O impacto dos paradigmas e linguagens de programação no ensino intermediário da programação de computadores

Gabriel Antoine Louis Paillard[1]

Leonardo Oliveira Moreira[2]

Resumo

Algoritmos e programação de computadores são conceitos abordados nos semestres iniciais em diversos cursos das áreas das ciências exatas e também em cursos que adotam o projeto e implementação de tecnologias digitais. Diversos trabalhos discutem e tratam dos desafios enfrentados no ensino da programação de computadores e relatam experiências da aplicação de metodologias de ensino. Já outros trabalhos mostram o uso de uma linguagem de programação específica e como a linguagem pode contornar ou amenizar alguns dos desafios encontrados no ensino. A maioria dos trabalhos encontrados foca ou destaca os desafios encontrados no ensino de algoritmos e programação de computadores em disciplinas introdutórias. Diante disso, surge uma oportunidade de investigação para verificar o impacto dos paradigmas de programação e das linguagens de programação no ensino intermediário dos conceitos de algoritmos e programação de computadores. Este artigo tem como objetivo principal expor relatos de experiências obtidas no ensino da programação de computadores envolvendo duas linguagens de programação de paradigmas de programação distintos. O objeto de estudo da pesquisa envolveu uma disciplina do ensino superior que trabalha conhecimentos intermediários de algoritmos e programação de computadores em um curso de natureza interdisciplinar. De acordo com os resultados obtidos no estudo, houve aspectos positivos e negativos em ambos os paradigmas de programação e também das linguagens de programação que foram abordadas neste trabalho.

Palavras-chaves: **Algoritmos. Paradigmas e Linguagens. Ensino de Programação**.

[1] Doutor em Informática pela Université Paris 13 (Paris-Nord, França). Professor Efetivo da Universidade Federal do Ceará (UFC), Fortaleza, Ceará, Brasil

[2] Doutor em Ciência da Computação pela Universidade Federal do Ceará (UFC) -Fortaleza, Ceará,Brasil. Professor Efetivo da Universidade Federal do Ceará (UFC), Fortaleza, Ceará, Brasil

INTRODUÇÃO

O conceito de algoritmos pode ser definido com um conjunto de instruções bem definidas e sem ambiguidade para resolução de problemas em um espaço finito de tempo (FORBELLONE; EBERSPÄCHER, 2005). Para ensinar a programação de computadores, os cursos iniciam o ensino com a apresentação dos conceitos de algoritmos para que o aluno inicie e se familiarize com o raciocínio lógico para resolução de problemas por meio dos programas de computadores. Gomes et al. (2008) discutem que esse contato inicial já apresenta uma enorme taxa de insucesso na aprendizagem, destacando que existe uma grande dificuldade, por parte dos alunos, na compreensão destes conceitos iniciais.

Para tratar esses problemas iniciais na aprendizagem do ensino da programação de computadores, diversas iniciativas foram propostas e empregadas nos cursos ou minicursos que ensinam programação de computadores. Dentre as iniciativas podem-se destacar o uso de metodologias que favorecem a aprendizagem colaborativa (MARINHO et al., 2016; EDUARDO et al., 2016), ferramentas e atividades lúdicas (SILVA et al., 2016) e a escolha por uma linguagem de programação que simplifique o entendimento e que atenda aos requisitos esperados nas disciplinas de programação de computadores (RODRIGUES, 2013).

As linguagens de programação são utilizadas para descrever as instruções, advindas dos algoritmos, para um computador (SEBESTA, 2011). Cada linguagem de programação pode seguir um ou mais paradigmas de programação. Um paradigma de programação determina a visão ou a forma que o programador deve seguir para a estruturação e execução de um programa de computador. Nos cursos de graduação, principalmente os tradicionais, ainda predomina a adoção de linguagens de programação que utilizam o paradigma de programação estruturado para o ensino inicial da programação de computadores. Não existe uma padronização na linguagem adotada nos cursos, principalmente no primeiro contato do aluno com a programação de computadores. Portanto, podem surgir casos onde o aluno tenha que compreender a forma de programar em um ou mais paradigmas de programação.

