

Apresentação do Java

Apresentação



Fonte: <https://goo.gl/NMGAFA>

O foco desta disciplina é a compreensão do paradigma orientado a objetos, bem como a sua aplicação prática, ou seja, a resolução dos problemas computacionais se dará por meio de uma linguagem de programação orientada a objetos, no caso de nossa disciplina, o Java.

Nesta aula, você compreenderá de forma geral a tecnologia, os acrônimos utilizados e como interagir com eles. Alguns detalhes não serão esgotados, e nem devem ser, pois o conteúdo da disciplina respeita a hora certa de apresentar e se aprofundar em certos conteúdos. Nada será deixado para trás. Entretanto, somente quando você estiver pronto, alguns conceitos serão minuciosamente tratados.

Logo, é muito importante que você faça o que é proposto em cada aula, sem se preocupar em esgotar todos os detalhes.

No final desta aula, você deve conseguir criar programas com entrada de dados para alguns tipos e gerar saídas para os usuários, além de executar seus programas, interagindo com o Kit de desenvolvimento Java. Pegue uma boa xícara de café e siga em frente!

Desafio

Já passou por aquela avenida que antes tinha aquele lote vazio e em pouco tempo "pluft" aparece uma obra de proporções surpreendentes? A Engenharia Civil é bela, não é mesmo?

A Engenharia Civil é sociedade, é humanidade, é história. Há registros de obras na região da Mesopotâmia com mais de 5 mil anos. Esta importante área escreve a história da humanidade por meio de concreto e tijolos. Os engenheiros criam estruturas que abrigam, protegem, conectam e trazem avanços. Fantástico! Agora... Como a Engenharia Civil consegue tanta eficiência e previsibilidade de tempo e custo?

Esta área é milenar. Consequentemente, a *expertise* está entranhada na sociedade e seus profissionais aprendem cada vez mais com as experiências passadas, o que facilita a evolução de **ferramentas, métodos de construção e gestão**. Afinal, a Engenharia é o campo da ciência que tem por objetivo o aprimoramento e a criação de conhecimentos com utilidade técnica e científica, a partir de embasamentos teóricos de origem na Matemática, Física, Química, entre outras.

Mas o que isso tem a ver com Tecnologia da Informação? Será que você está matriculado em outro curso?

Consegue perceber que a TI atingiu patamares de relevância proporcionais aos da Engenharia Civil? Sem a intenção de criar uma competição, já que cada área tem seu grau de importância, uma coisa é certa, a área de Tecnologia da Informação é um bebê que nasceu em 1943 (idade do primeiro computador ENIAC), frente à milenar Engenharia Civil. Entretanto, a TI vem avançando exponencialmente diante da demanda estratosférica sobre ela e com todas as potencialidades dos meios de comunicação. E não há previsão de limites!

Faça um comparativo entre os jogos de console da década de 1980 (ATARI) e os jogos de realidade virtual, por exemplo. Surreal, não é mesmo? Provavelmente, deve ter feito esta comparação sem saudosismo, sim?

Pois bem, os softwares se materializam por intermédio de uma sequência de passos ditadas por um programador utilizando uma linguagem de programação. Isso você já sabe. Logo, os programadores fazem uso de linguagens para resolver problemas computacionais.

Em meados de 1952, surgiu a primeira linguagem de programação. Entretanto, já na década de 1970, a TI começou a enfrentar crises horrorosas com dificuldades para cumprir prazos dentro de um custo aceitável. O que já sinalizava a necessidade urgente de melhorias nas suas **ferramentas, métodos de construção e gestão**. Ou seja, é a TI passando pelo mesmo caminho da Engenharia Civil.

Usar métodos ou paradigmas que facilitem o desenvolvimento de aplicações é sempre a busca da TI, da mesma forma que a Engenharia Civil. A Programação Orientada a Objetos é uma pequena parte deste contexto na busca de evolução. Já se imaginou construindo sistemas como se fossem pré-moldados? Potencializar a reutilização, facilitando as mudanças? Aplicar padrões entendíveis a toda uma comunidade de desenvolvedores? Observe que há muitos ganhos em termos de agilidade.

