

Programação Orientada a Objetos - POOS3

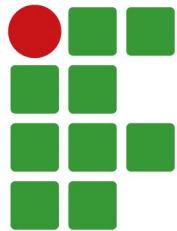
1

Tecnologia em Análise e Desenvolvimento de Sistemas

Aula 1

Apresentação da disciplina, Avaliação
Diagnóstica e Introdução ao Java

2º semestre de 2018



INSTITUTO FEDERAL

São Paulo

Câmpus Araraquara

Conteúdo

- Sobre a disciplina
 - Ementa
 - Conteúdo Programático
 - Bibliografia
 - Avaliação
 - Ferramentas
- Avaliação Diagnóstica
- Introdução a Plataforma Java
- Programação em Java

Sobre a disciplina

- **Números**

- 66,7 horas
- 80 aulas no semestre
- 4 aulas semanais

- **Ementa**

- Desenvolvimento de sistemas de software baseados no paradigma orientado a objetos.

- **Objetivos**

- Tornar o aluno apto a entender e aplicar os conceitos de orientação a objetos no desenvolvimento de sistemas.

- **Conteúdo Programático**

- Fundamentos da orientação a objetos. Aplicação dos conceitos de orientação a objetos.

Sobre a disciplina

• BIBLIOGRAFIA BÁSICA:

- DEITEL, P. J.; DEITEL, H. M. Java: como programar. 8. ed. São Paulo: Pearson Education do Brasil, 2010. 1144 p.
- SIERRA, K. Use a cabeça!: Java. 2. ed. Rio de Janeiro: Alta Books, 2010. 484 p. (Use a cabeça!).
- STELLMAN, A.; GREENE, J. Use a cabeça!: C#. 2. ed. Rio de Janeiro: Alta Books, 2011. 797 p. (Use a cabeça!).

• BIBLIOGRAFIA COMPLEMENTAR:

- DEITEL, H. M. et al. C#: como programar. São Paulo: Pearson Education do Brasil, 2003. 1153 p.
- LARMAN, C. Utilizando UML e padrões: uma introdução à análise e ao projeto orientados a objetos e ao desenvolvimento iterativo. 3. ed. Porto Alegre: Bookman, 2007. 695 p.
- MENDES, D. Rocha. Programação Java com ênfase em orientação a objetos. São Paulo: Novatec, 2009. 463 p.
- SANTOS, R. Introdução à programação orientada a objetos usando Java. 2 ed. Rio de Janeiro: Elsevier, 2013. 313 p.
- SHARP, J. Microsoft Visual C# 2010: Passo a passo. Porto Alegre: Bookman, 2011. 775 p.

• OUTRAS

- WINDER, R; ROBERTS, G. Desenvolvendo software em java. 3. ed. Rio de Janeiro: LTC, 2009. xxii, 696 p.

Sobre a disciplina

- **Material**

- https://github.com/ednilsonrossi/POOS3_2s2018

- **Grupo de discussão**



- Foi criado um grupo no Whatsapp para que os alunos postem dúvidas durante os intervalos de aula.
 - <https://chat.whatsapp.com/6NWaE4igc202YDa8tckynp>

Sobre a disciplina

- Critério de avaliação

Prova individual – P1 (20%)
Prova individual – P2 (20%)
Laboratórios Práticos – LP (10%)
Exercícios avaliativos – EX (30%)
Projeto Final – PF (20%)

Os laboratórios são estudos dirigidos desenvolvidos pelo aluno em laboratório de informática. Ao final da aula o laboratório proposto deverá estar operante.

Serão aplicados N exercícios avaliativos sendo a nota EX obtida pela média aritmética de todos os exercícios.

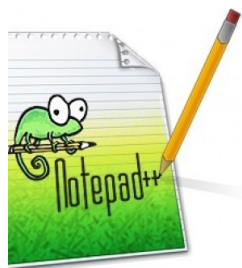
O projeto será elaborado durante o semestre e a no PF será obtida pelas entregas parciais considerando os pesos que serão definidos para cada parte do PF.

