



C206/C06 – Programação Orientada a Objetos
com Java

TIPOS PRIMITIVOS E CONTROLE DE FLUXO

Prof. Christopher Lima
christopher@inatel.br





Agenda

- ☕ Comentários
- ☕ Variáveis e tipos primitivos
- ☕ Casting
- ☕ Saída de Dados
- ☕ Entrada de Dados
- ☕ Estruturas de Decisão
- ☕ Controlando Loops
- ☕ Escopo das Variáveis



Comentários

//Esse é um comentário de uma linha

/*

Esse é um comentário em bloco


Tudo entre os símbolos /* e */ é ignorado pelo compilador

*/



Declarando e usando variáveis

 **O Java é uma linguagem fortemente tipada!**

 No momento da declaração, **devemos colocar o tipo obrigatoriamente**



Declarando e usando variáveis

☕ Dentro de um **bloco**, podemos declarar variáveis e usá-las. Em Java, **toda variável tem um tipo que não pode ser alterado**, uma vez declarado.

```
int idade;
```

☕ Agora temos uma variável “idade” do tipo inteiro (int).



Declarando e usando variáveis

☕ Podemos atribuir valores a essa nova variável

```
idade = 15;
```

☕ Agora a variável “idade” tem valor 15

```
int idade = 15;
```

☕ Podemos também declarar e atribuir na mesma linha!

Tipos Primitivos



Classificação	Tipo	Descrição
Lógico	boolean	Pode possuir os valores true (verdadeiro) ou false (falso)
Inteiro	byte	Abrange de -128 a 127 (8 bits)
	short	Abrange de -32768 a 32767 (16 bits)
	int	Abrange de -2147483648 a 2147483647 (32 bits)
	long	Abrange de -2^{63} a $(2^{63})-1$ (64 bits)
Ponto Flutuante	float	Abrange de $1.40239846 \times 10^{-46}$ a $3.40282347 \times 10^{38}$ com precisão simples (32 bits)
	double	Abrange de $4.940656458412465544 \times 10^{-324}$ a $1.7976931348623157 \times 10^{308}$ com precisão dupla (64 bits)
Caracter	char	Pode armazenar um caracter unicode (16 bits)



O Tipo String

☕ Para representar uma cadeia de caracteres, o Java possui uma classe especial chamada **String**. Porém, **pode ser usada como se fosse um tipo primitivo**.

☕ E ainda possui diversos métodos para sua manipulação.



O Tipo String

```
String nome = "Maria";
```

```
nome.length(); //Mostra o tamanho
```

```
nome.toUpperCase(); //Coloca em caixa-alta
```


```
nome.equalsIgnoreCase("maria"); //compara duas strings
```

```
nome.startsWith("M"); // Verifica se começa com uma string
```

```
nome.replace("Maria", "Ana"); //Troca a string "Java" por "C#"
```



Exercício 1

 Exercício 1 – Zé do Lanche





Exercício 1

Exercício 1 – Zé do Lanche

Você e seus amigos estão no famoso trailer do Zé e querem computar quantos lanches foram consumidos nas últimas três horas. Na primeira hora foram 10, na segunda 4 e na terceira 2. Faça um programa que calcule e imprima o número total e a média de lanches consumidos.



Casting e Promoções

☕ Alguns valores são incompatíveis se tentarmos fazer uma atribuição

```
1. double d = 3.1415;  
   int i = d; // não compila
```



Casting e Promoções

☕ Alguns valores são incompatíveis se tentarmos fazer uma atribuição

1.

```
double d = 3.1415;  
int i = d; // não compila
```
2.

```
double d = 5; // ok, o double pode conter um número inteiro  
int i = d; // não compila
```



Casting e Promoções

☕ Alguns valores são incompatíveis se tentarmos fazer uma atribuição

1.

```
double d = 3.1415;  
int i = d; // não compila
```
2.

```
double d = 5; // ok, o double pode conter um número inteiro  
int i = d; // não compila
```
3.

```
long x = 10000;  
int i = x; // não compila, pois pode estar perdendo informação
```



Casting e Promoções

☕ Como resolver esse problema?