Diante destas indagações, reflita sobre as seguintes questões: Quais são os benefícios para uma organização que possua uma TI que consiga resolver seus problemas computacionais de forma ágil e previsível? O que se ganha em fazer a TI acompanhar os anseios das organizações? Como você imagina a evolução da linguagem C como parte desse processo de evolução da TI, em busca de velocidade e previsibilidade de entregas?

Conteúdo

História do Java

Antes de se relacionar com alguém, é importante que você conheça um pouco da história dele, não é verdade? Então, vamos às devidas apresentações. Em 1991, a empresa chamada *Sun Microsystems* foi patrocinadora de um projeto audacioso para a época, chamado *Green*. Este projeto consistia em fazer com que bens de consumo de plataformas completamente diferentes pudessem interagir entre si. Já imaginou o micro-ondas ligar o forno porque tem uma carne descongelando? Pois é. A *Sun* pensou nisso. Para materializar a ideia, foi desenvolvida uma linguagem chamada OAK, que recebeu esse codinome por conta da organização em árvores dos arquivos e um carvalho (sim, uma árvore) que ficava à vista da janela de um engenheiro chamado *James Gosling*, criador da linguagem.

Apesar da admiração de Gosling pela árvore, descobriram que já existia uma linguagem com aquele nome e, consequentemente, outro nome deveria ser definido. O café consumido pela equipe do projeto era oriundo da ilha de Java, fizeram uma brincadeira com o nome e... pegou. Mudaram o nome da linguagem para Java, e é por isso que você já pode se acostumar a ver a imagem de uma xícara de café associado ao nome Java por todos os lugares, inclusive na marca.

Figura 1 – Marca registrada do Java sentido vertical.



Fonte: ([ORACLE](#) , 2017).

Com o passar do tempo, o *Green Project* perdeu força. Em 1993, com o surgimento da World Wide Web (WWW) , a *Sun* viu grande oportunidade em dinamizar o conteúdo, ou seja, além de páginas estáticas, ter sistemas de informações online

acessados por browser . Assim, não perderam mais tempo e realizaram algumas adaptações na linguagem, que foi publicada em 1995. Desde então, o Java é utilizado para desenvolver desde aplicações departamentais até corporativas de grande porte, além de aplicativos móveis e locais.

As características principais desta linguagem são:

- Baseada no paradigma orientado a objetos.
- Portabilidade de plataformas ou multiplataforma (escreva uma vez, execute em qualquer lugar).
- Extensas bibliotecas de comunicação TCP/IP e interação com protocolos como o HTTP e o FTP .
- Sintaxe similar ao C, C++ e C#.
- Alto nível da programação paralela e distribuída.
- Desalocação de memória automática.

Em 2009, a *Sun* foi adquirida pela *Oracle*, que agora é a responsável por manter o Java atualizado e em constante evolução para atender às novas tendências.

Tecnologia Java

O Java é subdivido em três grandes partes, no que se refere à aplicabilidade. São elas:

- **Java Standard Edition (JSE)** – é a base da tecnologia. Inclui o ambiente de execução e as bibliotecas comuns. Aplicável para desenvolvimento de aplicações *desktops* e de base para todas as demais finalidades descritas a seguir.
- **Java Enterprise Edition (JEE)** – a edição voltada para o desenvolvimento de aplicações corporativas e para internet.
- **Java Micro Edition (JME)** – a edição para o desenvolvimento de aplicações para dispositivos móveis e embarcados.

Importante

Os conteúdos abordados na disciplina estão completamente inseridos no JSE.

Plataforma Java

Muitos desenvolvedores que iniciam no Java assustam-se com a quantidade de sopa de letrinhas que sempre aparecem nos termos ou nos textos. Até agora, você conheceu as três principais abreviações que tratam da destinação das aplicações: JSE, JEE e JME. Outras serão apresentadas a seguir.

