto do diálogo entre o Miguel e o Henrique:

"Henrique: *Empresta aí. Desvia-o mais um pouco.*

Miguel: *Não.*

Henrique: *Miguel o terreno não é só teu.*

Miguel: *Eu sei, espera aí. Alterar altura.*

Henrique: *A seguir de alterares o tamanho, emprestas-me. Para pôr ao lado do Kodu. Alterar o valor. Vá, vá Miguel Catarino, vá Miguel."*

A tarefa/problema proposta aos alunos foi desenhada de acordo com as bases teóricas da resolução de problemas, isto é, foi apresentada aos alunos uma situação aberta que lhes exigiu uma atitude ativa para alcançarem uma solução (Echeverría, 1998). No desenho deste problema foi tida em consideração a utilização de um número limitado de conceitos e regras (estrutura) e de questões e variáveis (complexidade) (Jonassen, 2004). Foi também pensada para que os alunos adaptassem a sua compreensão do problema para chegarem a uma solução (dinamismo) (Jonassen, 2004), e utilizassem os conhecimentos previamente disponíveis (adquiridos nas aulas anteriores de Oferta Complementar) para lhe dar resposta (Echeverría, 1998). Para isso, eram obrigados a compreendê-

lo, a elaborar um plano, a executá-lo e a verificar os seus resultados (Pólya, 2003). Durante todo o processo de desenho e construção do jogo a investigadora esteve sempre presente e acompanhou os alunos em todas as atividades.

Consideramos que a estratégia desta aula funcionou na totalidade. Os alunos resolveram a tarefa proposta, procurando explorar o ambiente de jogo Kodu Game Lab de forma autónoma.

É de referir que alguns alunos conseguiram desenvolver jogos bastante criativos. Acrescentaram outros elementos no jogo, colocaram questões pertinentes e chegaram a desafiar os colegas a introduzirem mais funcionalidades nos seus ambientes de jogo, o que atribuiu um ritmo dinâmico à aula.

A terceira e última aula observada decorreu no dia 3 de dezembro de 2018. Para a preparação desta terceira aula tivemos em consideração os aspetos positivos e os aspetos menos positivos da última aula. Assim, através da reflexão individual da aula anterior procedemos a pequenas alterações estratégicas. O facto de termos constatado que os alunos apresentavam alguma dificuldade em "saber esperar", levou-nos a optar por concentrar o momento expositivo no início da aula, deixando o resto da aula para teste e correção dos erros identificados durante o jogo.

Os alunos concluíram a tarefa no tempo previsto tendo-se verificado um maior grau de atenção e motivação para o cumprimento dos objetivos da aula.

Apesar destas evidências, verificaram-se casos pontuais de alunos menos atentos e mais conversadores. Para prevenir esta situação procuramos reduzir o momento expositivo, sem comprometer os conteúdos, dando assim a possibilidade dos alunos aprenderem a aprender autonomamente e aprenderem uns com os outros (Papert, 1993).

Em resumo:

Os projetos/jogos criados pelos grupos de trabalho enumeram os personagens/objetos utilizados e programados e os construtos utilizados.

Foram construídos oito Projetos, um por cada grupo de trabalho, baseados na história "O Príncipe Feliz". Para a sua construção os grupos de trabalho projetaram e posteriormente criaram os seus jogos no ambiente de Jogo Kodu Game Lab.

O presente estudo inclui a exploração da utilização do ambiente de programação Kodu Game Lab (Fowler et al., 2012) como um ambiente para estimular a criatividade e diversão da criança na resolução de problemas, através da realização de jogos.

O estudo de caso que aqui desenvolvemos foi iniciado com uma pesquisa bibliográfica no intuito de identificar as habilidades e competências cognitivas, sociais e atitudinais que os alunos desenvolvem através de uma estratégia de criação de jogos no ambiente de jogo Kodu Game Lab. De seguida, pesquisamos modelos de avaliação de competências do pensamento computacional no contexto da programação visual, nomeadamente no ambiente de jogo Kodu Game Lab. Dos frameworks para avaliação do pensamento computacional, analisados, o de Brennan e Resnick (2012) apresentou-se como o mais abrangente para avaliação de competências do pensamento computacional em programação visual. Embora tenha sido concebido sob a perspetiva do ambiente Scratch, avaliamos sua aplicabilidade e limitações no contexto da programação visual, no ambiente Kodu.

Do ponto de vista da investigação, as atividades tiveram simultaneamente propósitos específicos de desenvolvimento e análise das competências cognitivas, sociais e atitudinais que os alunos desenvolvem através da criação de jogos no ambiente de jogo Kodu Game Lab e resultaram também do forte investimento de aprofundamento teórico do tema. Ainda, no âmbito da

investigação, havia a necessidade de fazer com que as atividades (e sua exploração) possibilitassem uma forma de recolha de dados compatível com a realização da investigação.

Na condução das aulas, a exploração das atividades foi pensada com este duplo propósito e permitiu à investigadora ter presente, em simultâneo, as questões de ensino e as questões da investigação. Este facto permitiu, naturalmente, uma abordagem na exploração das atividades, diferente se estes papéis fossem assumidos por diferentes pessoas, pois ao professor que aplica atividades criadas pelo investigador poderá não ser fácil perceber as questões de investigação subjacentes e, ao investigador poderá ser difícil gerir e interpretar as questões do ensino.

A experiência de ensino neste nível de escolaridade, neste Agrupamento, por parte da investigadora, permitiu adequar as propostas de atividades aos alunos em questão e antecipar as possíveis respostas e dificuldades que pudessem surgir. Neste sentido, e no que concerne à dimensão de conteúdo, a contribuição da programação para o desenvolvimento do pensamento computacional em alunos do 1º ciclo do ensino básico, exigia um conhecimento particular da professora, uma vez que se pretendia identificar aspetos do pensamento computacional de forma articulada com os conteúdos programáticos do 4º ano de escolaridade e com os projetos desenvolvidos pelos alunos.

Por exemplo, não bastava apenas trabalhar as questões relacionadas com os personagens e as ações da história apresentada, mas era necessário trabalhá-los de forma a promover a sua articulação com os personagens do ambiente de jogo Kodu Game Lab. Acrescem a essas atividades a complexidade de exploração de um tema não muito habitual no 1.º ciclo. Nesse sentido, o investimento feito enquanto investigadora através do aprofundamento teórico do tema permitiu que a condução das aulas fosse intencionalmente feita para promover o desenvolvimento do pensamento computacional e da promoção das mais variadas competências.

Relativamente à dimensão pedagógica que orientou a experiência de ensino, a natureza das aulas realizadas revestiu-se de uma exigência particular, como anteriormente referido. Neste sentido, a experiência da investigadora que, enquanto professora desenvolve, habitualmente, essa prática nas suas aulas, beneficiou o sucesso da experiência de ensino. Desta forma, a prática de ensino-aprendizagem exploratória foi assumida ao longo do estudo e incorporada, de forma gradual, como cultura de sala de aula, pela professora/investigadora e pelos alunos.

No que concerne aos alunos do 1º ciclo do ensino básico a literatura tem enfatizado a pertinência das estratégias e abordagens narrativas para promover processos de aprendizagem. (Szurmak & Mindy, 2013). Por este motivo, propõe-se uma abordagem para, estimulando o pensamento computacional, incluir um palco narrativo a nível preliminar. Uma história pode assim simbolizar o passo inicial no desenvolvimento de uma linha que suporta a ancoragem das ações e das personagens na criação dos eventos da narrativa que irão ser traduzidos em eventos computacionais. A concretização destes eventos no ambiente de programação reflete o desenvolvimento de alguns dos princípios e conceitos de pensamento computacional.

Esta tarefa estimula a criatividade de crianças e atua como processo de compreensão para a formulação de uma abstração (formulação do problema). A formulação do problema será a segunda etapa do processo que visa identificar elementos-chave do enredo: personagens, protagonistas, antagonistas, elementos de palco virtual, meta, ações de personagens e regras. A próxima etapa é a formulação da solução, que por sua vez, representa o primeiro passo para o desenvolvimento do jogo. Esta etapa envolve o processo de codificação, identificando os elementos-chave com instruções.

A fase final consiste em deixar os alunos jogar e testar (execução e avaliação) o seu jogo para identificar erros e avaliar os resultados da sua atividade.

Atendendo a que o estudo em causa pressupôs o desenvolvimento de um projeto da investigadora com os alunos em contexto extracurricular e, tendo em conta o funcionamento da escola e a legislação em vigor, foi necessário apresentar, no início de novembro de 2018, um pedido de autorização para a sua implementação (em Anexo). O projeto foi aprovado, contando de imediato com a recetividade e disponibilidade do Diretor, não implicando, contudo, a dispensa dos pedidos de autorização aos Encarregados de Educação, documento que pode ser consultado em Anexo.

As sessões de trabalho realizaram-se na Sala de Informática da Escola. A sala está equipada com 20 mesas para os alunos, cada uma com um computador, acessível a dois alunos. A sala possui ainda um quadro interativo, videoprojector e a mesa do professor está também munida com um computador.

O caminho da aprendizagem foi implementado num contexto onde os alunos estão envolvidos numa atividade de aprendizagem lúdica que estimula os princípios da programação (Chiazzese et al., 2015). Na verdade, os alunos foram envolvidos num processo colaborativo que visou conceber e desenvolver um jogo utilizando o software Kodu Game Lab.

Figura 7 Abordagem narrativa proposta para promover o pensamento computacional

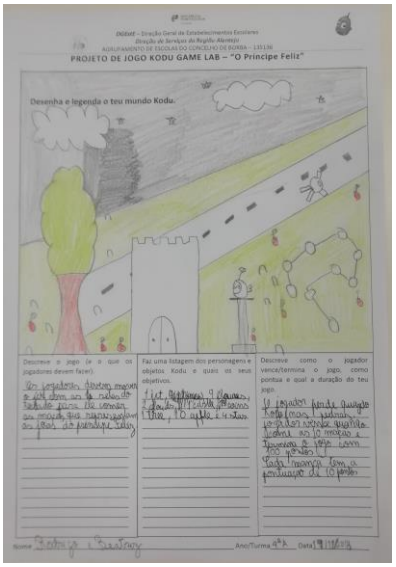
Etapa	Descrição	Modelo Pedagógico desenvolvido
Storyline (Narrativa)	A descrição de personagens e eventos de uma história.	
Formulação de Problemas (Abstração)	Formulação da história em termos de atores, protagonistas, antagonistas, elementos de palco virtuais, metas, regras e ações dos personagens.	
Formulação da Solução (Automação)	QUANDO algo acontece FAZ alguma coisa (paradigma de programação do ambiente virtual Kodu)	
Execução e avaliação (análise e depuração)	Testar o jogo construído utilizando a ferramenta PLAY do ambiente virtual Kodu.	

Figura 8 Esquema e duração das unidades de ensino

Unidade	Duração (Sessões)	Conteúdos/Objetivos	Tarefas/Atividades
Vamos falar sobre jogos	1 sessão (1 hora)	Introdução ao projeto; Jogar jogos de amostra; Revisão do processo e das etapas para o desenvolvimento de um jogo.	
Desenvolver o seu próprio jogo	2 sessões (2 horas)	Criação do jogo (projeto) em suporte de papel; - Criação do jogo baseado na história "O Príncipe Feliz", no ambiente de jogo Kodu Game Lab; ·	Desenho do cenário de jogo virtual em suporte de papel; Projeção dos personagens, cenário e ações; Desenho do cenário de jogo no ambiente de jogo Kodu Game Lab; Desenvolvimento do jogo, personagens e programação.
Avaliar	1 sessão (1 hora)	· Testa e corrige os erros de programação.	Teste e correção dos erros identificados.