Este artigo tem como objetivo principal exibir relatos de experiências no ensino da programação de computadores envolvendo duas linguagens de programação de paradigmas distintos. Com isso, analisar e entender o impacto dos paradigmas de programação no ensino intermediário da programação de computadores no ensino superior.

REFERENCIAL TEÓRICO

Esta seção comenta sobre os conhecimentos necessários para o entendimento dos conceitos apresentados nesta pesquisa. Neste sentido, são discutidos os conceitos de algoritmos e programação

de computadores, as linguagens de programação, os paradigmas de programação e duas linguagens de programação bastante difundidas nos cursos que utilizam os conhecimentos em programação de computadores.

Algoritmo pode ser definido como um conjunto finito de instruções que expressa a resolução de um determinado problema (FORBELLONE; EBERSPÄCHER, 2005). Neste sentido, o objetivo do algoritmo é expressar a ideia, a forma ou a metodologia para resolver um determinado problema, abstraindo as complexidades e particularidades sintáticas das linguagens de programação. Programas de computadores e algoritmos não são sinônimos, pois as instruções definidas no algoritmo devem ser mapeadas e escritas em uma linguagem de programação para a criação de um programa de computador.

Uma linguagem de programação pode ser definida como um método que emprega regras sintáticas e semânticas para comunicar instruções ao computador (FISCHER; GRODZINSKY, 1993). Neste sentido, as linguagens de programação podem ser utilizadas para implementar algoritmos nos computadores. Comumente, uma linguagem de programação é composta por um conjunto palavras e regras para formulação do código-fonte. O código-fonte, após concluído e as regras verificadas, é traduzido para linguagem de máquina para ser executado pelo computador. As linguagens de programação podem ser classificadas como linguagens compiladas e interpretadas em relação a forma de tradução do código-fonte para código de máquina (SEBESTA, 2011).

As linguagens compiladas traduzem todo o código-fonte em código de máquina para depois executar no computador. Já as linguagens interpretadas os trechos do código-fonte são traduzidos e tem sua execução imediata. Neste sentido, as linguagens interpretadas tendem a produzir programas mais lentos que as linguagens compiladas, pois esse conjunto de etapas de tradução ao longo do código-fonte torna a execução dos programas mais sobrecarregada. Além disso, as linguagens compiladas garantem que o código-fonte está sintaticamente correto se o código de máquina for produzido. Em contrapartida, os programas em linguagens interpretadas tendem a ser mais portáveis para outros ambientes computacionais do que as linguagens compiladas, pois os interpretadores podem atuar como mediadores entre o código-fonte e o sistema operacional ou hardware físico. Nas linguagens interpretadas os erros são identificados em tempo de execução da instrução que detém o erro.

O paradigma de programação tem como objetivo principal determinar a visão ou o modo que o programador deve seguir para a estruturação e execução de um programa de computador desenvolvido em uma linguagem de programação (SEBESTA, 2011). Existem diversos paradigmas de programação e as linguagens podem seguir os modos ou regras de um ou mais destes paradigmas. Por

exemplo, a linguagem JavaScript, originalmente implementada como parte dos navegadores web, possui suporte aos paradigmas estruturado e orientado a objetos.

O paradigma de programação estruturado é a forma de programação de computadores que se baseia em três estruturas básicas de controle e nos aspectos de modularização do programa em subprogramas (SEBESTA, 2011). Neste paradigma de programação todo o problema computacional pode ser resolvido pelas três estruturas básicas de controle: sequência, seleção e iteração. Já os aspectos de modularização pregam que um problema maior pode ser subdividido ou simplificado em problemas menores. Neste sentido, um programa é dividido em subprogramas que são expressos pelos procedimentos e funções. Essa redução de um programa em subprogramas favorece ao reuso de código. Por exemplo, uma função raiz quadrada pode ser definida uma vez como um subprograma e ser utilizada em diversos outros programas que utilizam funções matemáticas em suas soluções.