Serão aplicados exercícios de fixação para acompanhar o aprendizado, é importante que sejam realizados.



Sobre a disciplina

- Ferramentas
 - Java Oracle 8
 - Notepad++ / Geany / Outros
 - Eclipse (Mars) [Eclipse IDE for Java Developers]



Notepad++



Geany



Avaliação Diagnóstica

- Exercícios para apontar as competências dos alunos da turma.
- Exercícios individuais.
- Implementar os exercícios em linguagem C ou Java.

Exercício 1

- Implemente uma função recursiva que verifica se um número, denominado chave, existe ou não no vetor.
- A função retorna a posição do vetor se a chave existir e -1 se a chave não existe no vetor.
- Os argumentos da função são: um vetor de inteiros, o tamanho do vetor e o valor da chave.
- Para testar a função, implemente um programa que gere um vetor de 10^5 com valores aleatórios e verifique, usando a função, se o chave gerada na posição zero do vetor se repete no vetor. Caso a chave exista deve-se imprimir qual a posição, caso contrário uma mensagem informando que a chave não foi encontrada.

Exercício 2

- Considere um estacionamento no centro da cidade. O estacionamento foi implementado em um terreno estreito que permite apenas que os carros sejam estacionados em fila, ou seja, um atrás do outro. Além disso, o estacionamento possui acesso a duas ruas, ou seja, é possível atravessar de uma rua a outra por dentro do estacionamento.
- Assim, o proprietário definiu que a entrada do estacionamento se dá por uma rua e a saída por outra.
- Implemente as funções que faltam no programa a seguir de forma a permitir o funcionamento deste estacionamento.

Exercício 2

```
#include <stdio.h>
#include <stdlib.h>
#define NROVAGAS 30

typedef struct estacionamento Estacionamento;
struct estacionamento{
    int vetor[NROVAGAS];
    int qtdade;
};

Estacionamento criaEstacionamento(){
    Estacionamento variavel;
    variavel.qtdade = -1;
    return variavel;
}

int estaCheio(Estacionamento variavel){
    return variavel.qtdade == NROVAGAS-1?1:0;
}

void mostraEstacionamento(Estacionamento garage){
    int i;
    printf("\nCarros estacionados: ");
    if(! (garage.qtdade < 0) ){
        for(i=0; i<garage.qtdade;i++){
            printf("%d - ", garage.vetor[i]);
        }
        printf("%d", garage.vetor[i]);
    }
}
```

```
Estacionamento entraCarro(Estacionamento garage, int
placa){ }

Estacionamento saiCarro(Estacionamento garage){ }

int main()
{
    Estacionamento garagem;
    garagem = criaEstacionamento();
    garagem = saiCarro(garagem);
    garagem = entraCarro(garagem, 1234);
    garagem = entraCarro(garagem, 2345);
    garagem = entraCarro(garagem, 3456);
    garagem = entraCarro(garagem, 4567);
    garagem = saiCarro(garagem);
    garagem = entraCarro(garagem, 9876);
    garagem = entraCarro(garagem, 8765);
    garagem = saiCarro(garagem);
    garagem = saiCarro(garagem);
    Return 0;
}
```



Introdução a Plataforma Java



Esclarecimento

A disciplina é sobre Programação Orientada a Objetos e utilizaremos Java como uma ferramenta na disciplina.

O que é Java?

- É uma linguagem de programação de alto nível cuja principal característica é ser **orientada a objetos**.
 - Lançada pela Sun Microsystems em 1995.
 - Esta linguagem de programação é compilada para uma linguagem intermediária denominada **Java Bytecode**, que é constituída por instruções que podem ser executadas pela Plataforma Java.



O que é Java?

- Então:
 - Linguagem de Programação
 - Plataforma de Desenvolvimento

Como surgiu o Java?