Seis sessões foram realizadas na sala de informática da escola:

- 1 sessão (1hora): revisão da programação visual e do software Kodu. Nesta sessão, os alunos reviram os conceitos básicos de programação visual pela manipulação de telhas físicas do software Kodu (Touretzky, 2014).

- 2 sessões (2 horas): projeto de um cenário de jogo virtual - definição de objetivos, escolha das regras, construção de ações de jogos, criação de sistema de pontuação; codificação - desenvolvimento e construção dos jogos.

Os alunos foram guiados na construção autónoma de jogos simples a partir da descrição narrativa da história “O Príncipe Feliz”.

- 1 sessão de avaliação (1 hora): teste e correção dos erros identificados durante o jogo.

Revela-se importante para a clareza do estudo a utilização de múltiplas fontes de evidência.

Tal como é recomendado por Anderson (2002), Pardal e Correia (1995) e Yin (2003), foi realizada a recolha de dados através de várias fontes, tais como:

- observação das aulas, com registo escritos nas notas da investigadora;
- análise dos projetos de programação dos alunos com recurso ao Kodu.

Instrumentação

Nesta seção descrevemos com detalhe os instrumentos de recolha de dados utilizados durante a investigação.

Observação de Aulas

A observação das aulas foi realizada durante o processo de elaboração dos Projetos de programação em Kodu. Segundo Ludke e André (1986), a observação é um dos instrumentos básicos para a recolha de dados na investigação qualitativa. Na verdade, é uma técnica de recolha de dados, utilizando os sentidos, de forma a obter informação de determinados aspetos da realidade. Obriga o investigador a um contacto mais direto com a realidade, ajudando-o a identificar e a obter provas a respeito de objetivos sobre os quais os indivíduos não têm consciência, mas que orientam o seu comportamento (Lakatos & Marconi, 1990; Santos 1999, 2002).

Quer a observação seja estruturada ou não, o seu papel consiste em observar e registar da forma mais objetiva possível e depois interpretar os dados recolhidos. Como vantagens para esta técnica, podemos referir o facto de a observação permitir chegar mais perto da "perspetiva dos sujeitos" e a experiência direta ser melhor para verificar as ocorrências (Ludke & Andre, 1986), ou ainda o permitir a evidência de dados que não seriam possíveis de obter nas respostas a questionários (Lakatos & Marconi, 1990).

Tuckman (2000) refere que na investigação qualitativa, a observação visa examinar o ambiente através de um esquema geral para nos orientar e que o produto dessa observação é registado em notas de campo. A observação ajuda e muito na recolha dos dados, pois permite comparar aquilo que se diz ou não diz, com aquilo que se faz.

Para registar a observação no decorrer das aulas optámos pela utilização das grelhas de observação/avaliação (Brenann, 2014) traduzidas por Ramos (2015) e que nos permitiu captar com maior pormenor e profundidade as aprendizagens e os processos desenvolvidos pelos alunos no decorrer dos seus projetos. Apesar das grelhas terem sido, inicialmente, construídas para avaliação de dimensões do pensamento computacional através do Scratch, concluímos que se adaptam ao ambiente de jogo Kodu Game Lab.

O ambiente de programação do Kodu Game Lab pretende encorajar a auto-aprendizagem, convidando o utilizador a experimentar e recebendo feedback em simultâneo. Toda esta interação encontra-se facilitada, pois o Kodu possui uma interface do utilizador baseada na filosofia When-Do (Quando-Faz). Partindo deste pressuposto os alunos são "convidados" a experimentar e interagir, a testar e a corrigir os seus projetos/jogos.

No caso específico do trabalho a observação traduziu-se nos registos de dados nas notas da investigadora e na utilização das grelhas de observação/avaliação o que permitiu recolher

evidências práticas do desempenho dos alunos e das dimensões do pensamento computacional trabalhadas e desenvolvidas.

Projetos de programação dos alunos

Um segundo instrumento foram os projetos de jogos Kodu criados pelos alunos e os critérios de avaliação de desempenho dos alunos em ambientes computacionais. Os projetos de jogos em "linguagem Kodu" constituíram evidências de aprendizagem, dos conceitos de programação do ambiente Kodu Game Lab.

Análise dos dados

Os registos e os materiais obtidos durante o estudo foram objeto de análise. No que diz respeito às observações, que se materializaram num conjunto de registos escritos foram objeto de análise de conteúdo com recurso às dimensões previamente definidas no estudo e que correspondem aos constructos sobre os quais pretendemos investigar: dimensão cognitiva, dimensão social e dimensão atitudinal do pensamento computacional. Os segmentos dos registos escritos que manifestamente não se enquadravam nestas categorias, foram considerados numa categoria designada "não relevante".

Com função de complemento e adicionalmente foram ainda utilizadas algumas "vinhetas em vídeo" que correspondem a gravações das aulas. A gravação do conteúdo dos discursos de professora e alunos durante as aulas foi transcrito e podem ser consultadas nos Anexos. Os ruídos, conversas de interpretação inconclusiva e impasses (enquanto ninguém está a falar) bem como os assuntos que não considerámos pertinentes para o estudo, não foram transcritos.

Quanto à análise dos projetos criados pelos alunos através da linguagem de programação Kodu, foram objeto de classificação através da adoção dos critérios propostos por Pedro (2017). Adicionalmente e porque surgiu essa oportunidade estes projetos foram ainda analisados através de um programa de computador especificamente desenhado para esta função, o software Dr. Kodu.

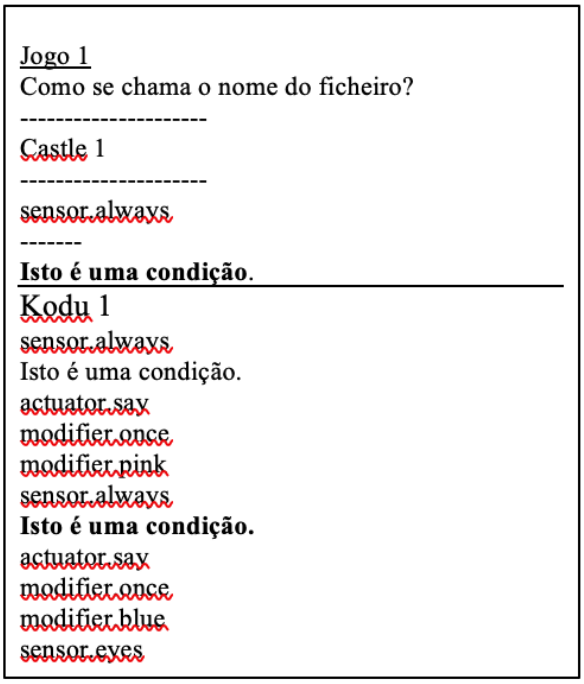
O software Dr. Kodu é uma ferramenta de análise do ambiente de jogo Kodu Game Lab desenvolvida por Miguel Perdigão Verdugo, à época aluno do 12º ano na Escola Secundária Rainha Santa Isabel de Estremoz.

Desenvolvido em 2019, o Dr. Kodu surgiu pela necessidade de um programa que analisasse os jogos construídos pelos alunos, em ambiente de jogo Kodu Game Lab, de forma rápida e eficaz.

O Dr. Kodu permite aos utilizadores analisar os jogos construídos no software Kodu, com o objetivo de visualizar com maior facilidade a programação que cada personagem tem.

Dr. Kodu é um aplicativo programado em Python que permite, através do ato de "Copy and Paste" do nome do ficheiro ".xml" (ficheiro gerado automaticamente pelo próprio software Kodu) mostrar todos os tipos de blocos de programação que cada identidade contenha nela, e desta forma revelar a lógica de programação que está por detrás da sintaxe construída pelos alunos bem como os construtos associados a cada tipo de programação (Figura 9).

Figura 9 Comandos do Programa Dr. Kodu



Ao analisar um jogo o Dr. Kodu apresenta ao utilizador todos os blocos que foram programados. Quando se inicia a função de "Teste" no "mundo", todos os personagens que têm algum tipo de programação, ou seja, que tenham algum desempenho que alterem o rumo do teste, são apresentadas com o Dr. Kodu e consequentemente toda a sua lógica e competências de pensamento computacional.

Resultados

Nesta seção apresentam-se os resultados obtidos durante a investigação com recurso à observação das aulas e à análise de dados obtidos através dos projetos de programação construídos pelas crianças e jovens participantes na investigação. Os resultados foram organizados a partir das

dimensões relativas aos padrões de desempenho dos alunos e às dimensões do pensamento computacional levando em consideração os objetivos da investigação.

Bogdan e Biklen (1994), descrevem os resultados da análise como «o trabalho com dados, a sua organização, divisão em unidades manipuláveis, síntese, procura de padrões, descoberta dos aspetos importantes e do que deve ser apreendido e a decisão sobre o que vai ser transmitido aos outros».

Os resultados decorrem da realização das atividades de Oferta Complementar desenvolvida com 17 alunos do 4ºano de escolaridade, turma A do Agrupamento de Escolas de Borba (8-10 anos) e os dados foram recolhidos a partir da Observação das Aulas, das Notas da Investigadora e da Análise dos Registos e Projetos dos Alunos. Os 17 alunos foram divididos em grupos de trabalho, privilegiando o "pair programming". Foram constituídos oito grupos de trabalho, sete com 2 alunos cada e um com 3 alunos.

Competências de aprendizagem da programação e padrões de desempenho

A partir da análise aos registos da observação e da correspondente análise de conteúdos, foram obtidos os seguintes resultados:

Figura 10 Competências associadas ao pensamento computacional observadas na sala de aula

Competências de aprendizagem	**Evidências da observação das aulas**
Cognitiva	1 - "Acrescentaram outros elementos no jogo, colocaram questões pertinentes…" 2 - "alguns alunos conseguiram desenvolver jogos bastante criativos." 3 - "Gabriel: Agora temos é que virar. Débora: Está aqui, rodar. Roda mais." 4 – "Gabriel: Vamos adicionar objeto." 5 – "Miguel: Eu sei, espera aí. Alterar altura." [abstração, modelação]
Social	1 - "No caso do Gabriel, da Débora e do Guilherme o Pair Programming criou a possibilidade para que todos opinassem…" [colaboração]

	2 - “permitiu aos pares a articulação do trabalho e no caso do Miguel e do Henrique permitiu também a partilha de conhecimentos” [colaboração]
Atitudinal	1 - “Os alunos apresentaram-se impacientes sem saber esperar pela sua vez de programar…” 2 - “permitiu aos pares a articulação do trabalho e no caso do Miguel e do Henrique permitiu também a partilha de conhecimentos”

Da análise aos registos da observação e às apreciações realizadas ao trabalho dos alunos relativos às atividades de programação com recurso ao ambiente computacional Kodu, e em particular as categorias de Experimentação e Interação, foram obtidos os seguintes resultados:

Figura 11 Padrões de desempenho relativos à Experimentação e Interação

Padrões de desempenho	**Baixo**	**Médio**	**Alto/Elevado**
Experimentar e interagir			
Descreve a construção do Projeto, passo a passo.	-	6 grupos	2 grupos
Enumera exemplos que experimentou ao longo da elaboração do Projeto.	4 grupos	2 grupos	2 grupos
Apresenta as revisões que foram sendo feitas.	4 grupos	3 grupos	1 grupo
Descreve as diferentes abordagens que experimentou no Projeto, ou quando tentou fazer algo novo.	6 grupos	-	2 grupos

Da análise e apreciação dos dados da observação das aulas e da apreciação aos programas construídos pelos alunos com recurso ao Kodu e relativos aos Testes e Correções, registados em Grelhas de Observação (ver Anexos), foram obtidos os seguintes resultados:

Figura 12 Níveis de desempenho relativos aos Testes e Correções

Padrões de Desempenho	**Baixo**	**Médio**	**Alto/Elevado**
Testar e Corrigir			
Descreve o que aconteceu com o projeto de diferente em relação ao pretendido.	-	3 grupos	5 grupos
Descreve de que forma foi feita a leitura do código para encontrar a causa do problema	-	5 grupos	3 grupos
Descreve que alterações fez e como as testou para verificar os resultados.	2 grupos	3 grupos	3 grupos
Descreve outras formas de resolver o problema.	2 grupos	3 grupos	3 grupos
Descreve se encontrou inspiração em outros projetos e na leitura do código disponível	-	6 grupos	2 grupos
Descreve como selecionou uma parte de outro projeto e como a adaptou ao seu.	3 grupos	3 grupos	2 grupos
Refere/cita as pessoas cujo trabalho inspirou o seu.	8 grupos	-	-

Pensamento computacional

Da análise dos dados decorrentes da observação das aulas e da apreciação dos projetos de programação e em particular nas dimensões relativas ao pensamento computacional, foram obtidos os seguintes resultados:

Figura 13 Análise das Dimensões do Pensamento Computacional

Dimensões do pensamento computacional	**Baixo**	**Médio**	**Alto - Elevado**
Descrição da construção do jogo.	-	-	8 grupos Forneceram detalhes sobre as diferentes componentes do projeto e descreveram o modo como foram desenvolvidos, de forma ordenada.
Experimentação ao longo da elaboração do projeto.	4 grupos Não apresentaram exemplos específicos do que experimentaram.	2 grupos Deixaram transparecer de forma genérica que experimentaram outras coisas no projeto.	2 grupos Forneceram exemplos específicos de outras coisas que experimentaram no projeto.
Revisão	4 grupos Afirmaram não ter feito revisões, ou terem feito algumas sem exemplificar.	3 grupos Descreveram uma revisão específica que fizeram ao projeto.	1 grupo Descreveram aspetos específicos de coisas que acrescentaram ao projeto e justificaram.
Descrição das diferentes abordagens experimentadas no projeto.	6 grupos Não revelaram evidências de ter experimentado algo novo.	-	2 grupos Descreveram com detalhe coisas novas que experimentaram no projeto.
Descrição do que aconteceu com o projeto de diferente em relação ao pretendido.		3 grupos Descreveram o que correu mal no projeto, mas não o que pretendiam fazer.	5 grupos Exemplificaram detalhadamente o que aconteceu e o que pretendiam, quando executaram o programa.
Descrição da forma como realizaram a leitura do código para encontrar a causa do problema.		5 grupos Descreveram como realizaram a leitura mas não apresentaram um exemplo específico de encontrar um problema no código.	3 grupos Descreveram como realizaram a leitura e apresentaram um exemplo específico de encontrar um problema no código.

Descrição das alterações e testes realizados para verificar os resultados.	2 grupos Não descreveram que problemas tiveram ou a solução.	3 grupos Forneceram um exemplo genérico sobre as alterações feitas e os testes feitos para verificar o funcionamento.	3 grupos Forneceram um exemplo específico sobre as alterações feitas e os testes feitos para verificar o funcionamento.
Descrição de outras formas de resolver o problema.	2 grupos Não apresentaram uma forma para encontrar uma solução para o problema.	3 grupos Apresentaram uma forma genérica para encontrar uma solução para o problema.	3 grupos Apresentaram um exemplo específico de como encontrar uma solução para o problema.
Descrição de fontes de inspiração em outros projetos e na leitura do código disponível.		6 grupos Forneceram uma descrição geral de um projeto que o inspirou.	2 grupos Forneceram um exemplo específico do projeto que os inspirou.
Descrição da forma como selecionaram uma parte de outro projeto e como a adaptaram ao seu.	3 grupos Não descreveram como adaptaram as ideias, scripts ou recursos de outros projetos.	3 grupos Identificaram scripts, ideias ou recursos que adaptaram de outros projetos.	2 grupos Forneceram exemplos específicos de scripts, ideias ou recursos que adaptaram de outros projetos e como.
Refere/cita as pessoas cujo trabalho inspirou o seu.	8 grupos Não identificaram as fontes e os autores em que se inspiraram para a realização do seu projeto.	-	-

Inspirado no projeto inicial de “Iniciação à Programação no Ensino Básico” que pretendia promover o desenvolvimento de um conjunto de competências e capacidades nos alunos, nomeadamente: o trabalho em equipa, a estruturação e organização de ideias, a criatividade, o espírito crítico, a resolução de problemas, entre outros, foi solicitado a cada grupo de trabalho que desenhasse o seu jogo, que o descrevesse, que enumerasse os personagens e que descrevesse como termina.

Através da análise dos projetos e da observação durante a resolução, foi possível identificar as três dimensões do pensamento computacional: conceitos computacionais; práticas computacionais; e perspetivas computacionais.

Estiveram sobretudo em evidência conceitos como condições e, em alguns casos, (alunos criativos) lógica booleana e lógica booleana indentada. Também as práticas computacionais (observadas durante o apoio prestado aos alunos) estiveram presentes no desenvolvimento da tarefa na medida em que os alunos foram desenvolvendo, testando e desenvolvendo um pouco mais (ação interativa e incremental), e foram verificando se tudo funcionava (teste e depuração). As perspetivas computacionais foram evidentes aquando da criação do programa (expressão), pois os alunos interagiram e criaram em conjunto, uma vez que estavam agrupados em pares (conexão), e questionaram as funcionalidades da ferramenta (questionar).

Recorde-se que as práticas computacionais são a segunda grande dimensão relatada por Brenann e Resnick (2012) e foram evidenciadas a partir de observações realizadas nos projetos desenvolvidos pelos alunos. “Práticas computacionais estão concentradas nos processos de pensar e aprender, focando não somente naquilo que se está a aprender, mas em como se está a aprender” (Brennan e Resnick, 2012).

Figura 14 Análise dos projetos/jogos em ambiente de jogo Kodu Game Lab

	Personagens/ objetos utilizados	**Personagens/ objetos programados**	**Pontuação**	**Construtos**
Jogo 1	20	12	Sim	Condição, Lógica Booleana, Lógica Booleana Indentada
Jogo 2	19	2	Não	Condição
Jogo 3	11	3	Sim	Condição, Lógica Booleana, Lógica Booleana Indentada
Jogo 4	35	3	Não	Condição
Jogo 5	16	1	Sim	Condição, Lógica Booleana, Lógica Booleana Indentada
Jogo 6	29	3	Sim	Condição, Lógica Booleana, Lógica Booleana Indentada
Jogo 7	24	3	Não	Condição
Jogo 8	11	2	Sim	Condição, Lógica Booleana, Lógica Booleana Indentada

Seiter e Foreman (2013) propõem um modelo de Progression of Early Computational Thinking Model (PECM). Trata-se de um modelo de avaliação do pensamento computacional a partir dos conceitos — procedimentos e algoritmos — decomposição de problemas — paralelização e sincronização — abstração e representação de dados. Para chegar a estes conceitos os autores mapeiam os padrões do desenho que o programador utiliza ao programar: animação da apresentação, animação por movimento, conversas entre personagens, colisões, pontuação de jogos, interação com o programador. Estes padrões de desenho são obtidos a partir das evidências

de comandos utilizados na linguagem Kodu, nomeadamente, som, movimento, variáveis, lógica booleana, condicionais e simulação.

Partindo da análise da Figura 16 compreendemos que o construto Condição está presente em todos os jogos programados no ambiente de jogo Kodu Game Lab já que o próprio conceito Quando/Faz é uma Condição.

O ambiente de jogo Kodu Game permite também que os seus utilizadores expressem a Lógica Booleana, a partir dos jogos criados, sempre que na programação de um personagem ou objeto é utilizada a negação. Cinco dos grupos de trabalho utilizaram a Lógica Booleana na programação de personagens e objetos.

Foram também esses cinco grupos de trabalho que utilizaram a Lógica Booleana Indentada. O Kodu Game Lab permite aos alunos indentar linhas de código, o que cria uma dependência lógica onde as regras do código indentado são avaliadas apenas quando a condição da regra "mãe" é cumprida.

Para este estudo a gravação das aulas proporcionou-nos a observação do ponto de vista dos alunos em relação à utilidade, jogabilidade e interação entre pares na utilização do ambiente de jogo Kodu Game Lab.

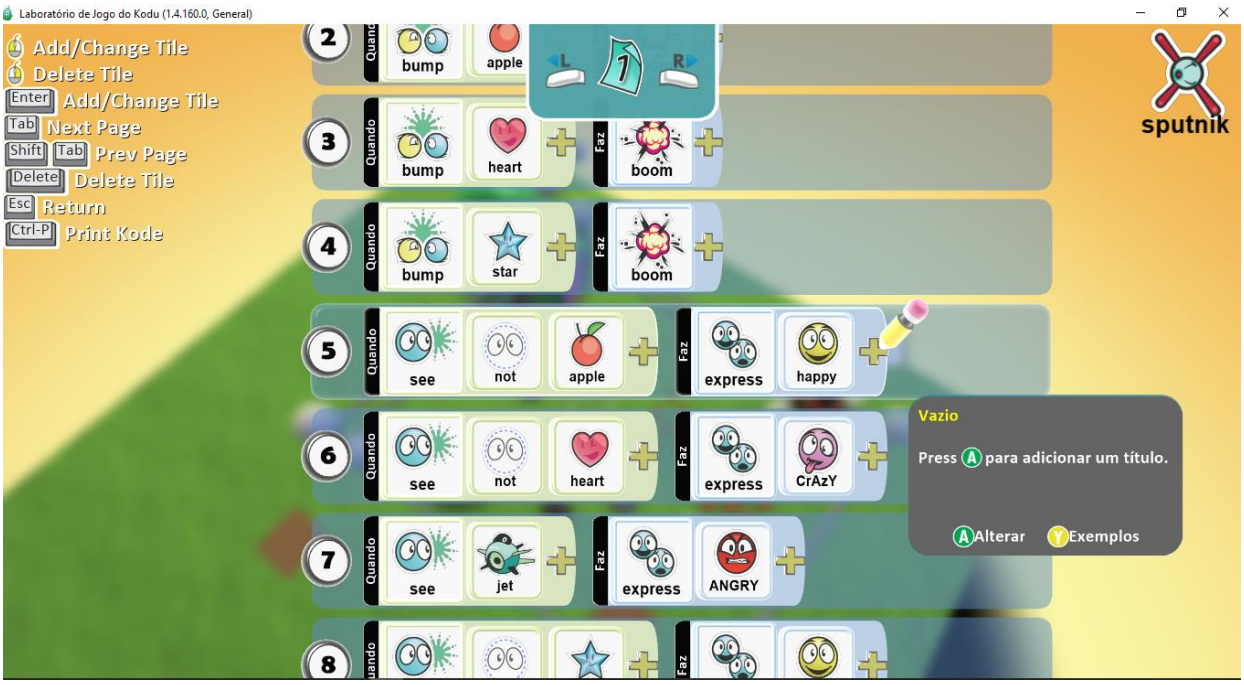
De acordo com o estudo realizado pelo Department of Education and Early Childhood Development − Melbourne /Austrália (2010), referido na fundamentação teórica deste trabalho, o Kodu oferece um suporte educacional ao desenvolvimento do pensamento crítico e de habilidades na resolução de problemas. Desenvolve também a colaboração, habilidades sociais e cognitivas favorecendo o desenvolvimento pedagógico e o envolvimento dos alunos nas atividades propostas pelo professor, que, nesse caso, passará a atua como um mediador e parceiro nas experiências interativas.