Já o paradigma de programação orientada a objetos modela um problema de programação como uma coleção de objetos que se comunicam por meio das trocas de mensagens (BARNES; KOLLING, 2009). Desta forma, o programador neste paradigma de programação tenta resolver os problemas por meio da identificação dos objetos que compõem os problemas e como eles interagem. Objetos que possuem características, informações e comportamentos em comum são definidos como objetos de uma mesma classe. Neste sentido, a classe é composta por um conjunto de código que especifica o que o objeto deve fazer. Classes podem herdar características informações e comportamentos de outras classes, favorecendo também o reuso de código. É válido ressaltar que as três estruturas básicas utilizadas no paradigma estruturado também são utilizadas no paradigma orientado a objetos. O paradigma orientado a objetos pode ser visto como um complemento ou evolução do estruturado, empregando algumas características a mais de abstração, reuso e a forma de modelar os problemas para simplificar o entendimento e a codificação.

A linguagem C, criada em 1972 por Dennis Ritchie e Brian Kernighan, é uma das linguagens mais populares e que influenciou a criação de diversas outras linguagens de programação (SEBESTA, 2011). Um dos aspectos interessantes da linguagem C é o desempenho dos programas desenvolvidos, pois é possível implementar instruções em um nível mais próximo da máquina (PINHEIRO, 2012). Pelo fato de programar em um nível mais próximo da máquina, a linguagem C permite a implementação de uma grande variedade de soluções de software que vão desde de sistemas operacionais até programas de usuário. Todo esse poder da linguagem C pode acarretar em problemas no aprendizado da linguagem, pois existem uma variedade de funções, estruturas e recursos de manipulação do hardware e do sistema operacional. Os programas escritos na linguagem C passam por um processo de compilação, que transforma o código-fonte em um código executável para um determinado sistema operacional (SEBESTA, 2011). Neste sentido, se um programa for desenvolvido

para o sistema operacional Windows, deve-se usar algum compilador que produza código executável para Windows. Tal aspecto de compilação da linguagem C torna dificultoso a portabilidade um mesmo programa executável em diferentes sistemas operacionais.

Java é uma linguagem de programação considerada híbrida, em termos de compilação e interpretação, e que segue o paradigma orientado a objetos (SEBESTA, 2011). O código-fonte escrito na linguagem Java passa por uma etapa de compilação para geração dos *bytecode*. O *bytecode* é um código intermediário entre o código-fonte e o programa, contendo as vantagens de uma linguagem compilada e sem dependência de um sistema operacional ou hardware físico. Esta característica torna os programas escrito em Java portáveis para qualquer sistema operacional e hardware físico, pois o *bytecode* gerado é interpretado por uma máquina virtual que abstrai as particularidades de um sistema operacional ou hardware físico. Neste sentido, um mesmo programa Java pode executar em qualquer computador que possua a máquina virtual Java instalada. Com isso, é possível portar um programa Java para computadores de diversas características, celulares, geladeiras, televisores, *tablets*, sistemas embarcados etc.

METODOLOGIA

Nesta seção são apresentados o objeto de estudo e os aspectos metodológicos empregados nesta pesquisa. Além disso, o instrumento e a forma de como os dados foram obtidos também são mostrados nesta presente seção.

A disciplina Estrutura de Dados é considerada de nível intermediário no tocante aos conhecimentos em algoritmos programação de computadores. Geralmente, essa disciplina ocorre entre o terceiro e quinto semestre nos cursos de graduação que possuem vínculos com as áreas de computação. Neste sentido, o público-alvo desta disciplina já possui conhecimentos básicos em algoritmos, programação de computadores e pelo menos uma linguagem de programação. Como objeto de estudo deste presente trabalho, foram utilizados os dados obtidos na turma da disciplina Estrutura de Dados ministrada no semestre de 2016.2 no curso Bacharelado em Sistemas e Mídias Digitais (SMD) da Universidade Federal do Ceará (UFC).