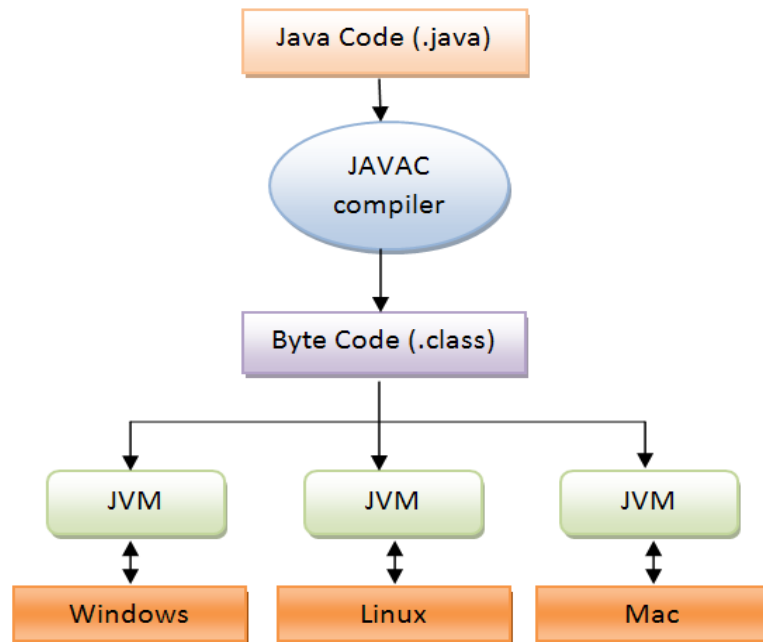
- Em 1991 era grande o interesse da SUN no mercado de dispositivos eletrônicos inteligentes destinados ao consumidor final.
- Projeto Green, que resultou na criação de uma linguagem baseada em C/C++ chamada Oak (carvalho), nome de uma árvore da frente da janela do escritório da SUN.
- Mais tarde descobriu-se que existia uma linguagem de mesmo nome, quando a equipe da SUN visitou uma cafeteria local chamada JAVA, nome do café importado.
- O projeto Green atravessava dificuldades. Por sorte, em 1993 estoura a popularidade da Internet e os pesquisadores da SUN visualizam o potencial do JAVA para Web.



Write once run anywhere.

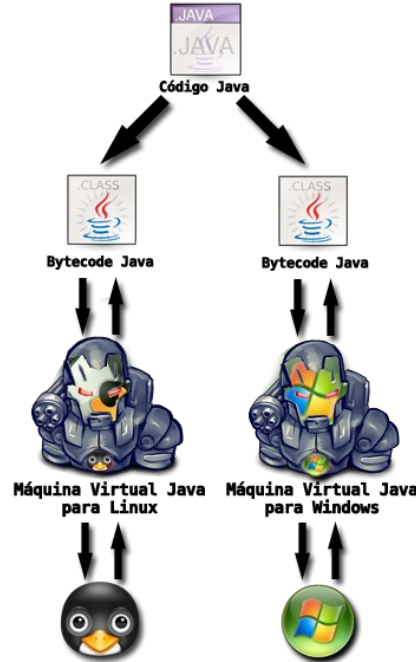
O que isso significa?

- Escreva uma vez e execute em qualquer lugar!
 - Pela característica da plataforma Java, o mesmo código pode ser executado em qualquer hardware que possua uma JVM (Java Virtual Machine – Máquina Virtual Java) implementada.



Portabilidade

- “Conjunto de atributos que evidenciam a capacidade do software de ser transferido de um ambiente para outro”



O que garante a portabilidade do Java?

- **A Java Virtual Machine (JVM);**
 - Pode-se afirmar que para qualquer dispositivo com uma JVM implementada, os programas Java são executados.