Através da gravação das aulas, tivemos a noção de que o Pair Programming facilita a partilha de conhecimento. Nos 8 grupos de trabalho ambos os programadores (alunos) estiveram fortemente envolvidos nas atividades de programação e trabalharam de forma colaborativa na resolução dos problemas, “Miguel: *Agora metemos outro Rover. Faz tu, faz. Eu deixo-te um bocadinho*.” (Ver Anexo)

A gravação das aulas também nos permitiu ter a perceção que os grupos de trabalho reveem a codificação utilizada e debatem entre si quais as melhores soluções para solucionar uma situação concreta. “Miguel: *Faz "Weak*". Henrique: *O que é "Weak*"? Miguel: *Fraquinho. Vamos meter um vento fraquinho. Já não dá mais…* Henrique: *Ah, já está*!” (Ver Anexo).

Perante o exposto e após a análise dos dados recolhidos temos a perceção de que os alunos envolvidos neste estudo de caso podem entender e efetivamente utilizar conceitos de programação básicos, quando projetam os seus próprios jogos digitais, criando e programando jogos que são funcionais e jogáveis. Os conceitos de programação mais comumente utilizados pelos alunos foram Condições e Lógica Booleana, como pode ser observado nos blocos de programação da figura. O paradigma simples da linguagem “Quando/Faz” é uma condição de partida para a definição de cada uma das ações a ser realizadas pelas personagens e ou objetos programados.

Figura 15 Parte de programação de jogo criado por aluno



O estudo evidenciou que todos os jogos criados pelos alunos continham personagens programados, isto é, personagens que tinham código criado para torná-los mais interativos do que o seu estado original. Consideramos, então que os alunos para além de utilizarem a imaginação para criarem os personagens, também demoram algum tempo para os tornar mais complexos (Figura 16).

Figura 16 Evidências das Dimensões do Pensamento Computacional

Pensamento computacional	Evidências
Dimensões cognitivas	• Conceitos de programação: ✓ Condições e Lógica Booleana • Criatividade
Dimensões sociais	✓ Trabalho colaborativo – objetivos partilhados na criação do jogo; negociação de soluções, compreensão de diferentes perspetivas na resolução de problemas; ✓ Interação ✓ Expressar opiniões
Dimensões atitudinais	✓ Partilha de conhecimento ✓ Saber esperar

Adicionalmente, e tal como referido por Lewis (2011) observámos que as explicações entre pares resultam melhor para a compreensão dos estudantes em comparação com outros recursos e as tarefas cognitivas são melhor sucedidas quando trabalhadas em pares (Kutnick et al, 2005)

Os alunos demonstraram ainda criatividade na elaboração de estratégias de resolução de problemas e no *Game design*.

Partindo da simulação de uma situação problema que exige soluções práticas e imediatas, o que estimula o planeamento das ações, os alunos construíram os seus jogos, criaram os seus personagens e deram-lhes vida através da programação.

Todos os alunos participantes no estudo demonstraram capacidade de identificação e correção de erros.

Tendo como base as orientações da revisão da literatura e uma análise preliminar aos dados recolhidos os resultados demonstram que os alunos desenvolveram determinadas competências sociais tais como a interação e a partilha; desenvolveram também a capacidade de resolver problemas, de criar, de executar e melhoraram as capacidades de teste e correção.

Conclusões e reflexões finais

Existe um movimento crescente de educadores e entidades representativas das ciências da computação que aceitaram a proposta de Jeannette Wing (2006) com o intuito de promover o Pensamento Computacional desde os primeiros anos de escolaridade. Acreditamos que essa promoção pode ser facilitada pela exploração de ambientes computacionais de apoio à aprendizagem que estimulem práticas e atitudes inerentes ao pensamento computacional, tais como: formular problemas de forma que permita usar um computador e outras ferramentas para solucioná-los; organizar e analisar dados de forma lógica; representar dados através de abstrações, modelos e simulações; automatizar soluções através de pensamento algorítmico; identificar, analisar e implementar soluções possíveis com o objetivo de alcançar a melhor e mais eficiente combinação de passos e recursos; generalizar e transferir esse processo de resolução de problemas a uma grande variedade de um mesmo tipo de problema (CSTA, 2012).

Com o intuito de contribuir para a o aprofundamento do conhecimento no que diz respeito ao pensamento computacional em contexto educativo apresentamos neste trabalho os resultados alcançados através de um estudo de caso desenvolvido com um grupo de alunos do Ensino Básico no Agrupamento de Escolas de Borba. O projeto procurou investigar como as dimensões cognitivas, sociais e atitudinais do pensamento computacional, podem ser trabalhadas através de atividades que envolvem a resolução de desafios de lógica, usando a linguagem de programação Kodu Game Lab, com recurso a instrumentos de recolha de dados apropriados à investigação nestes contextos: observação de aulas e análise dos projetos/jogos.

A experimentação foi realizada através da adoção de uma estratégia que deu lugar a uma suportada numa abordagem narrativa no ambiente de jogo Kodu Game Lab.

Considerando o referencial teórico adotado neste estudo, consideramos pertinente inscrever o software de jogo Kodu Game Lab num leque de ferramentas com potencial cognitivo. Apesar de reconhecermos o seu potencial cognitivo, este software assumiu um carácter de utilidade pedagógica nas disciplinas de Língua Portuguesa, na adaptação de uma história, personagens e ações, de Matemática, nas noções de área, volume, altura e lugar, e de outras áreas curriculares, quando são definidos objetivos e orientações metodológicas no processo de intervenção pedagógica com esta ferramenta.

No que respeita à articulação do software Kodu Game Lab com o currículo do 1º ciclo do ensino básico, inferiu-se que, num ambiente caracterizado pela informalidade, foi possível desenvolver o currículo, particularmente, alguns dos conteúdos pretendidos, nomeadamente o Storyline.

O *Storyline* permitiu aos alunos a apresentação e identificação do problema, o desenvolvimento do problema/situação (abstração), prever o resultado do problema (automação) e solucionar o problema (análise e depuração).

O Storyline mostrou-se útil tanto para a organização quanto para o controlo do projeto. Permitiu a criação independente da programação e também ajudou na metacognição, quando o aluno avalia o seu próprio trabalho ajustando o projeto às restrições, facilidades, prazos e competências próprias.

Os problemas que os alunos tiveram que resolver na concretização dos seus projetos em Pair Programming, e o domínio que desenvolveram na linguagem de programação com o software de programação Kodu Game Lab confirmam, que é possível lançar aos alunos desafios que permitam

uma aprendizagem curricular e promover as potencialidades de desenvolvimento cognitivo e social destas ferramentas.

O pensamento computacional aparece com naturalidade em projetos de desenvolvimento no software de programação Kodu Game Lab e este estudo mostrou competências relevantes e comuns com o pensamento computacional, inclusive da valorização do trabalho colaborativo e da preocupação com a autonomia e criatividade como fatores socialmente importantes. A abstração, articulação lógica e resolução de problemas, são competências comuns ao pensamento computacional e ao Kodu Game Lab. A consciência do pensamento computacional e a competência de programar computadores podem ser motivadores para ensinar as mais diversas áreas programáticas utilizando ferramentas como o Kodu Game Lab. A interação social é desejada e trabalha a favor do ambiente de aprendizagem. Os alunos encontram apoio e auxílio nos seus colegas, incentivando a realização dos seus programas. Os jogos e animações criaram uma empatia cultural entre estes alunos identificados como nativos digitais.

Em suma a análise efetuada nesta dissertação demonstrou que as características do ambiente de jogo Kodu Game Lab estão alinhadas com os três campos dos saberes que os alunos precisam dominar para o desenho de interfaces para o pensamento computacional: teorias da aprendizagem, Interação Homem-Computador e Ciências da Computação. A ferramenta permite a aprendizagem de práticas, conceitos e perspetivas do pensamento computacional, está alinhada com a teoria construcionista da aprendizagem e adota uma linguagem de programação visual para estimular a participação de crianças a partir de 8 anos de idade.

Relativamente às limitações dos métodos de recolha de dados, a natureza do estudo de caso realizado não permitiu extrapolações para outros contextos devido às limitações dos métodos de recolha de dados. A observação e o registo por parte da observadora que assume os papéis de

professora e investigadora, em simultâneo não é de todo fácil, pois dentro da sala de aula é muito difícil gerir estas duas facetas. Primeiro, porque não se adequa parar no meio de uma observação ou de um diálogo com os interlocutores da realidade que se observa para tomar notas, ou então gravar tudo o que vai ocorrendo: é necessária alguma seletividade naquilo que se pretendeu registar. Segundo, porque os critérios de seleção dos eventos a observar muitas das vezes não são tão óbvios quanto se deseja, tendo a investigadora dificuldade em avaliar a adequação dos critérios usados. Apontamos, assim, o problema da seletividade/cobertura limitada, já referenciada por Kenrick. (1999)

Esperamos que estes valores e conceções possam influenciar outros professores e também os meus alunos já que este trabalho se trata de uma maneira de ―" como fazer" – que desperta possibilidades de alterações e adaptações.

Referências Bibliográficas

Aitken, L., Marshall, A., Elliot, R., McKinley, S. (2011). Comparison of “think aloud” and observation as data collection methods in the study of decision making regarding sedation in intensive care patients. International Journal of Nursing Studies, 48, 318-325.

Akcaoglu, M. (2014). Learning problem-solving through making games at the game design and learning summer program. Educational Technology Research and Development. 62. 10.1007/s11423-014-9347-4.

Anderson, G. (2002). Case study. Em G. Anderson, Fundamentals of educational research (2ª ed., pp. 152-160). London: Routledge Falme

André, M. E. (1995). Etnografia da prática escolar (4ª ed.). Campinas: Papiros

Association (CSTA). (2011). Operational Definition of Computational Thinking for K–12 Education. Obtido de Operational Definition of Computational Thinking for K–12 Education. (CSTA). Retrieved from http://www.iste.org/docs/ct-documents/computational-thinking-operationaldefinition-flyer.pdf?sfvrsn=2

Bardin, L. (2011). Análise de conteúdo. São Paulo: Edições 70

Barr, V., Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? ACM Inroads, 2 (1), 48–54.doi: 10.1145/1929887.1929905.

Baytak, A., Land, S. M. (2010). A case study of educational game design by kids and for kids. Procedia - Social and Behavioral Sciences, 2(Innovation and Creativity in Education), 5242–5246.

Bell, J. (2002). Como Realizar um Projecto de Investigação. Lisboa: Gradiva

Bogdan, R., Biklen, S., (1994). Investigação Qualitativa em Educação – uma introdução à teoria e aos métodos. Porto: Porto Editora.