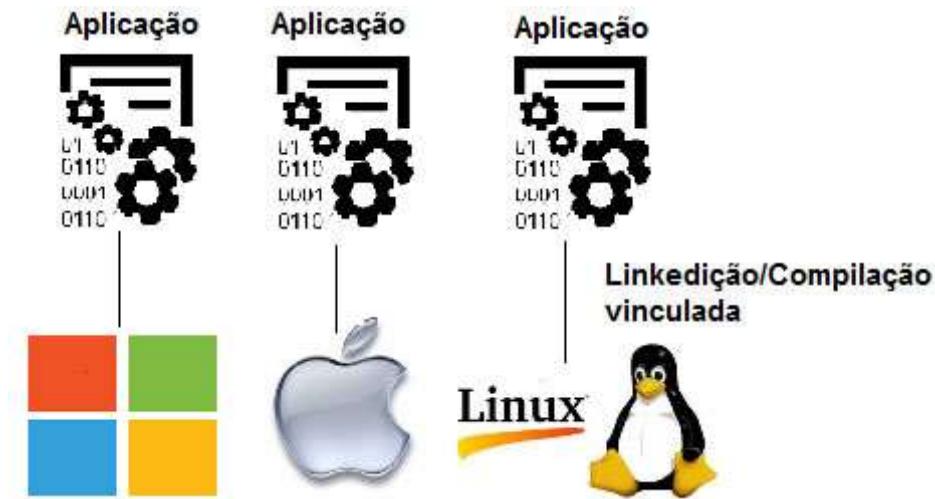
Uma das características do Java que o colocou em posição de vantagem em relação a muitas outras linguagens é a multiplataforma.

Diferentes sistemas operacionais existem no mercado, sendo os mais conhecidos MAC OS, Linux e Windows. Segundo [Silberschatz et al](#) (2008), o papel de um Sistema Operacional (SO) basicamente é reduzir a complexidade de interação com hardware, concorrência, integridade, etc. Lidar com esses temas, computacionalmente falando, é relativamente complexo. A programação de Sistemas Operacionais é peculiar. Portanto, quando encontrar com um desenvolvedor de Sistemas Operacionais, tire uma *selfie* com ele e pague um café.

Os programas que são executados em uma máquina também são clientes dos Sistemas Operacionais, e o Java consegue ser executado em qualquer SO, sendo compilado apenas uma vez, diferentemente de outras linguagens.

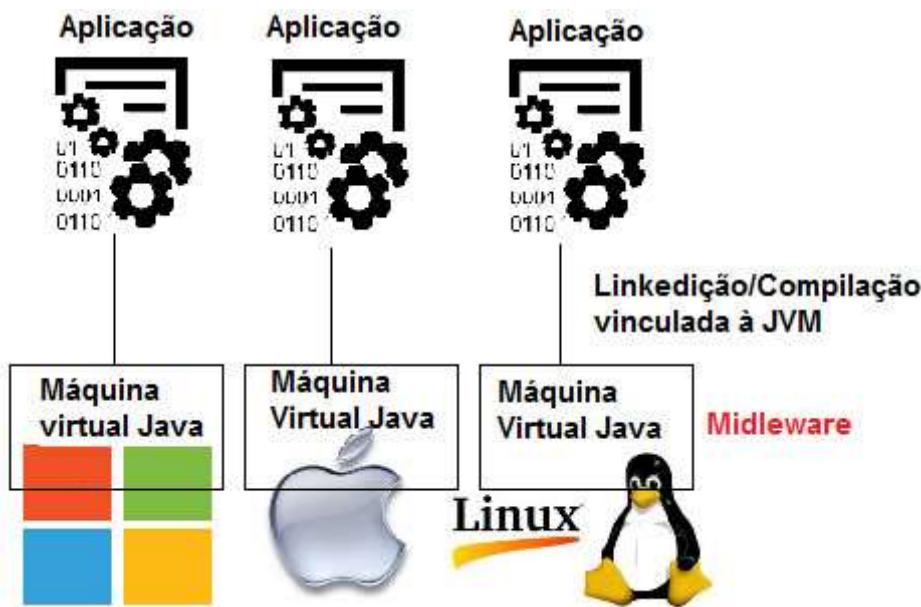
Quando um programa é compilado, ele fica vinculado ao SO na qual a atividade de compilação e de linkedição aconteceu. O código binário resultante da compilação deve interagir com o SO para que seja executado. Como levar uma compilação feita no Windows para uma máquina Linux? Complexo, pois o processo é vinculante. Além disso, a implementação em si pode ser destinada a um SO específico também. A título de exemplo, implementar programação paralela em C em sistema operacional Linux é diferente do que em sistema operacional Windows. Observe a Figura 2.