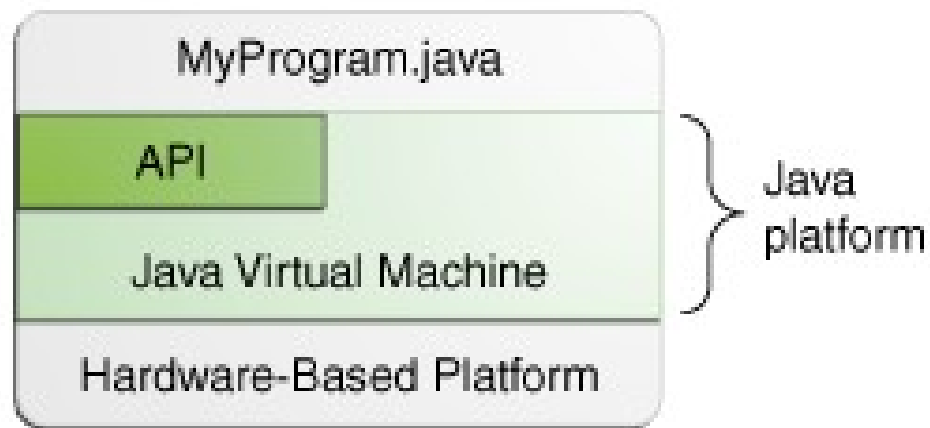
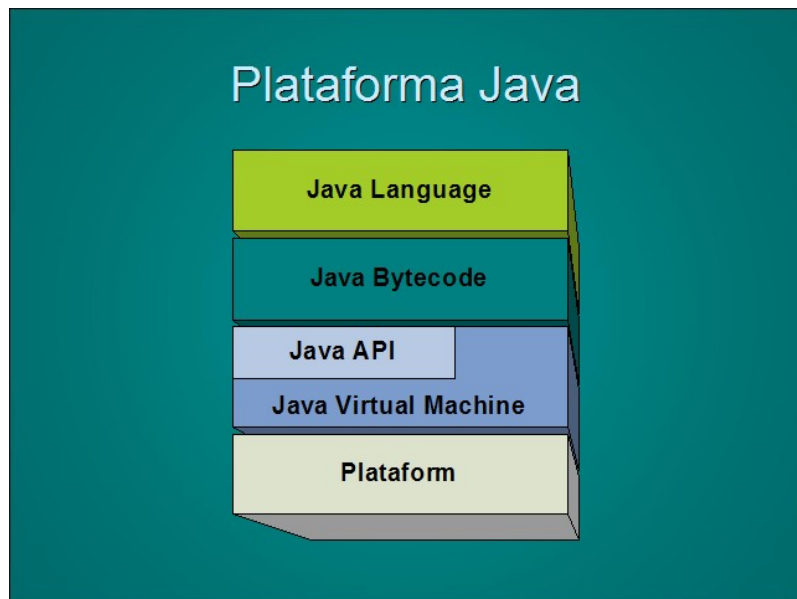
O que se entende por Plataforma?

- Existem vários sistemas operacionais (Linux, MS Windows, MacOS, Solaris, Android, etc) que podem ser executados em hardwares diferentes.
- Uma plataforma é o ambiente de hardware e/ou software onde uma aplicação é executada.
- Normalmente, a plataforma é descrita pela combinação de hardware e sistema operacional.



Plataforma Java

- A plataforma Java é a plataforma onde os Java *bytecodes* são executados.
- Ela é diferente das plataformas usuais pelo fato de ser composta somente de software, que é executado em outra plataforma.



Plataforma Java

- Significa que a plataforma Java, na verdade, interpreta os Java *bytecodes*.
- Cada instrução na forma de *bytecode* deve ser analisada pela JVM e interpretada por ela. Uma vez que isso ocorre, a JVM envia um comando para o hardware executar a instrução de fato.
- A plataforma Java foi desenvolvida para ser segura, possibilitar manipulação de exceções e coleta de lixo (*garbage collection*), que libera, automaticamente, a memória alocada dinamicamente.