A metodologia utilizada neste trabalho, em um primeiro momento, se deu pelo o uso de um questionário como instrumento de coleta de dados para que os discentes fornecessem informações sobre suas experiências em relação ao impacto das linguagens e paradigmas de programação na assimilação do conteúdo proposto na turma que foi objeto do estudo. Diante das informações colhidas

na pesquisa por meio do questionário, foi feita a elaboração de uma pesquisa explicativa (GIL, 2007) e contendo aspectos do método fenomenológico (SIANI, 2016) para análise e discussão dos resultados.

O questionário foi elaborado contendo algumas questões subjetivas e outras objetivas seguindo a escala Likert (MAFRA, 1999). A escala Likert é uma forma de medir o nível de concordância ou não a uma afirmação, utilizando opções de resposta que variam de um extremo a outro. Um esquema típico da escala Likert aplicada a um item de questão é: (1) Não concordo totalmente; (2) Não concordo parcialmente; (3) Indiferente; (4) Concordo parcialmente; (5) Concordo plenamente.

A pesquisa foi submetida aos discentes da disciplina, foram excluídos da pesquisa os alunos que efetuaram o trancamento ou abandonaram a disciplina. Neste sentido, a amostra foi composta apenas dos alunos ativos e que estavam assíduos no decorrer da disciplina, para garantir que os alunos tiveram o contato com ambos os paradigmas de programação. No questionário não foram incluídos campos que identificassem os participantes da pesquisa. A Tabela 1 sintetiza as sete questões, e seus respectivos tipos, que foram elaboradas e aplicadas aos discentes da turma de Estrutura de Dados no semestre de 2016.2.

Tabela 1. Tabela que exibe o conteúdo do questionário aplicado aos discentes

Identificador	Enunciado	Tipo de Resposta
Q01	Quais as linguagens de programação que você já implementou algum algoritmo?	Subjetiva
Q02	O paradigma de programação adotado pela linguagem de programação simplificou a implementação dos exercícios propostos na disciplina?	Escala Likert
Q03	Os aspectos sintáticos das linguagens de programação contribuíram para facilitar a implementação dos exercícios propostos na disciplina?	Escala Likert
Q04	Dentre as linguagens usadas na disciplina, qual a linguagem que você acha que simplificou a implementação dos exercícios propostos na disciplina?	Objetiva: (a) C (b) Java
Q05	Quais os aspectos positivos e negativos da linguagem de programação C?	Subjetiva
Q06	Quais os aspectos positivos e negativos da linguagem de programação Java?	Subjetiva
Q07	Qual o paradigma de programação que você acha mais	Objetiva:

	adequado para implementação dos exercícios propostos na disciplina?	(a) Estruturado (b) Orientado a Objetos

RESULTADOS E DISCUSSÕES

A análise dos dados foi feita por meio dos resultados dos questionários que foram submetidos aos alunos. Dos 10 alunos que foram convidados a participar da pesquisa, 6 alunos aceitaram e responderam o questionário. A questão 01 tem como objetivo esclarecer o perfil ou nível da amostra em relação aos conhecimentos de linguagens de programação. Mesmo que a pergunta não revele a fluência do aluno com a linguagem de programação, mas pelo menos informa que o aluno já teve algum contato e conseguiu implementar algum algoritmo. A Tabela 2 exibe informações sobre as linguagens de programação que os alunos possuem algum conhecimento.

Tabela 2. Discentes e seus contatos com linguagens de programação

Identificador	Linguagens de Programação
A01	C, Java, JavaScript, Ruby e Processing
A02	C, C#, Pascal, Assembly 8086, Assembly 8051, JavaScript, Java e GML
A03	C, Java e JavaScript
A04	PHP, JavaScript, Java, C, C++ e C#
A05	C, Java, JavaScript, C++ e Python
A06	JavaScript, C, C#, Java e Processing

Pela análise das respostas obtidas na questão Q01 pode-se observar que os todos os alunos, que responderam, conheciam pelo menos três linguagens de programação. Dentre as linguagens que foram relatadas, todos os alunos tinham conhecimentos em pelo menos dois paradigmas de programação. Além da análise dos paradigmas, pode-se observar que as linguagens que foram informadas possuem aplicabilidade no desenvolvimento de sistemas convencionais e também de sistemas web, mostrando que os alunos tiveram alguma experiência em programação para ambos os ambientes. Neste sentido, pode-se dizer que a amostra de discentes, que responderam ao questionário, possuem uma boa experiência em programação.