Bonnardel, N., Zenasni, F. (2010). The Impact of Technology on Creativity in Design: An Enhancement? Creativity and Innovation Management. 10.1111/j.1467-8691.2010.00560.x.

Brennan, K., Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. Paper presented at annual American Educational Research Association meeting, Vancouver, Canada, BC.

Britain, G. (2013). Children and parents: Media use and attitudes report. Ofcom. Retrieved from https://www.ofcom.org.uk/__data/assets/pdf_file/0020/108182/children-parents-media-use-attitudes-2017.pdf

Buckingham, D., Burn, A. (2007). Game literacy in theory and practice. Journal of Educational Multimedia and Hypermedia, 16(3), 323.

C.T.C.W. Mettes & A. Pilot, (1980) Over het leren oplossen van natuurwetenschappelijke problemen. Een methode voor ontwikkeling en evaluatie van onderwijs, toegepast op een kursus Thermodynamica (On Learning to Solve Physics' Problems. A Method for Designing and Evaluating Instruction, Applied to a Thermodynamics Course), Enschede: Technische Hogeschool Twente.

Cavalcante, A., (2016). Pensamento computacional e programação introdutória: um estudo de caso sobre competências desenvolvidas na programação em blocos com o code.org., Rio Tinto.

Charters, E. (2003). The use of think-aloud methods in qualitative research: An introduction to think-aloud methods. Brock Education, 12 (2), 68-82.

Chiazzese, G., Fulantelli, G., Pipitone, V., Taibi, D. (2015). Promoting computational thinking and creativeness in primary school children. 1-7. 10.1145/3144826.3145354.

Costa, S., (2016), Desenho de interface para o desenvolvimento do pensamento computacional no Ensino Básico: análise do Scratch, Retrieved from http://hdl.handle.net/10362/19935

Coutinho, C., Chaves, J. (2002). O estudo de caso na investigação em tecnologia educativa em Portugal. Revista Portuguesa de Educação.15 (1). 221-243

Creswell, J. W. (2005). Educational research: Planning, conducting, and evaluating quantitative. Upper Saddle River, NJ: Prentice Hall.

CSTA (2011). Computational Thinking in K–12 Education leadership toolkit. Retrieved from http://csta.acm.org/Curriculum/sub/CurrFiles/471.11CTLeadershiptToolkit-SPvF.pdf

CSTA. (2011). K–12 Computer Science Standards. Retrieved from https://csta.acm.org/Curriculum/sub/K12Standards.html

De Vos, A.S.; Strydom, H.; Fouché, C.B. and Delport, C.S.L. (2011). Research at Grass Roots: For the Social Sciences and Human Service Professions: 4 th edition. Pretoria: Van Schaik Publishers.

Denner, J., Werner, L. (2007). Computer Programming in Middle School: How Pairs Respond to Challenges. Journal of Educational Computing Research - J EDUC COMPUT RES. 37. 131-150. 10.2190/12T6-41L2-6765-G3T2.

Denner, J., Werner, L., Ortiz, E. (2012). Computer games created by middle school girls: can they be used to measure understanding of computer science concepts? *Computers & Education, 58*(1), 240–249.

Denning, P. J. (2009). The profession of IT: Beyond computational thinking. Communications of the ACM, 52 (6), 28–30. doi: 10.1145/1516046.1516054. doi: 10.1145/1118178.1118215

Echeverría, M. P. P.; Pozo, J. I. (1998). Aprender a resolver problemas e resolver problemas para aprender. In: POZO, J. I. (Org.). A solução de problemas: aprender a resolver, resolver para aprender. Porto Alegre: ArtMed, 13-42

Ericsson, K. A., & Simon, H. A. (1993). Protocol analysis: Verbal reports as data. Cambridge, MA: MIT Press

Fowler, A. (2012). Enriching student learning programming through using Kodu. Proceedings of the 25th Annual Conference of the National Advisory Committee on Computing Qualifications, 33–39

Fowler, A., Cusack, B. (2011). Kodu Game Lab: Improving the motivation for learning programming concepts. Proceedings of the 6th International Conference on the Foundations of Digital Games, FDG 2011. 10.1145/2159365.2159398

Fullerton, T. (2008). Game design workshop. Boston, MA: Elsevier

Garneli, V., Giannakos, M., Chorianopoulos, K., (2015) Computing education in K-12 schools: A review of the literature, IEEE Global Engineering Education Conference (EDUCON), Estonia

Gauthier. B. (2003). Investigação social: Da problemática à colheita de dados (3ª ed.). Loures: Lusociência.

Gee, J. (2003). What Video Games Have to Teach Us about Learning and Literacy. New York, Palgrave Macmillan.

Grimes, S. Fields, DA. (2015). Children's media making, but not sharing: The potential and limitations of child-specific, Australia, Media International.

Groot, A.D. (1946). Het denken van den schaker (Thought and Choice in Chess), Amsterdam: University of Amsterdam

Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. Educational Researcher, 42 (1), 38–43. doi: 10.3102/0013189X12463051

Hanks, B., Brandt, M. (2009). Successful and unsuccessful problem solving approaches of novice programmers. In Symposium on Computer science education, pages 24–28, New York, NY, USA, ACM

Hwang, G.-J., Hung, C.-M., & Chen, N.-S. (2013). Improving learning achievements, motivations and problem-solving skills through a peer assessment-based game development approach. *Educational Technology Research and Development*. doi:10.1007/s11423-013-9320-7

International Society for Technology in Education (ISTE) and the Computer Science Teachers

Jonassen, D.H. (2004). Learning to Solve Problem. An instructional Design Guide. San Francisco: John Wiley & Son, Inc.

Kafai, Y. (1995). Minds in play: Computer game design as a context for children's learning. Mahwah, EUA: Lawrence Erlbaum.

Kafai, Y. (2006). Playing and Making Games for Learning. Games and Culture. 1. 36-40. 10.1177/1555412005281767.

Kafai, Y. (2015). Connected code: A new agenda for K-12 programming in classrooms, clubs, and communities. Paper presented at the Math + Coding Symposium, Western University, London, Ontario, Canada

Kafai, Y. B., Burke, Q. (2014). Connected code: Why children need to learn programming. Cambridge, MA: The MIT Press

Kafai, Y. B., Burke, Q. (2015). Constructionist gaming: Understanding the benefits of making games for learning. Educational Psychologist, 50(4), 313–334

Kafai, Y., Harel, I. (1991). Learning through design and teaching: Exploring social and collaborative aspects of constructionism. In I. Harel, & S. Papert (Eds.) Constructionism (pp. 85-106). Norwood, USA: Ablex

Kafai, Y., Resnick, M. (1996). Constructionism in practice: Designing, thinking, and learning in a digital world. Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

Ke, F. (2014). An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing. Computers & Education, 73, 26–39

Kenrick, D. T., Neuberg, S. L., Cialdini, R. B. (1999). Social Psychology: Unravelling the mystery. Needham Heights, MA: Allyn & Bacon.

Kutnick, P., Sebba, J., Blatchford, P., Galton, M., Thorp, J., MacIntyre, H., & Berdondini, L. (2005). The effects of pupil grouping: Literature review. Research report RR688. Nottingham: DFES Publications

Lakatos, E. M., Marconi, M. A. (1990). Fundamentos de Metodologia Científica. São Paulo: Editora Atlas S.A.

Lewis, C. M. (2011). Is pair programming more effective than other forms of collaboration for young students? Computer Science Education, 21(2), 105-134. https://doi.org/10.1080/08993408.2011.579805

Ludke, M., André, M. E. D. A. (1986). Pesquisa em Educação. Abordagens Qualitativas. São Paulo: E. P. U.

MacLaurin, M. B. (2011). The design of kodu: A tiny visual programming language for children on the xbox 360. ACM Sigplan Notices, 46(1) 241-246

Margolis, J., Estrella, R., Goode, J., Holme, J., Nao, K. (2008). Stuck in the Shallow End: Education, race and computing. Cambridge MA: MIT Press

Martins, R., Reis, R., Marques, A. (2016). Inserção da programação no ensino fundamental, Uma análise do jogo Labirinto Clássico da Code.org através de um modelo de avaliação de jogos educacionais, Retrieved from http://dx.doi.org/10.5753/cbie.wie.2016.121

Muntjewerff, A.J. (2001). An Instructional Environment for Learning to Solve Legal Cases: PROSA (Ph.D. thesis University of Amsterdam), Amsterdam: University of Amsterdam.

National Research Council. (1999). Being fluent with information technology. Washington, DC: National Academy Press. Retrieved from http://www.nap.edu/download.php?record_id=6482 on November 11, 2018.

National Research Council. (2010). Report of a workshop on the scope and nature of computational thinking. Washington, DC: National Academy Press. Retrieved from http://www.nap.edu/openbook.php?record_id=12840

Palhares, P., (1997). Histórias com problemas construídas por futuros professores de Matemática. In Fernandes, D., Lester, F., Borralho, A. & Vale, I. (Coords.). Resolução de problemas na formação inicial de professores de Matemática – múltiplos contextos e perspectivas. Aveiro: GIRP/JNICT. p.p. 154-188.

Papert, S. (1995). Preface: Games to be played – Games to be made. In Y. Kafai, Minds in play: Computer game design for children's learning. Hillsdale, NJ: Laurence Erlbaum Associates (pp. ix-xii).

Papert, S., (1980) Mindstorms: children, computers, and powerful ideas, New York, Basic Books, Inc., Publishers

Papert, S., (1985) Computer Criticism vs. Technocentric Thinking. Logo 85 Theoretical Papers, MIT

Papert, S., (1991). Situating constructionism. In I. Harel & S. Papert (Eds.), Constructionism (pp. 1–11). Norwood: Ablex

Papert, S., (1994). A Máquina das Crianças: Repensando a Escola na Era da Informática. Porto Alegre: Artes Médicas

Pardal, L.; Correia, E. (1995). Métodos e técnicas de investigação social. Porto: Areal

Pedro, A., Matos, J. F., Piedade, J., & Dorotea, N. (2017). Probótica – Programação e Robótica no Ensino Básico (Linhas Orientadoras). *Instituto de educação da Universidade de Lisboa*.

Peppler K., Kafai Y. B. (2007). What video game-making can teach us about learning and literacy: Alternative pathways into participatory culture. In: Baba A., editor. *Situated play: Proceedings of the Third International Conference of the Digital Games Research Association (DiGRA)* Tokyo, Japan: The University of Tokyo; pp. 369–376.

Pólya, G. (2003). Como resolver problemas (Tradução do original inglês de 1945). Lisboa: Gradiva.

Polya, G., (1945). How to solve it; A new aspect of mathematical method. Princeton: Princeton University Press.

Prensky, M. (2001)., Aprendizagem baseada em jogos digitais. São Paulo: Senac.

Quivy, R., Campenhoudt, L.V. (1992). Manual de Investigação em Ciências Sociais. Lisboa: Gradiva

Ramos, J. L., (2016). Desafios da introdução ao pensamento computacional e à programação no 1º ciclo do Ensino Básico: racionalizar, valorizar e atualizar. Retrieved from

https://dspace.uevora.pt/rdpc/bitstream/10174/20201/1/Conselho%20Nacional%20de%20Educa%c3%a7%c3%a3o_2017.pdf

Ramos, J. L., Espadeiro, R. G. (2015). Pensamento computacional na escola e práticas de avaliação das aprendizagens. Uma revisão sistemática da literatura. Retrieved from http://dspace.uevora.pt/rdpc/bitstream/10174/14227/1/challenges%202015br.pdf

Robertson, J. (2012). Making games in the classroom: Benefits and gender concerns. Computers and Education, 59(2), 385–398.