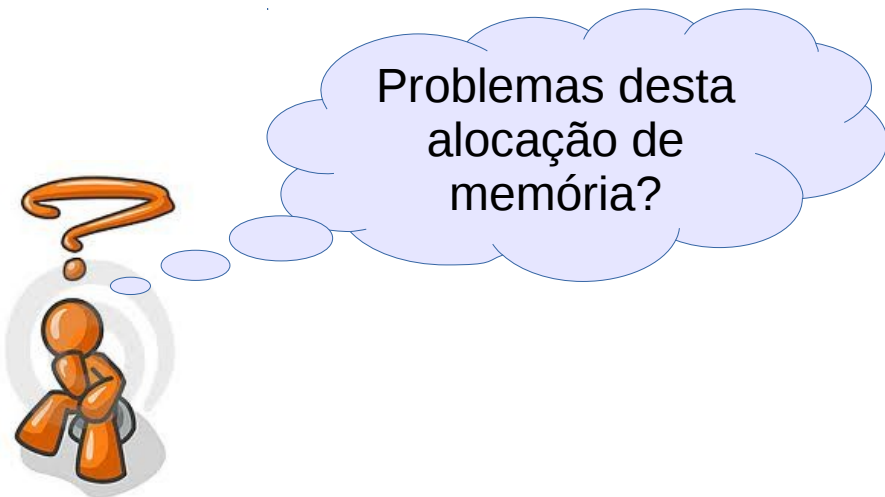
Plataforma Java

- A plataforma Java é composta de duas partes:
 - **Máquina Virtual Java (JVM):** tem como objetivo executar os *bytecodes*, traduzindo-os para o código nativo. Ela pode ser implementada em hardware (processadores dedicados) ou software (JDK);
 - **Interface para Programação de Aplicações (API):** oferece um conjunto de pacotes de classes (*packages*) com funcionalidades semelhantes às bibliotecas de C/C++.