A questão 02 tem como objetivo verificar se os paradigmas de programação adotados pela Linguagem C e Java simplificaram ou não a implementação dos exercícios propostos na disciplina. A Figura 1 mostra um gráfico com os resultados obtidos.

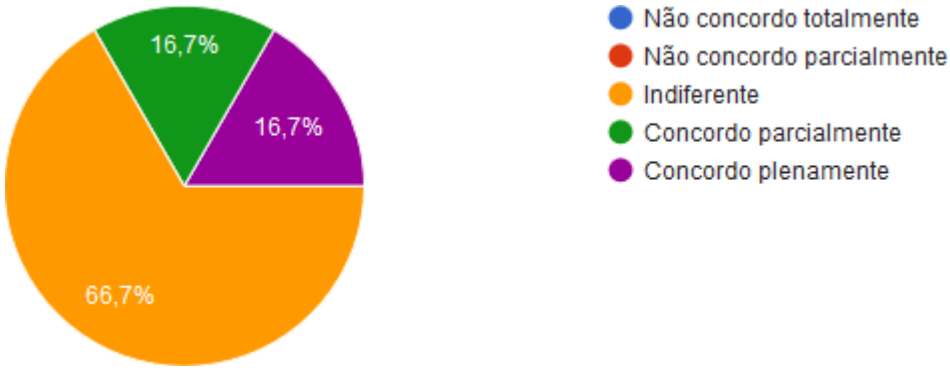

Figura 1. Resultados obtidos na questão Q02

Pela observação das respostas obtidas na questão Q02, pode-se perceber que a maioria dos alunos são indiferentes quanto ao paradigma adotado pela linguagem de programação em relação a resolução dos exercícios propostos na disciplina. No entanto, houve respostas que tendem a mostrar que o paradigma adotado pela linguagem pode simplificar a resolução dos exercícios propostos em Estrutura de Dados. De acordo com as respostas, 4 alunos responderam que são indiferentes, 1 aluno assinalou que concorda parcialmente e 1 aluno informou que concorda plenamente. É válido ressaltar que todos os alunos tinham conhecimentos em linguagens que abordam os dois paradigmas pesquisados, conforme mostrado pelas respostas da questão Q01.

A questão Q03 visa verificar se os aspectos sintáticos das linguagens de programação contribuíram para facilitar a implementação dos exercícios propostos na disciplina. A Figura 2 exibe um gráfico com os resultados obtidos.

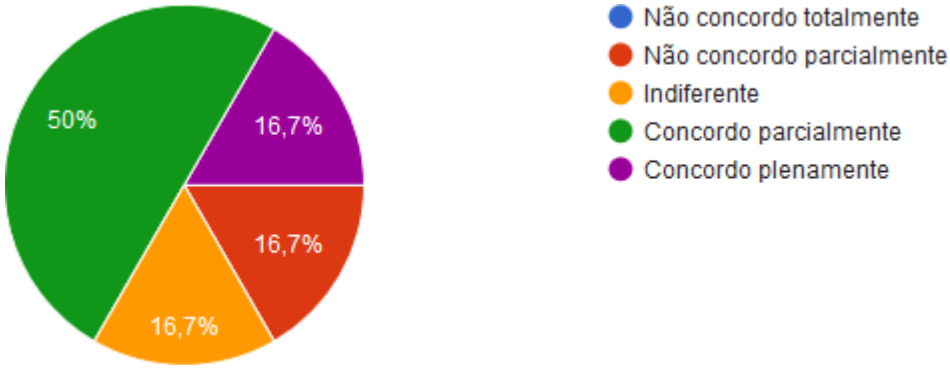

Figura 2. Resultados obtidos na questão Q03

De acordo com resultados, 3 alunos informaram que concordam parcialmente, 1 aluno informou que concorda plenamente, 1 aluno se mostrou indiferente e 1 aluno não concordou parcialmente. Diante destes resultados, pode-se perceber que os alunos, em sua maioria, acham que os aspectos sintáticos das linguagens podem contribuir na facilidade de implementação dos algoritmos. As questões Q04, Q05 e Q06 podem esclarecer melhor os resultados obtidos nas respostas da questão Q03.