Santos, M. C. (1999). Trabalho Experimental na aprendizagem em Ciências. O Desenvolvimento de Competências Científicas na disciplina de Técnicas Laboratoriais de Biologia. Lisboa: Universidade Nova de Lisboa, Faculdade de Ciências e Tecnologia

Santos, M. C. (2002). Trabalho Experimental no Ensino das Ciências. Lisboa: Ministério da Educação, Instituto de Inovação Educacional

Seiter, L., Foreman B. (2013). Modeling the learning progressions of computational thinking of primary grade students. ICER '13 Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research

Shute, V., Chen, S., Asbell-Clarke, J., (2017) Demystifying computational thinking, Educational, USA, Research Review 22, 142e158

Someren, M., Barnard, Y., Sandberg, J. (1994). The think aloud method. A practical guide to modeling cognitive processes. London, Academic Press

Stewart, D. W., Shamdasani, P. N.; Rook, D. W. (2007). Focus groups: Theory and practice (2nd ed.). Thousand Oaks, California: Sage

Stolee, K. T., & Fristoe, T. (2011). Expressing computer science concepts through Kodu game lab. In Proceedings of the 42nd ACM technical symposium on Computer science education. ACM

Szurmak, J.; Mindy, T., (2013). Tell me a story: The use of narrative as a tool for instruction In Imagine, Innovate, Inspire: The Proceedings of the ACRL 2013 Conference, ACRL, Indianapolis, USA, IN

Teixeira, J. (2017). Contribuições para o ensino de programação de computadores a futuros professores de matemática. Tese de doutoramento. Universidade do Minho. Retrieved from https://repositorium.sdum.uminho.pt/bitstream/1822/48711/1/Jaylson%20Teixeira.pdf.

The Royal Society. (2012). Shut down or restart? The way forward for computing in UK schools. London: The Royal Society.

Touretzky, D. (2014). Teaching Kodu with physical manipulatives. ACM Inroads. 5. 44-51. 10.1145/2684721.2684732

Transactions of the Royal Society of London A. (1881). Mathematical, Physical and Engineering Sciences, 366, 3717–3725. doi: 10.1098/rsta.2008.0118

Tuckman, B. (2000). Manual de Investigação em Educação. Lisboa: Fundação Calouste Gulbenkian

Ventura, A., (2017) O Scratch Promotor do Pensamento Computacional no Processo de Ensino - Aprendizagem da Geometria no 1.º CEB, Dissertação de Mestrado, Retrieved from http://hdl.handle.net/20.500.11796/2487

Vieira, C. (1999). A credibilidade da investigação científica de natureza qualitativa: questões relativas à sua fidelidade e credibilidade. Revista Portuguesa de Pedagogia

Vygotsky, L. S. (1978). Mind in society: The development of higher psychological processes. Massachusetts: Harvard University Press

Vygotsky, L. S. (1980). Mind in society: The development of higher psychological processes. Harvard University Press

Weinert, F. E. (1999). Definition and selection of competencies: Concepts of competence. Munich: Max Planck Institute for Psychological Research

Werner, L., Denner, J., Bliesner, M., Rex, P. (2009). Can middle-schoolers use Storytelling Alice to make games? Results of a pilot study. FDG 2009 - 4th International Conference on the Foundations of Digital Games, Proceedings. 207-214. 10.1145/1536513.1536552.

Wing, J. (2006). Computational thinking. Communications of the ACM, 49 (3), 33–36

Wing, J. (2008). Computational thinking and thinking about computing. Philosophical

Wing, J., (2014) "Computational Thinking Benefits Society". Social Issues in Computing. New York: Academic Press. Retrieved from Socialissues.cs.toronto.edu.

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding a 21st century problem solving in K-12 classrooms. TechTrends.doi: 10.1007/s11528-016-0087-7.

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. ACM Transactions on Computing Education, 14 (1), 1–16. doi: 10.1145/2576872.

Yin, R., (2003). Case study research: design and methods (3ª ed). Thousand Oaks, CA: SAGE Publications

Yin, R., (2005) Estudo de caso: planejamento e métodos (3. ed). Porto Alegre: Bookman

Apêndices

Apêndice I - Diário da Investigadora (Pro.mp4)

26/11/2018

Henrique: Vamos lá, novo mundo.

Miguel: Vamos ter que aumentar o terreno.

Henrique: Fazemos isto à vez está bem? Eu sou profissional. Tu és melhor que eu em Fortnight, eu sou melhor do que tu em Kodu Game Lab. Pára lá de brincar com isso.

Miguel: Não estou a brincar, estou a tentar meter isto tudo no ecrã.

Henrique: Emprestas isso? Passa cá o rato. Passa o rato. Teimoso, passa o rato que eu faço isso.

Miguel: Vá avança lá para o fundo. Agora metes o Kodu aí, mas tens que reduzir o tamanho dele.

Vou alterar a altura.

Miguel: Faz "Weak".

Henrique: O que é "Weak"?

Miguel: Fraquinho. Vamos meter um vento fraquinho. Já não dá mais...

Henrique: Ah, já está!

Miguel: Mete lá. E é mesmo fixe. Mesmo fixe, não se vê, hehehehehe. Não, mete o invisível. Mete lá Esc. Agora deixa-me fazer. Mete a rodar 360. Mete no máximo.

Henrique: Isto está mal. Vamos salvar: Nome, O Príncipe Feliz.

03/12/2018

Henrique: Adiciona lá aí mais terreno. Espera, empresta lá aí o rato.

Miguel: Vamos ter que mover o coração lá para cima.

Henrique: Olha lá adiciona lá aí mais terreno.

Miguel: Espera, falta o tamanho do heart. Pronto, já está 1 e 8.

Henrique: Fizeste um castelo com escudo.

Miguel: Como assim?

Henrique: Sim, isso é um castelo com escudo.

Miguel: Ah yah, com um coração.

Henrique: Não, como tens dois objetos juntos é tipo um exército, uma fortaleza.

Miguel: Então espera, deixa-me tentar outra vez. Alterar tamanho…

Henrique: Não.

Miguel: Eu sei fazer. Eu agora tenho que me por aqui.

Henrique: Vá agora põe a altitude.

Miguel: É isso que eu ia coiso. Alterar tamanho, altura.

Henrique: Está bom. Está bom. Daqui a nada queres que ele voe, não?

Miguel: Não, espera.

Henrique: Ele é para ficar mais para trás. Empresta lá aí o ratinho.

Miguel: Não agora vou meter aqui a Star.

Henrique: Empresta aí. Desvia-o mais um pouco.

Miguel: Não.

Henrique: Miguel o terreno não é só teu.

Miguel: Eu sei, espera aí. Alterar altura.

Henrique: A seguir de alterares o tamanho, emprestas-me. Para por ao lado do Kodu. Alterar o valor. Vá, vá Miguel Catarino, vá Miguel.

Miguel: Deixa lá ver, 2 Rovers. Deixa só meter um Rover e tu metes outro.

Henrique: Uau, vá, um Rover mais outro Rover.

Miguel: Rodar 229. Meter o Rover assim.

Henrique: Assim está bom. Fazemos tipo uma banca, vá, vá, empresta lá aí o rato. Olha, olha, vai lá ao Rover e muda ele para roxo. Não, vai com o rato.

Miguel: Eu sei.

Henrique: Não cliques, agora as setas. Vá. Uau, é muito giro.

Miguel: Agora metemos outro Rover. Faz tu, faz. Eu deixo-te um bocadinho.

Henrique: Primeiro.

Miguel: Não, não aumentes tanto.

Henrique: Isso está muito pouco. Vá mais um pouquinho, mais um pouquinho. Assim chega. Até aqui, até aqui. Assim vá. Vamos mover esta ventoinha para aqui.

Miguel: Olha lá, mete invisível, não metas fantasma. Mete-o invisível. Tu é que sabes. Meteu o Blimp que é a andorinha. A ventoinha, precisamos dela por causa do Blimp. Boa, deixa estar assim. Agora aumenta o tamanho do Blim. Aumenta o tamanho do Blimp.

<u>26/11/2018</u>

Gabriel: Isto não está a dar. Temos que pôr uma folha debaixo do rato.

Guilherme: Eu tenho ali. Eu vou buscar.

Gabriel: Eu vou começar já assim. Diminuímos ou aumentamos o mundo?

Guilherme: Temos que adicionar mundo.

Débora: O que é que estás a fazer?

Gabriel: É para termos espaço para tudo.

Guilherme: Não, vira para aquele lado.

Gabriel: Onde é que estão os castelos?

Débora: Assim, assim, assim sim.

Gabriel: Agora precisamos de um Kodu.

Débora: Ah mas a torre tem que ser maior. Senão é um Kodu gigante em cima de uma torre minúscula.

Gabriel: Agora temos é que virar.

Débora: Está aqui, rodar. Roda mais.

Gabriel: Vamos adicionar objeto.

Débora: Muda de cor primeiro.

Gabriel: Calma meu, temos tempo para tudo.

Débora: É esse.

Gabriel: Olha aqui ele a voar, yeh.

Débora: Não, tem que ser maior. Carrega lá.

Gabriel: Chega.

Débora: Tem que estar mais alto. Vai lá a alterar altura.

Gabriel: Ele está bem deste tamanho. Hum que fofinho.

Débora: Vais meter primeiro as maçãs? Então e as casas? Então e a casa? Ah, fica aqui à frente? Vira-a.

Gabriel: Ah não, não vamos tentar meter 20 moedas.

Débora: Vira o Kodu… Isso…. Agora a casa…. Olha aqui a casa. Vira-a. Isso. Vira-a toda para a frente. Aqui tem a porta para a frente.

Gabriel: Deixa-me, fica bonito assim.

Débora: Agora é as moedas.

Gabriel: Calma.

Débora: Vá, vamos guardar. Escreve aí, “O Príncipe Feliz”, Gabriel, Débora e Guilherme. Pincete?

Gabriel: Em vez de estares a cantar, tinhas-me ajudado. Epá vocês só me estão a baralhar, sabem que eu tenho dislexia. Metem-me a escrever isto com dislexia.

Guilherme: Eu sei fazer isso. Metes agora aqui e metes o “n”.

Débora: Apaga o “e”

Gabriel: Cala-te.

Débora: É príncipe feliz, agora deixas 3 espaços e escreves Gabriel, Débora e Guilherme.

Apêndice II - Projeto de jogo Kodu Game Lab

Desenha e legenda o teu mundo Kodu.

Descreve o jogo (e o que os jogadores devem fazer).

_____
_____
_____
_____
_____
_____
_____
_____

Faz uma listagem dos personagens e objetos Kodu e quais os seus objetivos.

_____
_____
_____
_____
_____
_____
_____
_____

Descreve como o jogador vence/termina o jogo, como pontua e qual a duração do teu jogo.