Figura 2 - Linkedição e compilação vinculada aos sistemas operacionais.



O Java conseguiu se desvincular disso. Mas como? Aplicações Java independem do Sistema Operacional e tudo isso é possível devido à forma como a plataforma foi estabelecida. O processo de compilação tem foco em máquina virtual, e não no sistema operacional instalado. Observe a figura 3.

Figura 3 - Linkedição e compilação vinculada à máquina virtual.



Para cada SO, existirá uma Máquina Virtual Java (JVM) que a aplicação não terá interação direta com o Sistema Operacional, mas sim com a máquina virtual. Dessa forma, a JVM fará o papel de middleware entre as aplicações Java e o Sistema Operacional.

Kit de Desenvolvimento e Execução

Dando continuidade à sopa de letras existentes no Java, temos o *Java Development Kit* (JDK). Ele é um kit de desenvolvimento de aplicações em Java e é composto de várias partes para a construção. Os principais a serem levados em consideração são: o compilador Java, o ambiente de execução (máquina virtual) e o documentador.

O *download* do JDK deve ser feito no site da ORACLE, e deve ser compatível com o seu sistema operacional e a arquitetura (32 ou 64 bits).

O *Java Runtime Environment* (JRE) é o ambiente de execução Java. A independência das aplicações Java dos sistemas operacionais é possível por conta deste ambiente. As aplicações Java são executadas por ele, independentemente do Sistema Operacional em que elas se encontram.

Um cliente que receberá uma aplicação desenvolvida por você deverá ter instalado na sua máquina o JRE. Já, você, que a construiu, deverá ter o JDK.

Saiba que, dentro do JDK, já existe o ambiente de execução (JRE); afinal, espera-se que você teste as suas aplicações. Logo, você, como desenvolvedor, não precisa baixar o JRE.

Importante

Faça a instalação do JDK, seguindo os tutoriais a seguir, de acordo com o seu sistema operacional. A versão atual do Java é a 8.

- [Linux - Instalando o JDK e as variáveis de ambiente](#) ↗
- [Windows - Instalando o JDK e as variáveis de ambiente](#) ↗
- [MAC OS - Instalando o JDK e as variáveis de ambiente](#) ↗
- [Download do JDK atualizado](#) ↗

Primeiro Programa

Agora que você instalou o JDK de acordo com o Sistema Operacional, conforme os tutoriais indicados anteriormente, e ainda configurou as variáveis de ambiente corretamente, você deve estar ansioso para ver um programa em Java sendo executado. Já no primeiro momento, você perceberá a similaridade da sintaxe do Java e do C, que é uma linguagem com a qual você já teve contato.

Observe a Tabela 1 a seguir e compare os códigos de um programa que inicializa duas variáveis flutuantes, calcula a média e apresenta:

Tabela 1 – Exemplo de codificação em C e em Java

Programa em C: Programa.c	Programa em Java: Programa.java
---------------------------	---------------------------------

Programa em C: Programa.c	Programa em Java: Programa.java
<pre>#include <stdlib.h> #include <stdio.h> #include <conio.h> int main() { //inicio double nota1, nota2, media; nota1 = 10; nota2 = 7; media = (nota1 + nota2) / 2; printf("\n A media eh: %f\n", media); //fim system("pause"); return(0); }</pre>	<pre>public class Programa{ public static void main (String args[]){ //inicio double nota1, nota2, media; nota1 = 10; nota2 = 7; media = (nota1 + nota2) / 2; System.out.printf("\n A media eh: %f \n ", media); //fim } }</pre>

É perceptível a semelhança entre a sintaxe das duas linguagens. Não é mesmo? Observe que a codificação entre os comentários se torna diferente somente na saída de dados, que foi acrescentado o `System.out` ao `printf`. Mais confortável agora? Certo. Mas lembre-se de que ainda não foi discorrido sobre a orientação a objetos.