O objetivo da questão Q04 é verificar quais das duas linguagens de programação, abordadas na disciplina, foi a que mais simplificou a implementação dos algoritmos propostos na disciplina. A Figura 4 mostram os resultados obtidos por meio de um gráfico.

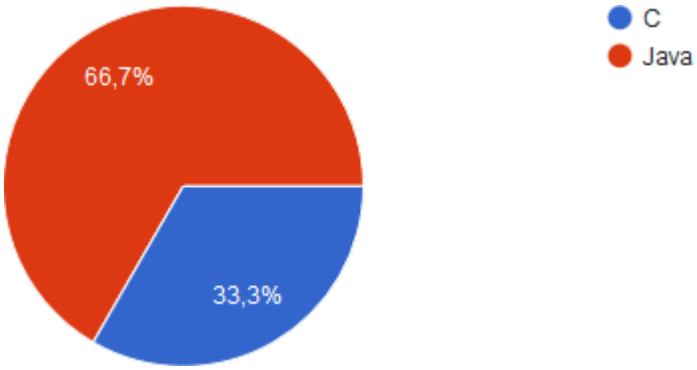

Figura 3. Resultados obtidos na questão Q04

Pelos resultados, 4 alunos responderam que Java foi a que simplificou, já 2 alunos informaram que C foi a linguagem que mais simplificou. Pelo fato de Java adotar o paradigma de orientação a objetos e abstrair algumas complexidades, em particular o gerenciamento da memória, pode acarretar em uma simplicidade maior na implementação dos algoritmos. Com isso, os exercícios podem ser resolvidos com uma maior atenção ao problema proposto e não nas particularidades dos recursos computacionais da máquina física, por exemplo o gerenciamento de dados na memória.

A questão Q05 explora as vantagens e desvantagens da linguagem C na visão dos alunos. De acordo com os resultados, os aspectos positivos mais recorrentes foram: (a) linguagem mais adequada para a proposta da disciplina; (b) atenção ao gerenciamento da memória e da máquina física; e (c) menos abstração na implementação das estruturas de dados. Em relação às desvantagens, os aspectos mais recorrentes foram: (a) complexidade na implementação; (b) o modo como a linguagem foi ensinada teve o foco mais científico e menos voltado para aplicações de usuário.

De acordo com os aspectos positivos (b) e (c) a linguagem C, apesar de complexa, obriga a aprendizagem de outras características que não são tão explícitas e comuns em outras linguagens de programação. Para alguns alunos essa característica é uma vantagem, pois formam programadores mais completos e com conhecimentos mais avançados. Em contrapartida, outros alunos enxergam essa característica como uma desvantagem, pois o programador deve se preocupar com outros aspectos além do problema computacional a ser resolvido. Outro ponto interessante nos resultados é que o modo de ensino deve ter um cunho mais voltado para aplicações próximas da realidade do aluno, acredita-se que tal aspecto pode motivar a uma melhor compreensão da importância da disciplina e do assunto ministrado.

A questão Q06 destaca as vantagens e desvantagens da linguagem Java, na visão dos alunos, para o contexto da disciplina. Pelos resultados os aspectos positivos foram: (a) mais simples e abstrata do que o C; (b) baseia-se na programação a orientado a objetos. Já as desvantagens, o aspecto mais recorrente foi a questão de omitir muitos detalhes relacionados ao gerenciamento dos recursos computacionais. Uma observação importante é que um aspecto que foi positivo, o item (a), também foi considerado como negativo para alunos distintos. Alunos com o perfil mais voltado para programação mais avançada acreditam que Java pode formar programadores com poucos conhecimentos nas características dos recursos computacionais. No entanto, C é mais complexo que Java e os programadores C devem se preocupar com diversas particularidades dos recursos computacionais além do problema a ser resolvido pelo algoritmo.