Plataforma Java

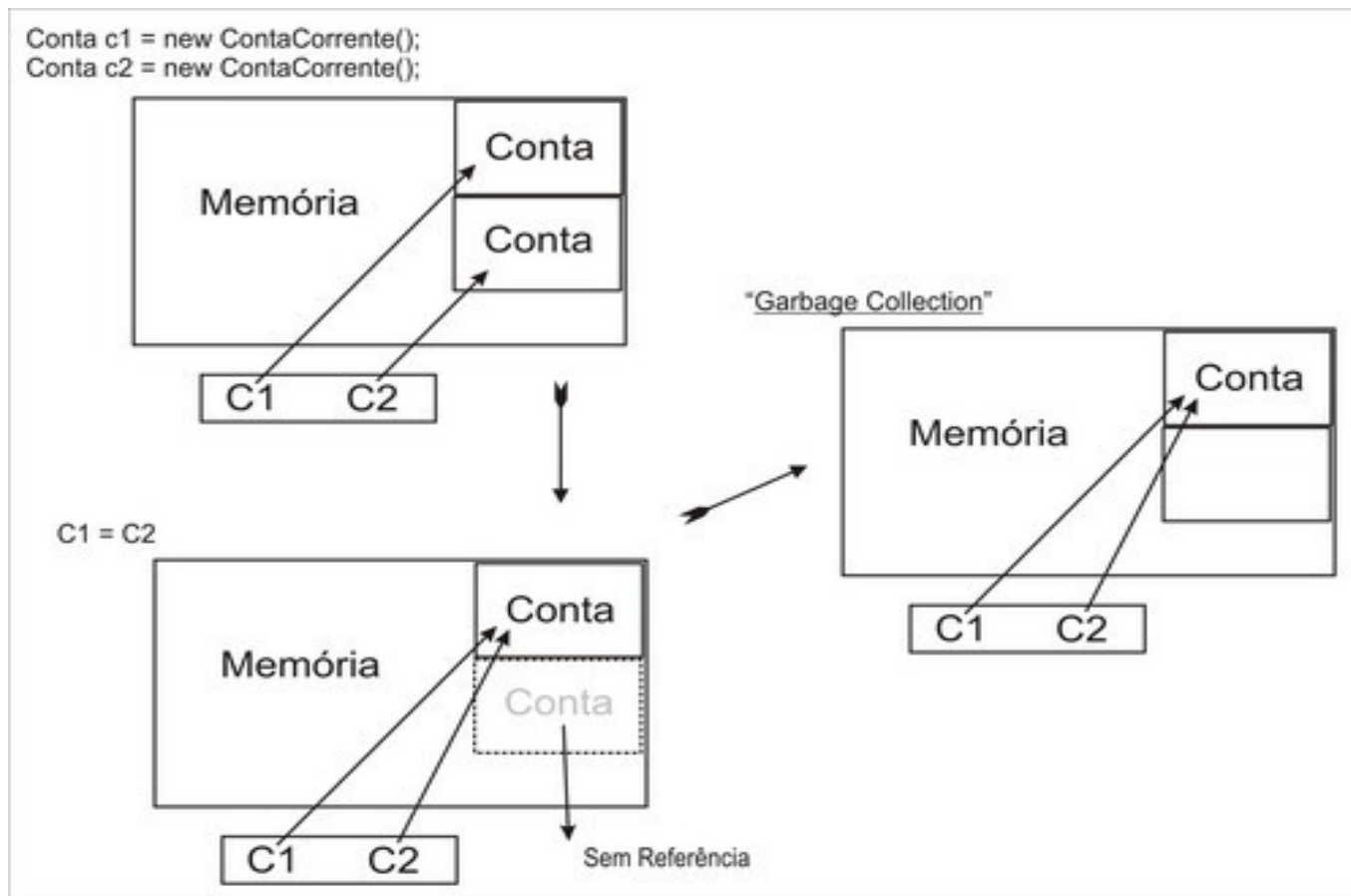
Coletor de Lixo

- A maior parte das linguagens possuem alguma instrução que permite ao programador requisitar memória dinamicamente, ou seja, durante a execução do programa.
- Da mesma forma, existe um comando que permite ao programador liberar essa memória quando ela não for mais utilizada.



Plataforma Java

Coletor de Lixo






Plataforma Java

Coletor de Lixo

26

- O Coletor de Lixo faz parte da **Plataforma Java** e não da linguagem Java.
 - Errado comparar este aspecto entre Java e C++.

Oracle

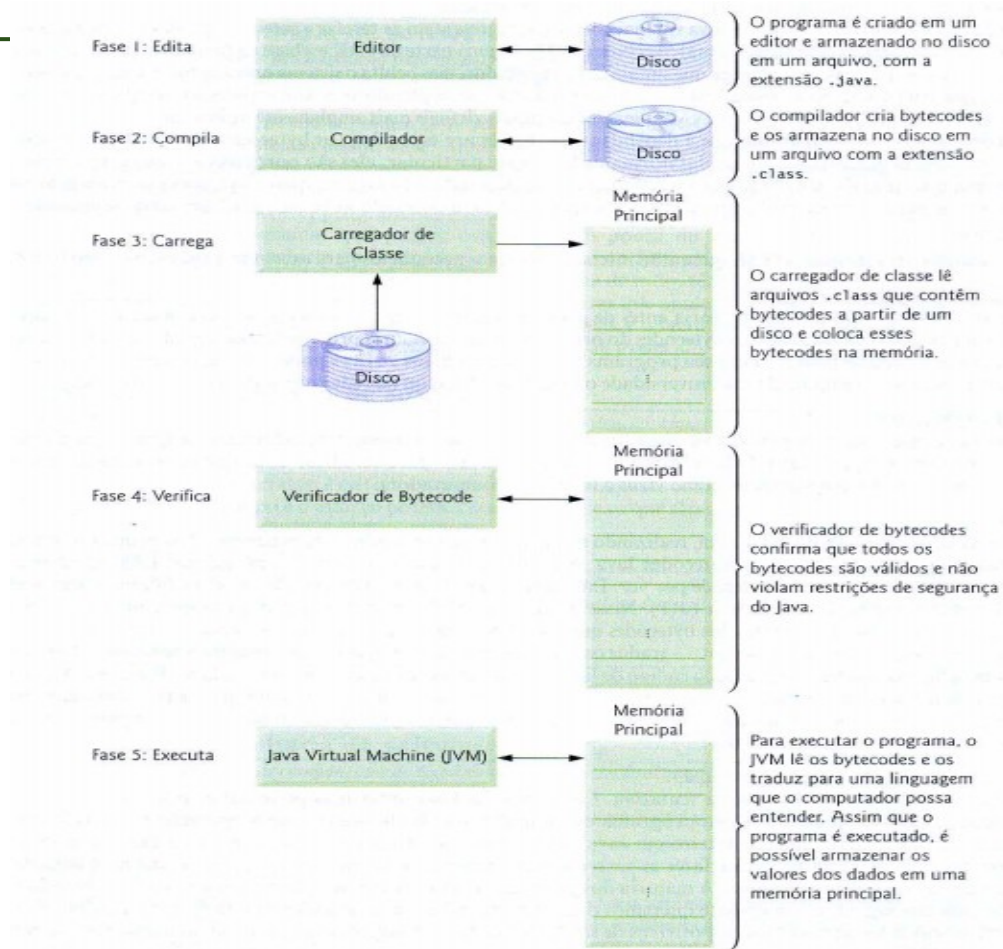
- Oracle anuncia compra da Sun por mais de US\$ 7 bilhões 