Para executar o programa, você deve utilizar o JDK, que você já instalou em sua máquina. Acompanhe a sequência de passos a seguir:

1. Salve o código java acima como um arquivo `.java` em uma pasta a sua escolha com o nome `Programa.java`.
2. Abra o CMD/Terminal e navegue até a pasta onde está o arquivo.
3. execute: `javac Programa.java`
4. execute: `java Programa`
5. A saída será: *A media eh: 8,5*

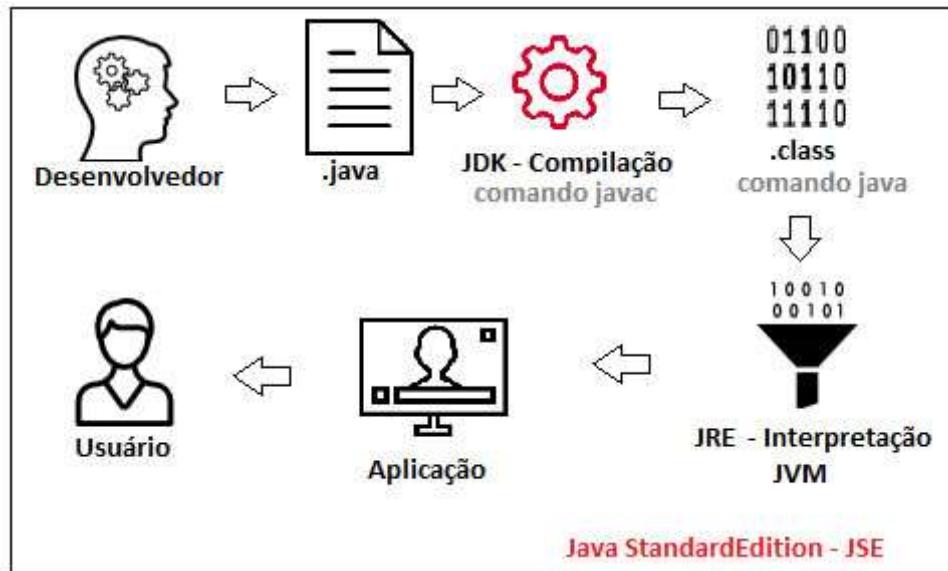
No passo 3, o compilador Java foi invocado para compilar o seu programa.

No passo 4, o ambiente de execução foi acionado para executar o seu programa.

Fácil, sim?

Quando você informou o comando do passo 3 (javac), para o arquivo **.java**, foi realizada a compilação da aplicação, e o resultado da compilação foi um arquivo **.class**, conhecido como **bytecode** destinado à máquina virtual; no passo 4, a JVM foi invocada para interpretar o arquivo gerado, e o resultado dessa interpretação é a aplicação executando. Observe a Figura 4, que esquematiza esse processo.

Figura 4 – Esquema do processo de compilação e interpretação do Java.



Para Refletir ⚡

Um desenvolvedor iniciante no Java acabou de desenvolver um programa para um grande amigo de cálculo do IMC. O desenvolvedor enviou os arquivos **.java** e o cliente instalou o ambiente de execução. Dentro deste cenário, existe alguma inconsistência? Caso sim, qual seria e por quê?

Lembre-se! O Java consegue ser multiplataforma por conta da máquina virtual Java que existe para Sistema Operacional. Para execução, é necessário o Ambiente de execução. Esse ambiente necessita de arquivos compilados em linguagem de máquina (bytecode ou **.class**) e não código fonte (**.java**).