A questão Q07 tem como objetivo verificar, com os discentes, qual paradigma é mais adequado para a implementação dos exercícios propostos na disciplina Estrutura de Dados. Nesta questão, houve um empate em relação a escolha do paradigma a ser considerado para a condução da disciplina. No entanto, houve alguns questionamentos interessantes, diante das respostas obtidas nas duas questões anteriores. Os alunos acham que o paradigma orientado a objetos com Java abstrai as complexidades do gerenciamento de memória. Tal característica simplifica a implementação, mas esconde alguns conhecimentos importantes para a implementação de algoritmos mais eficientes. Além disso, a linguagem Java que é híbrida, esconde a complexidade das iterações com o sistema operacional e a máquina física. Do ponto de vista dos alunos, esses conhecimentos são importantes para formar programadores de nível avançado. No entanto, segundo os alunos essa complexidade da linguagem C acarreta em outras preocupações que não necessariamente estão ligadas ao problema proposto pela disciplina, dificultando a implementação dos algoritmos.

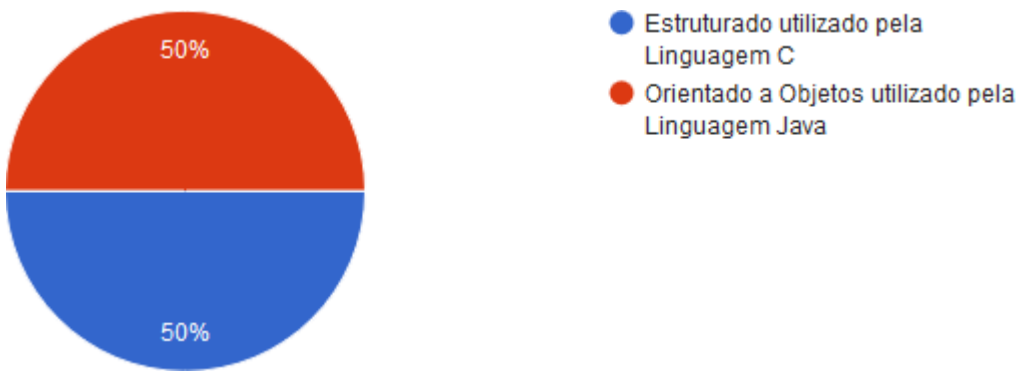

Figura 4. Resultados obtidos na questão Q07

CONSIDERAÇÕES FINAIS

Este artigo apresentou os resultados da experiência obtida na utilização de duas linguagens de programação, de paradigmas de programação distintos, para o ensino intermediário da programação de computadores. Após o término da disciplina, um questionário foi elaborado e submetido aos alunos ativos e assíduos na disciplina. De acordo com os resultados, houve aspectos positivos e negativos em ambas as linguagens dos distintos paradigmas que foram estudados nesta pesquisa. Na visão dos alunos, o paradigma estruturado com a linguagem C fornece uma menor abstração e simplicidade que desperta aspectos complexos para programadores mais avançados. Em contrapartida, a linguagem Java e seu paradigma orientado a objetos favorece uma maior abstração e simplicidade na codificação de algoritmos, mas uma maior abstração pode esconder algumas complexidades que podem ser importantes para a formação de programadores.