☕ Com o *Casting*!

☕ A Promoção pode ser vista como um *Casting* implícito!

```
double numero= 3.14;
```

```
int numero2 = (int) numero;
```

```
int x= 10;
```

```
long x1 = x; //Promoção
```



Saída de Dados

☕ É quando pegamos algo da memória e mostramos para o usuário

float nota = 7.5;

1) *System.out.print ("Sua nota é:" + nota);*

Mostra o valor sem quebra de linha;



Saída de Dados

☕ É quando pegamos algo da memória e mostramos para o usuário

`float nota = 7.5F;`

1) `System.out.print ("Sua nota é:" + nota);`

Mostra o valor sem quebra de linha;

2) `System.out.println ("Sua nota é:" + nota);`

Mostra o valor com quebra de linha;



Entrada de Dados

☕ Para realizar a entrada de dados pelo terminal precisamos importar o pacote `java.util.Scanner`

```
//Declara o scanner  
Scanner valorTeclado = new Scanner(System.in);
```



Entrada de Dados

☕ Para realizar a entrada de dados pelo terminal precisamos importar o pacote `java.util.Scanner`

```
//Declara o scanner  
Scanner valorTeclado = new Scanner(System.in);  
  
//Lê um inteiro digitado  
int meuInteiro = valorTeclado.nextInt();
```



Entrada de Dados

☕ Para realizar a entrada de dados pelo terminal precisamos importar o pacote `java.util.Scanner`

```
//Declara o scanner
Scanner valorTeclado = new Scanner(System.in);

//Lê um inteiro digitado
int meuInteiro = valorTeclado.nextInt();

//Lê um float digitado
float meuFloat = valorTeclado.nextFloat();
```



Entrada de Dados

☕ Para realizar a entrada de dados pelo terminal precisamos importar o pacote `java.util.Scanner`

```
//Declara o scanner  
Scanner valorTeclado = new Scanner(System.in);
```

```
//Lê um inteiro digitado  
int meuInteiro = valorTeclado.nextInt();
```

```
//Lê um float digitado  
float meuFloat = valorTeclado.nextFloat();
```

```
//Lê uma cadeia de caracteres digitadas  
String meuNome = valorTeclado.nextLine();
```



Exercício 2

Exercício 2 – Zé do Lanche





Exercício 2

Exercício 2 – Zé do Lanche 2

Faça um programa que consiga ler o número de lanches consumidos no trailler do Zé. Mostre a soma e a média. Utilize o pacote *java.util.Scanner* para ler os dados e utilize a saída com o método *System.out.println()*

```
Scanner entrada = new Scanner(System.in);  
int numero = entrada.nextInt();
```

Após a utilização do Scanner, é necessário fechá-lo:

```
entrada.close();
```



Estruturas de Decisão

If - Else

```
if(expressão booleana){  
    //Comandos  
} else{  
    //Comandos  
}
```




Estruturas de Decisão

```
int quantidadeLanche;
```

```
if(quantidadeLanche <= 0){ //Verifica se a quantidade  
de lanche é <= 0
```

```
    System.out.println("Coma mais lanche");
```

```
} else{
```

```
    System.out.println("Coma uma sobremesa");
```

```
}
```



Exercício 3

Exercício 3 – Vida de Estudante!

Faça um programa que receba uma NPA e informe se o aluno passou ou se ficou de NP3. Caso tenha ficado de NP3, informe a nota e faça a soma com NPA, para o cálculo da NFA. Imprima se ele passou ou não! Utilize a classe *Scanner* para ler as notas do aluno:

Dica: *Scanner entrada = new Scanner(System.in);*

ps: Lembre-se de importar o pacote para a classe Scanner.