_____
_____
_____
_____
_____
_____
_____

Nome _____ Ano/Turma_____ Data _____

Anexos

Anexo I - Consentimento Informado, Esclarecido e Livre para Autorização de Participação em Estudos de Investigação

CONSENTIMENTO INFORMADO, ESCLARECIDO E LIVRE PARA AUTORIZAÇÃO DE PARTICIPAÇÃO EM ESTUDOS DE INVESTIGAÇÃO

(de acordo com a Declaração de Helsínquia e a Convenção de Oviedo)

Título do estudo: Contributos da Programação para o desenvolvimento do pensamento computacional em alunos do 1.º ciclo do Ensino Básico. Um estudo de caso no Agrupamento de Escolas de Borba.

Enquadramento: Estudo de Caso no âmbito da tese de Mestrado em Administração, Regulação e Políticas Educativas pela Universidade de Évora.

Investigadora – Elsa Severo Rôlo

Orientador - Professor Doutor José Luís Ramos.

Explicação do estudo: Estudo implementado de 12 de novembro a 10 de dezembro de 2018 nas aulas de Oferta Complementar com a turma A do 4ºAno do Agrupamento de Escolas de Borba.

Instrumentos de recolha e pesquisa de dados:

- Observação de aulas;

- Análise dos registos e projetos dos alunos;

- Gravação dos relatos dos alunos, por forma a facilitar a recolha dos dados. No final de cada sessão, as gravações devem ser reproduzidas em escrito de modo fidedigno, permitindo a obtenção de dados válidos e confiáveis (Someren,et al., 1994).

Condições e financiamento: A participação neste estudo de caso é de caráter voluntário e não acarreta quaisquer custos ou prejuízos, assistenciais ou outros, caso não autorize a participação do/a seu/sua educando/a.

Confidencialidade e anonimato: Será garantida a confidencialidade dos dados recolhidos os quais serão para uso exclusivo do presente estudo.

A identificação dos/as alunos/as participantes será anónima (não existindo registo de dados de identificação).

Todos os contactos serão feitos em ambiente de sala de aula, garantido normalidade e privacidade aos/às alunos/as envolvidos/as no estudo de caso.

A Investigadora – Elsa de Fátima Velez Severo Rôlo

Docente de 1ºciclo do Ensino Básico

Agrupamento de Escolas de Borba.

Contacto telefónico -966811865

Endereço eletrónico – elsarolo@gmail.com

Por favor, leia com atenção a seguinte informação. Se achar que algo está incorreto ou que não está claro, não hesite em solicitar mais informações. Se concorda com a proposta que lhe foi feita, queira assinar este documento.

A Investigadora _____

Declaro ter lido e compreendido este documento, bem como as informações que me foram fornecidas pela Investigadora.

Desta forma, autorizo a participação do/a meu/minha educando/a neste estudo e permito a utilização dos seus dados que de forma voluntária forneço, confiando em que apenas serão utilizados para esta investigação e nas garantias de confidencialidade e anonimato que me são dadas pela investigadora.

Nome do/a aluno/a: _____

Assinatura do/a Encarregado/a de Educação _____

ESTE DOCUMENTO É COMPOSTO POR 2 PÁGINAS E FEITO EM DUPLICADO: UMA VIA PARA A INVESTIGADORA, OUTRA PARA O/A ENCARREGADO DE EDUCAÇÃO

Anexo II - Requerimento para implementação de Estudo de Caso

Ex.mo Senhor Diretor

Agrupamento de Escolas

De Borba

Assunto: Requerimento para implementação de Estudo de Caso

Elsa de Fátima Velez Severo Rôlo, docente do Quadro de Zona Pedagógica, Grupo 110, portadora do Cartão de Cidadão nº 10613701, vem por este meio requerer a V. Exa. autorização para desenvolver um Estudo de Caso, no quarto ano, turma A, sobre os contributos da Programação para o desenvolvimento do pensamento computacional em alunos do 1.º ciclo do Ensino Básico realizado no âmbito da elaboração da dissertação de Mestrado em Administração, Regulação e Políticas Educativas pela Universidade de Évora.

Os instrumentos de recolha e pesquisa de dados a utilizar serão os seguintes:

- Observação de aulas;
- Análise dos registos e projetos dos alunos;
- Gravação dos relatos dos alunos, de forma a facilitar a recolha dos dados. No final de cada sessão, as gravações devem serão reproduzidas em escrito de modo fidedigno, permitindo a obtenção de dados válidos e confiáveis (Someren,et al., 1994).

Certa da importância de uma reflexão partilhada acerca da existência e desenvolvimento de novas competências associadas à organização do pensamento e da contribuição do referido estudo para a operacionalização positiva de práticas pedagógicas, pede deferimento.

Borba, 12 de novembro de 2018

A Requerente _____

Anexo III – Grelhas de Observação

Grelhas de Observação – Estudo de Caso – 4ºA

(propostas por Brenann (2014), traduzidas por Ramos (2015))

Experimentar e Interagir / Grupos de trabalho – Alunos/as		Descreve a construção do projeto, passo a passo.			Enumera exemplos que experimentou ao longo da elaboração do projeto			Apresenta as revisões que foram sendo feitas.			Descreve as diferentes abordagens que experimentou no projeto, ou quando tentou fazer algo novo.		
		Padrões de desempenho			Padrões de desempenho			Padrões de desempenho			Padrões de desempenho		
		Baixo (Fornece uma descrição elementar da construção do projeto, mas não detalha aspetos específicos do mesmo.)	**Médio** (Faz uma descrição genérica do projeto, de forma ordenada.)	**Alto-Elevado** (Fornece detalhes sobre as diferentes componentes dum projeto específico e descreve o modo como foram desenvolvidos, de forma ordenada.)	**Baixo** (Não apresenta exemplos específicos do que experimentou.)	**Médio** (Deixa transparecer de forma genérica que experimentou outras coisas no projeto.)	**Alto-Elevado** (Fornece exemplos específicos de outras coisas que foi experimentando no projeto.)	**Baixo** (Afirma não ter feito revisões, ou afirma ter feito algumas mas não exemplifica.)	**Médio** (Descreve uma revisão específica que fez ao projeto.)	**Alto-Elevado** (Descreve aspetos específicos de coisas que acrescentou ao projeto e justifica.)	**Baixo** (Não revela evidências de ter experimentado algo novo.)	**Médio** (Fornece um exemplo de algo novo que experimentou no projeto.)	**Alto-Elevado** (Descreve com detalhe coisas novas que experimentou no projeto.)
1	Henrique Passinhas e Miguel Catarino			X			X		X				X
2	Francisco Chouriço e Joana Infante		X		X				X		X		
3	Beatriz Maltinha e Rodrigo Gonçalves			X			X			X			X
4	Afonso Quaresma e Bruna Catarino		X		X			X			X		
5	Débora Bento, Gabriel Borrego e Guilherme Ganito		X		X			X			X		
6	Beatriz Romão e Leandro Silva		X		X				X		X		
7	Filipa Catarino e Soraia Fialho		X			X		X			X		
8	Gabriela Sola e Lara Franco		X			X		X			X		

Grelhas de Observação – Estudo de Caso – 4ºA

(propostas por Brenann (2014), traduzidas por Ramos (2015))

Testar e Corrigir / Grupos de Trabalho - Alunos/as		Descreve o que aconteceu com o projeto de diferente em relação ao pretendido.			Descreve de que forma foi feita a leitura do código para encontrar a causa do problema.			Descreve que alterações fez e como as testou para verificar os resultados.			Descreve outras formas de resolver o problema.		
		Padrões de desempenho			Padrões de desempenho			Padrões de desempenho			Padrões de desempenho		
		Baixo (Não descreve o que resultou diferente em relação ao pretendido.)	**Médio** (Descreve o que correu mal no projeto, mas não o que pretendia fazer.)	**Alto-Elevado** (Dá um exemplo detalhado do que aconteceu e o que pretendia, quando executa o programa.)	**Baixo** (Não descreve um problema.)	**Médio** (Descreve como faz a leitura mas não apresenta um exemplo especifico de encontrar um problema no código.)	**Alto-Elevado** (Descreve como faz a leitura e apresenta um exemplo especifico de encontrar um problema no código)	**Baixo** (Não descreve que problemas teve ou a solução)	**Médio** (Fornece um exemplo genérico sobre as alterações feitas e os testes feitos para verificar o funcionamento.)	**Alto-Elevado** (Fornece um exemplo especifico sobre as alterações feitas e os testes feitos para verificar o funcionamento.)	**Baixo** (Não apresenta uma forma para encontrar uma solução para o problema.)	**Médio** (Apresenta uma forma genérica para encontrar uma solução para o problema.)	**Alto-Elevado** (Apresenta um exemplo especifico de como encontrar uma solução para o problema.)
1	Henrique Passinhas e Miguel Catarino			x			x			x			x
2	Francisco Chouriço e Joana Infante			x		x			x			x	
3	Beatriz Maltinha e Rodrigo Gonçalves			x			x			x			x
4	Afonso Quaresma e Bruna Catarino		x			x			x			x	
5	Débora Bento, Gabriel Borrego e Guilherme Ganito		x			x		x			x		
6	Beatriz Romão e Leandro Silva			x			x			x			x
7	Filipa Catarino e Soraia Fialho			x		x			x			x	
8	Gabriela Sola e Lara Franco		x			x		x			x		

Grelhas de Observação – Estudo de Caso – 4ºA

(propostas por Brenann (2014), traduzidas por Ramos (2015))

Testar e Corrigir / Grupos de Trabalho - Alunos/as		Descreve se encontrou inspiração em outros projetos e na leitura do código disponível.			Descreve como selecionou uma parte de outro projeto e como a adaptou ao seu.			Como refere/cita as pessoas cujo trabalho inspirou o seu.		
		Padrões de desempenho			Padrões de desempenho			Padrões de desempenho		
		Baixo (Descreve como desenvolveu as ideias ou em que projetos se inspirou.)	**Médio** (Fornece uma descrição geral de um projeto que o inspirou.)	**Alto-Elevado** (Dá um exemplo específico do projeto que o/a inspirou.)	**Baixo** (Não descreve como adaptou as ideias, scripts ou recursos de outros projetos.)	**Médio** (Identifica scripts, ideias ou recursos que adaptou de outros projetos.)	**Alto-Elevado** (Fornece exemplos específicos de scripts, ideias ou recursos que ele adaptou de outros projetos e como.)	**Baixo** (Não identifica as fontes e os autores em que se inspirou para a realização do seu projeto.)	**Médio** (Identifica as fontes e os autores em que se inspirou para a realização do seu projeto.)	**Alto-Elevado** (Documenta no projeto as fontes e os autores que o inspiraram.)
1	Henrique Passinhas e Miguel Catarino			x			x	x		
2	Francisco Chouriço e Joana Infante		x		x			x		
3	Beatriz Maltinha e Rodrigo Gonçalves			x			x	x		
4	Afonso Quaresma e Bruna Catarino		x			x		x		
	Débora Bento, Gabriel Borrego e Guilherme Ganito		x		x			x		
6	Beatriz Romão e Leandro Silva		x			x		x		
7	Filipa Catarino e Soraia Fialho		x			x		x		
8	Gabriela Sola e Lara Franco		x		x			x		

Anexo IV – Análise dos Jogos pelo Dr. Kodu

Jogo 1
Como se chama o nome do ficheiro?d5d86bd2-ff50-4d76-998b-8f96f5eaa3b1

Castle 1
---
sensor.always

Isto é uma condição.

Kodu 1
---
sensor.always

Isto é uma condição.
---
actuator.say
modifier.once
modifier.pink
---
sensor.always

Isto é uma condição.
---
actuator.say
modifier.once
modifier.blue
---
sensor.eyes

Isto é uma condição.