Programação em Java



Ambiente de Desenvolvimento Java



Exemplo 1

- Digitar o texto abaixo em qualquer editor de texto:

```
public class Exemplo1{  
    public static void main(String args[ ]){  
        System.out.println("Lucca, Java é melhor que Python...");  
    }  
}
```

- Salvar o arquivo como Exemplo1.java
- Compilar: **javac Exemplo1.java**
- Executar: **java Exemplo1**

Tipos de Dados

TIPO	FAIXA DE VALORES	TAMANHO
byte	-128 a 127	1 byte
char	0 a 65.535	2 bytes
short	-32.768 a 32.767	2 bytes
int	-2.147.483.648 a 1.147.483.648	4 bytes
long	-9.223.372.036.854.775.808 a 9.223.372.036.854.775.807	8 bytes
float	$-3,4 \times 10^{-38}$ a $3,4 \times 10^{38}$	4 bytes
double	$-1,7 \times 10^{-308}$ a $1,7 \times 10^{308}$	8 bytes

Saída de Dados

- `System.out.print();`
- `System.out.println();`
- `System.out.printf();`

Entrada de Dados: Classe Scanner

```
import java.util.Scanner;

public class Exemplo2{

    public static void main(String args[]){
        //Instanciar um objeto para leitura
        Scanner input = new Scanner(System.in);

        int ano;

        System.out.printf("Ano de nascimento: ");
        ano = input.nextInt();

        System.out.printf("Idade: %d \n", 2018-ano);
    }
}
```

Entrada de Dados: Classe Scanner

- Principais métodos de leitura:
 - `nextInt();`
 - `nextDouble();`
 - `nextFloat();`
 - `nextLine();`

Entrada de Dados: Classe Scanner

35

```
import java.util.Scanner;

public class Exemplo3{

    public static void main(String args[]){
        Scanner input = new Scanner(System.in);
        char sexo;
        System.out.printf("Informe sexo (M, F ou I): ");
        sexo = input.nextLine().charAt(0);
        System.out.printf("Sexo digitado: %c \n", sexo);
    }
}
```

Estrutura Condicional

```
if( <condição> )
```

```
{
```

```
}
```

```
else
```

```
{
```

```
}
```

Estrutura de Seleção

```
switch( <variável>){  
    case valor: ...;  
    default ...;  
}
```

Estruturas de Repetição

- `while(<condição de parada>){ }`
- `do ... while (<condição de parada>);`
- `for(<inícios>;<condição>; <incrementos>){ }`

Trabalhando



- **Exercícios Avaliativos**
 - 1
- **Pesquisa**
 - Pesquise sobre portabilidade de software (NBR 9126) e reuso.
- **Leitura Obrigatória para próxima aula**
 - Winder, R.; Roberts, G. Desenvolvendo Software em Java. 3 ed. Rio de Janeiro: LTC, 2009.
 - Capítulo 1 e 2