Estruturas de Decisão



Switch - Case

```
switch (expressão que resulte em Inteiro/ String ){  
  
    case: valor1  
        //faz alguma coisa  
        break;  
    case: valor2  
        //faz outra coisa  
        break;  
    default:  
        //faz algo caso nenhuma outra opção seja acessada  
  
}
```



Estruturas de Decisão

Switch – Case

-  Caso o comando *break* não seja colocado, as demais opções também serão executadas, até encontrar algum *break* ou executar o último bloco.
-  O comando *default* não é obrigatório, mas pode ser interessante definir um comportamento padrão, caso nenhuma opção seja acessada!



Estruturas de Decisão

Switch – Case - Exemplo

```
int numAlunos;  
switch (numAlunos){  
    case 10:  
        //executa a ação para 10 alunos  
        break;  
    case 30:  
        //executa a ação para 30 alunos  
        break;  
    default:  
        //executa ação padrão – nada foi satisfeito  
        anteriormente  
}
```



Exercício 4

Exercício 4 – Sala de Aula!

Faça um programa que receba o número de alunos matriculados na disciplina C206/C06 e imprima a sala onde o curso será ministrado. Esse número pode ser 10, 20 ou 30 alunos. Caso o número seja 10 ou 20 a sala utilizada pode ser a I-16. Caso o número seja 30, então deve ser utilizada a sala I-22. Caso o número de alunos não seja nenhum desses 3, o software deve mostrar uma mensagem ao usuário. Use a estrutura *switch-case*.



Estruturas de Repetição

While

```
while (expressão verdadeira){  
    //executa o loop até que a condição seja falsa  
}
```

Do - While

```
do{  
    //Executa o loop até que a condição seja falsa  
    //Ele será executado pelo menos uma vez  
}while(expressão verdadeira);
```



Estruturas de Repetição

For

```
for (inicialização; condição; incremento/decremento){  
    //executa o loop até que a condição seja falsa  
}
```




Controlando o Loop

- ☕ Podemos utilizar os comandos *break* e *continue* para controlar os loops
- ☕ O *break* interrompe o loop
- ☕ O *continue* continua para a próxima iteração.



Controlando o Loop

Break

```
for (int i = 0; i < 10; i++){  
    if(i == 5)  
        break; //força o loop a terminar  
}  
//Continua a execução aqui
```



Controlando o Loop

Continue

```
for (int i = 0; i < 10; i++){  
    if(i == 5){  
        continue; //força o loop para o fim  
                //A condição será testada novamente  
    }  
    System.out.println(i); //O 5 não será impresso  
}
```



Escopo das Variáveis

- ☕ O acesso as variáveis depende de onde elas foram declaradas.
- ☕ Se forem declaradas dentro de um bloco, elas só existem dentro daquele bloco



Escopo das Variáveis

```
int x = 3; //Acessado de qualquer lugar
if(x < 10){
    int y = 30; //O y não pode ser acessado fora desse bloco "if"
}
y = y + 1; //Não compila
```



Escopo das Variáveis

```
int x = 3; //Acessado de qualquer lugar  
int y = 5;  
if(x < 10){  
    int y = 30; //não compila pois y está sendo redeclarado  
}
```





Exercício 5

🔥 Exercício 5 – Adivinhação!

Faça um programa que gere um número aleatório entre 1 e 10, e depois pergunte ao usuário qual número foi gerado. O jogo deve continuar até que o usuário adivinhe o número.

*Dica: use Random rand = new Random();
int x = rand.nextInt(10) + 1; //Gera número entre 1
 //e 10*



Exercício 6

Exercício 6 – Adivinhação Aprimorado!

Repita o exercício anterior, mas informe o usuário se o número que ele digitou é maior ou menor que o gerado pelo computador!



Resolução dos Exercícios

https://github.com/chrislima-inatel/C206_C125





Material Complementar



- ☕ Capítulo 3 da apostila FJ-11
 - ☕ Variáveis Primitivas e Controle de Fluxo