Fan 1
---
sensor.always

Isto é uma condição.
---
actuator.push
modifier.strongly
modifier.strongly

modifier.strongly

Blimp 1
---
sensor.keyboard

Isto é uma condição.
---
actuator.movement
filter.ArrowKeys
---
sensor.bumpers

Isto é uma condição.
---
actuator.eat
filter.coin
modifier.it
---
sensor.bumpers

Isto é uma condição.
---
actuator.grab
filter.heart
filter.red
modifier.it
---
sensor.bumpers

Isto é uma condição.
---
actuator.give

Trata-se de uma lógica booleana (intendada).
---
filter.rover
filter.blue
---
sensor.bumpers

Isto é uma condição.
---
actuator.grab

filter.star
filter.yellow
modifier.it
---
sensor.bumpers

Isto é uma condição.
---
actuator.give

Trata-se de uma lógica booleana (intendada).
---
filter.rover
filter.pink
---
sensor.keyboard

Isto é uma condição.
---
actuator.movement
filter.F1key
modifier.quickly
modifier.quickly
modifier.quickly
---
sensor.keyboard

Isto é uma condição.
---
actuator.camera.firstperson
filter.Ckey
---
sensor.bumpers

Isto é uma condição.
---
actuator.score
filter.coin
modifier.scorebucket.local.j
modifier.010points
---
sensor.scored

Isto é uma condição.
---

actuator.sound
filter.scorebucket.local.j
filter.scoregteq
filter.020points
filter.020points
modifier.mu_kodu
---
sensor.scored

Isto é uma condição.
---
actuator.victory
filter.scorebucket.local.j
filter.scoreequals
filter.050points
modifier.player.1
---
sensor.keyboard

Isto é uma condição.
---
actuator.launch
filter.Spacekey
modifier.strongly
modifier.once
modifier.angle.high

Heart 1

Heart 2

Star 1
actuator.glow
modifier.me
modifier.yellow

Rover 1

Rover 2

Coin 1

Coin 2

Coin 3

Balloon 1
---
sensor.always

Isto é uma condição.
---
actuator.movement
modifier.path.black
---
sensor.always

Isto é uma condição.
---
actuator.say
modifier.orange

Jet 1
---
sensor.always

Isto é uma condição.
---
actuator.say
modifier.blue
---
sensor.always

Isto é uma condição.
---
actuator.movement
modifier.quickly

Factory 1

Hut 1

Cycle 1
---
sensor.always

Isto é uma condição.
---
actuator.movement
modifier.path.white
modifier.quickly
---
sensor.bumpers

Isto é uma condição.
---
actuator.zap
filter.fastbot
modifier.me

Cycle 2
---
sensor.always

Isto é uma condição.
---
actuator.movement
modifier.path.white
modifier.slowly
---
sensor.bumpers

Isto é uma condição.
---
actuator.pop
filter.fastbot
modifier.me
---
sensor.bumpers

Isto é uma condição.

---
actuator.pop
filter.fastbot
modifier.it

Rover 3
---
sensor.eyes

Isto é uma condição.
---
actuator.scan
filter.windblimp
filter.closeby
modifier.once

Rock 1
---
sensor.timer

Isto é uma condição.
---
actuator.gameover
filter.60seconds

Jogo 2
Como se chama o nome do ficheiro?9791387d-dc9c-4eb9-b8b1-2a85bdda8988

Hut 1

Hut 2

Hut 3

Hut 4

Hut 5

Hut 6

Hut 8

Hut 9

Hut 10

Castle 1
---
sensor.bumpers

Isto é uma condição.
---
actuator.victory
filter.coin

Kodu 1

Coin 11

Cycle 1
---
sensor.keyboard

Isto é uma condição.
---
actuator.movement
filter.ArrowKeys
---
sensor.bumpers

Isto é uma condição.
---
actuator.gameover
filter.hut
---

sensor.bumpers

Isto é uma condição.
---
actuator.victory
filter.coin

Castle 2

Hut 7

Tree 1

Clam 1

Heart 1

Ball 1

Jogo 3
Como se chama o nome do ficheiro?0d7f2f0d-02cc-4184-91d0-99c82b7c4d5e

---
Cloud 1

Cloud 2

Hut 1

Castle 1

Kodu 1

Iceberg 1

Kodu 2

Iceberg 2

Iceberg 3

FlyFish 1
---
sensor.keyboard

Isto é uma condição.
---
actuator.movement
filter.ArrowKeys
modifier.quickly
---
sensor.bumpers

Isto é uma condição.
---
actuator.grab
filter.iceBerg
---
sensor.eyes

Isto é uma condição.
---
actuator.victory
filter.not

Isto é lógica booleana.
---
filter.iceBerg
---
sensor.timer

Isto é uma condição.
---
actuator.gameover
filter.60seconds

Rover 1
---
sensor.keyboard

Isto é uma condição.
---
actuator.movement
filter.WASDKeys
---
sensor.bumpers

Isto é uma condição.
---
actuator.grab
filter.flyfish
modifier.it
---
sensor.bumpers

Isto é uma condição.
---
actuator.drop
filter.hut
actuator.victory

Trata-se de uma lógica booleana (intendada).

Jogo 4
Como se chama o nome do ficheiro?40e77edc-c189-4c38-a0bf-7ceddce06723

Castle 1

Kodu 1

Apple 1

Apple 2

Tree 1

Hut 1
---
sensor.got
actuator.grab
filter.hut

Ball 1

Apple 3

Kodu 2
---
sensor.eyes

Isto é uma condição.
---
actuator.grab
filter.coin
---
sensor.got
actuator.victory
filter.coin

Apple 4

Apple 5

Coin 11

Coin 19

Ball 2

Ball 3

Kodu 3
---
sensor.keyboard

Isto é uma condição.
---
actuator.movement
filter.ArrowKeys
---
sensor.bumpers

Isto é uma condição.
---
actuator.grab
filter.coin
---
sensor.bumpers

Isto é uma condição.
---
actuator.drop
filter.bokubot
filter.white

Iceberg 1

Iceberg 2

Iceberg 3

Iceberg 4

Iceberg 5

Iceberg 6

Iceberg 7

Iceberg 8

Iceberg 9

Iceberg 10

Iceberg 11

Iceberg 12

Iceberg 13

Iceberg 14

Iceberg 15

Iceberg 16

Iceberg 17

Iceberg 18

Ignea 1

Jogo 5
Como se chama o nome do ficheiro?10ab802c-49b9-496e-baa8-f102679dbf93

Kodu 1
---

sensor.keyboard

Isto é uma condição.
----
actuator.movement
filter.ArrowKeys
---
sensor.keyboard

Isto é uma condição.
----
actuator.jump
filter.D0key
---
sensor.bumpers

Isto é uma condição.
----
actuator.grab
filter.coin
filter.red
modifier.it
---
sensor.bumpers

Isto é uma condição.
----
actuator.drop
filter.hut
filter.red
---
sensor.bumpers

Isto é uma condição.
----
actuator.grab
filter.coin
filter.white
modifier.it
---
sensor.bumpers

Isto é uma condição.
----
actuator.drop

filter.hut
filter.white
---
sensor.bumpers

Isto é uma condição.
---
actuator.grab
filter.coin
filter.blue
modifier.it
---
sensor.bumpers

Isto é uma condição.
---
actuator.drop
filter.hut
filter.blue
actuator.score

Trata-se de uma lógica booleana (intendada).
---
modifier.scorebucket.color.red
modifier.050points
---
sensor.scored

Isto é uma condição.
---
actuator.victory
filter.scorebucket.color.red
filter.scoreequals
filter.100points
filter.050points
---
sensor.bumpers

Isto é uma condição.
---
actuator.gameover
filter.tree

Castle 1

Hut 1

Tree 1

Hut 2

Hut 3

Tree 1

Hut 4

Turtle 1

Fish 1

Fish 2

Fish 3

Fish 4

Coin 1

Coin 2

Coin 3

Jogo 6
Como se chama o nome do ficheiro?9b995ba8-4dbc-4af6-b914-2a35cd80d54b

---
Castle 1

Tree 1

Apple 1

Apple 2

Apple 3

Apple 4

Apple 5
---
sensor.keyboard

Isto é uma condição.

Apple 7

Apple 8

Apple 9

Apple 10

Apple 11

Rock 1

Sedimentar 1

Desonhecida (Ignea) 1

Desonhecida (Ignea) 2

Rock 2
---
sensor.always

Isto é uma condição.
---
actuator.worldskychange

Kodu 1

Coin 1

Coin 2

Coin 3

Tree 2

Rock 3

Star 1

Star 2

Cloud 1

Jet 1
---
sensor.keyboard

Isto é uma condição.
---
actuator.movement
filter.WASDKeys
---
sensor.bumpers

Isto é uma condição.
---
actuator.eat
filter.fruit
actuator.score

Trata-se de uma lógica booleana (intendada).
---
modifier.scorebucket.color.red
modifier.010points
---
sensor.bumpers

Isto é uma condição.
---
actuator.gameover
filter.rock
---
sensor.eyes

Isto é uma condição.
---
actuator.victory
filter.fruit
filter.not

Isto é lógica booleana.
---
modifier.player.1
modifier.red
---
sensor.scored

Isto é uma condição.
---
actuator.victory
filter.scorebucket.color.red
filter.scoreequals
filter.100points

Star 3

Star 4

Jogo 7
Como se chama o nome do ficheiro?6cc58310-3d77-42c1-9509-07b513ae9c8d

---
Rover 1
---
sensor.keyboard

Isto é uma condição.
---
actuator.movement
filter.ArrowKeys
---
sensor.bumpers

Isto é uma condição.
---
actuator.say
filter.fruit
filter.purple
---
sensor.bumpers

Isto é uma condição.
---
actuator.victory
filter.coin
filter.Grey

Castle 1

Castle 2

Castle 3

Kodu 1
actuator.say

Apple 1

Apple 2

Apple 3

Apple 4

Coin 1

Coin 2

Coin 3

Tree 1

Octopus 1
actuator.say

Tree 1

Tree 1

Tree 1

Ball 1

Apple 5

Apple 6

Coin 4

Apple 7

Apple 8

Heart 1

Jogo 8
Como se chama o nome do ficheiro?5e6ba27b-ccfe-41cc-b3d0-df1c0653b3f5

Cloud 1

Cloud 2

Hut 1

Castle 1

Kodu 1

Iceberg 1

Kodu 2

Iceberg 2

Iceberg 3

FlyFish 1
---
sensor.keyboard

Isto é uma condição.
---
actuator.movement
filter.ArrowKeys
modifier.quickly
---
sensor.bumpers

Isto é uma condição.
---
actuator.grab
filter.iceBerg
---
sensor.eyes

Isto é uma condição.
---
actuator.victory
filter.not

Isto é lógica booleana.
---
filter.iceBerg
---
sensor.timer

Isto é uma condição.
---
actuator.gameover
filter.60seconds

Rover 1
---
sensor.keyboard

Isto é uma condição.
---
actuator.movement
filter.WASDKeys
---
sensor.bumpers

Isto é uma condição.
---
actuator.grab
filter.flyfish
modifier.it
---
sensor.bumpers

Isto é uma condição.
---
actuator.drop
filter.hut
actuator.victory

Trata-se de uma lógica booleana (intendada).