Como trabalhos futuros, almeja-se a realização do planejamento da disciplina que foi objeto de estudo desta pesquisa. No que foi exposto por alguns alunos, a metodologia adotada na disciplina não deixou claro a aplicabilidade das estruturas de dados em aplicações do usuário. O foco da disciplina foi na construção das estruturas de dados e não na aplicação destas estruturas. Acredita-se que um foco maior neste aspecto, pode aumentar a motivação dos alunos nos aspectos ensinados na disciplina. Foram relatados aspectos positivos e negativos de ambos os paradigmas e características das linguagens que foram adotadas na disciplina. Pretende-se, em novas edições da disciplina, utilizar a linguagem C e seu paradigma estruturado para a parte da construção das estruturas de dados, pois C e o paradigma estruturado não abstrair aspectos que os alunos acham importantes para a formação de programadores avançados. Já a linguagem Java e seus aspectos de abstração e facilidade de codificação pode ser interessante para focar no uso das estruturas de dados, mostrando a aplicabilidade de cada uma em aplicações de usuário. Neste sentido, a disciplina pode se servir dos aspectos importantes de ambos os paradigmas e linguagens de programação.

REFERÊNCIAS

BARNES, D. J.; KOLLING, M. Programação Orientada a Objetos com Java. 4 ed. São Paulo: Prentice Hall, 2009. 480 p.

FORBELLONE, A. L. V.; EBERSPÄCHER, H. F. Lógica de programação: a construção de algoritmos e estrutura de dados. 3. ed. São Paulo: Prentice Hall, 2005.

EDUARDO, J. P. O. V.; MARINHO, C. S. S.; MOREIRA, L. O. Proposta de Coding Dojo para o Bacharelado em Sistemas e Mídias Digitais da Universidade Federal do Ceará. Revista Sistemas e Mídias Digitais (RSMD); vol. 1, n. 2, 2016.

FISCHER, A. E.; GRODZINSKY, F. The Anatomy of Programming Languages. New Jersey: Prentice Hall, 1993. 557 p.

GIL, A. C. Como elaborar projetos de pesquisa. 4. ed. São Paulo: Atlas, 2007.

GOMES, A.; HENRIQUES, J.; MENDES, A. J. Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. Educação, Formação & Tecnologias; vol. 1(1), p. 93-103, 2008.

GOODRICH, M. T.; TAMASSIA, R. Estruturas de dados e algoritmos em Java. 5. ed. Porto Alegre: Bookman, 2013. 736 p.

MAFRA, S. C. T. Elaboração de check list para desenvolvimento de projetos eficientes de cozinhas a partir de mapas mentais e escala Likert. 1999. 188 f. Tese (Doutorado) - Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 1999.

MARINHO, C. S. S.; MOREIRA, L. O.; COUTINHO, E. F.; PAILLARD, G. A. L.; LIMA NETO, E. T. Experiências no Uso da Metodologia Coding Dojo nas Disciplinas Básicas de Programação de Computadores em um Curso Interdisciplinar do Ensino Superior. II Workshop de Ensino em Pensamento Computacional, Algoritmos e Programação (WAlgProg), 2016, Uberlândia. Anais do V Congresso Brasileiro de Informática na Educação (CBIE), 2016. p. 1097-1106.

PINHEIRO, F. A. C. Elementos de programação em C. Porto Alegre: Bookman, 2012. 548p.

RODRIGUES, R. S. Ensino de algoritmos e linguagem de programação no nível médio: um relato de experiência. 2013. 17 f. Trabalho de Conclusão de Curso (Graduação em Computação) - Centro de Ciências Exatas e Sociais Aplicadas, Universidade Estadual da Paraíba, Patos, 2013.

SEBESTA, R. W. Conceitos de linguagens de programação. 9. ed. Porto Alegre: Bookman, 2011. 792p.

SIANI, S. R.; CORREA, D. A.; LAS CASAS, A. L. Fenomenologia, método fenomenológico e pesquisa empírica: o instigante universo da construção de conhecimento esquadrinhada na experiência de vida. Revista de Administração da UNIMEP; vol. 1, n. 1, 2016.

SILVA, V.; SOUZA, A.; MORAIS, D. Pensamento computacional: um relato de práticas pedagógicas para o ensino de computação em escolas públicas. Revista Tecnologias na Educação; vol. 17, 2016.

Recebido em abril 2017

Aprovado em junho 2017