Entrada e Saída de Dados

Para que você consiga desenvolver seus programas, precisa interagir com o usuário. Certo? Esta interação significa receber do usuário alguns valores, para que você, por meio da programação, faça algum processamento e fatalmente apresente uma saída. Você deve se lembrar disso nas disciplinas iniciais de programação: entrada, processamento e saída, não é verdade?

O Java é uma linguagem com diversas bibliotecas disponíveis para entrada de dados. O conteúdo da aula tem um propósito de respeitar tudo aquilo que você já aprendeu e teve esforço despendido. Atualmente, você está bem familiarizado com a entrada de dados via console em C. Então, é justamente este método que será utilizado. Mas, lembre-se do foco do conteúdo da aula, mantenha o foco no que é apresentado, siga as instruções e não se preocupe com detalhes que não acrescentarão neste momento. Por exemplo: o que seria a palavra **new**? Neste momento do curso, pouco importa.

A seguir, na Tabela 2, é apresentado um programa que faz leitura de dois tipos de dados que você conhece: int e double.

Tabela 2 – Leitura de dados em Java.

Programa em Java: Leitura.java

```
import java.util.Scanner; //Lembra do #include ? Faça esta associação por
enquanto para conseguir ler.
public class Leitura{
    public static void main (String args[]){
//inicio
    double salario;
    System.out.print("Informe um salário: ");
    salario = new Scanner(System.in).nextDouble(); // Comando grande, né? É só
    isso para ler um double
    int idade;
    System.out.print("Informe uma idade: ");
    idade = new Scanner(System.in).nextInt(); // só isso para ler um int
    System.out.printf("\n O salário informado foi : %f", salario);
    System.out.print("\n A idade informada foi :" +idade);
//fim
    }
}
```

Bem, após você executar este programa, seguindo os mesmos passos do programa anterior, observando que agora o nome desse arquivo é **Leitura.java**, você perceberá que foi possível inserir os valores a sua escolha e os valores foram apresentados.

Importante

O Java é *case sensitive*, ou seja, muito cuidado ao escrever os comandos, pois System é completamente diferente de system.

Para Refletir

Em todas as linguagens de programação, o operador '+' está presente. Este operador é utilizado para realizar uma operação matemática que é a soma, não é verdade? Entretanto, esse operador no programa **Leitura.java** aparece de uma forma bem diferente daquela que você conhecia. O que esse operador está fazendo nesse contexto? O que esse operador faz quando está próximo de uma String?

Você deve ter observado que no contexto em questão, o operador '+' faz um papel de concatenação do valor da idade com o texto anterior. Há uma promoção da idade para String antes e, na sequência, uma concatenação entre as duas Strings (tudo isso implicitamente). Mais detalhes sobre String você conecerá ao longo do estudo da disciplina.

Na Prática

"Prezado(a) estudante,

Esta seção é composta por atividades que objetivam consolidar a sua aprendizagem quanto aos conteúdos estudados e discutidos. **Caso alguma dessas atividades seja avaliativa, seu (sua) professor (a) indicará no Plano de Ensino e lhe orientará quanto aos critérios e formas de apresentação e de envio.**"

Bom Trabalho!

Atividade 01

^

O Java possui uma importante característica, que é ser independente de plataforma.
Explique, com suas palavras, o que é essa característica e como o Java a fornece.

Atividade 02

^

Fazendo uso de um editor de texto básico, como o bloco de notas, crie um arquivo .java chamado **ExercicioDois.java**, que deverá resolver o seguinte problema:

Ler uma temperatura em graus Celsius e apresentá-la convertida em graus Fahrenheit. A fórmula de conversão é: $\text{FAR} = (9 * \text{CEL} + 160) / 5$, sendo FAR a temperatura em Fahrenheit e CEL em Celsius. Não se esqueça de assistir ao [vídeo 1](#) antes.

Finalizando... C

Nesta aula, você conheceu a história do surgimento do Java e suas características. Soube como o Java conseguiu implementar uma das suas principais características, que é o fato de ser multiplataforma.

Você também percebeu o quanto a sintaxe do Java é parecida com a que você já conhece. Afinal, James Gosling é um programador C.

Você foi direcionado a links importantes de instalação e configuração do seu ambiente. Não deixe de fazer as instalações corretamente, afinal, você precisa praticar.

Para te apoiar, assista ao vídeo 1 . Ele traz algumas dicas e um panorama da aula 1 para te deixar mais confiante.

Vídeo 1

Prática Profissional - Programação orientada à objeto - U...



Agora que você já assistiu ao vídeo, já está pronto para exercitar, utilizando o que já sabe de C. Acesse o item “Na prática” e faça TODOS os exercícios desta aula.

Atividade 03



Fazendo uso de um editor de texto básico, como o bloco de notas, crie um arquivo .java chamado **ExercicioTres.java**, que deverá resolver o seguinte problema:

Receber do usuário duas notas e realizar a média aritmética desses valores. Como saída, apresentar as notas informadas, a média e se o estudante alcançou aprovação ou não. Não se esqueça de assistir ao [vídeo desta aula](#)  antes.

Saiba Mais

Para ampliar seu conhecimento a respeito desse assunto, veja abaixo a(s) sugestão(ões) do professor:

- Para ampliar seus conhecimentos sobre essa parte inicial do Java, acesse o [site do Java](#).
- O Java possui um acelerador para a interpretação (execução) do código binário chamado Just IN TIME – JIT. Faça uma leitura do seguinte [artigo da IBM sobre esse compilador](#).

Referências

- BOOCH, G. **Object-Oriented Design with Applications**, Benjamin-Cummings, 1991.
- BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML – guia do usuário**. 2. ed. Rio de Janeiro: Campus. 2006.
- CARDELLI, L.; WEGNER, P. **On Understanding Types, Data Abstraction, and Polymorphism**. ACM Computing Surveys (CSUR). vol. 17, pp. 471-523. 1985.
- DEITEL H. M.; DEITEL, P. J. **Java, como programar**. 6. ed. Porto Alegre: Bookman. 2006.
- DICIONÁRIO AURELIO. 2017. Disponível em:
<https://contas.tcu.gov.br/dicionario/home.asp> Acesso em: 19 mar. 2017.
- ERICH, G. et al. **Padrões de projeto**: soluções reutilizáveis de software rientado a objetos. Porto Alegre: Bookman. 2000.
- HORSTMANN, C. S.; CORNELL, G. **Corejava 2 – Volume I – Fundamentals**. São Paulo: Makron Books. 2010.
- NEWRELIC. **The Most Popular Programming Languages of 2016**. 2016. Disponível em:
<https://blog.newrelic.com/2016/08/18/popular-programming-languages-2016-go>. Acesso em: 9 Mar 2017.
- NIEMEYER, P.; KNUDSEN, J. **Aprendendo Java**. Rio de Janeiro: Campus. 2000.
- ORACLE. **Java Licensing Logo**. 2017. Disponível em:
<http://www.oracle.com/us/technologies/java/java-licensing-logo-guidelines-1908204.pdf>. Acesso em: 13 mar. 2017.
- SIERRA, K.; BATES, B. **Certificação Sun para Programador JAVA 5 Guia de Estudo**. Rio de Janeiro: Alta Books, 2006.
- SILBERSCHATZ, A; GALVIN, P. B.; GAGNE, G. **Sistemas operacionais com Java**. 6. ed. Rio de Janeiro. Editora Campus, 2004.

- STUCKEY, P. J.; SULZMANN, M. **A theory of overloading**. International Conference on Functional Programming. Proceedings of the seventh ACM SIGPLAN international conference on Functional programming. Pittsburgh, PA, USA, 2002. pp. 167-178.
- WIKIPÉDIA. Desenvolvido pela Wikimedia Foundation. Conteúdo sobre o **Ambiente de Desenvolvimento Integrado**. 2017. Disponível em: <https://pt.wikipedia.org/wiki/Ambiente_de_desenvolvimento_integrado>. Acesso em: 5 mar. 2017.