

ECM251 – Linguagens de Programação I

Aula 03 – L1/1, L2/1 e L3/1

Engenharia da Computação – 3ª série

Variáveis, Operadores e Expressões Aritméticas ***(L1/1, L2/1 e L3/1)***

2025

ECM251 – Linguagens de Programação I

Aula 03 – L1/1, L2/1 e L3/1

Horário

Terça-feira: 2 x 2 aulas/semana

- L1/1 (07h40min-09h20min): *Prof. Calvetti*;
- L1/2 (09h30min-11h10min): *Prof. Calvetti*;
- L2/1 (07h40min-09h20min): *Prof. Menezes*;
- L2/2 (11h20min-13h00min): *Prof. Calvetti*;
- L3/1 (09h30min-11h10min): *Prof. Evandro*;
- L3/2 (11h20min-13h00min): *Prof. Evandro*.

ECM251 – Linguagens de Programação I

Variáveis e Operadores

Tópico

- A classe ***Math***

A classe **Math**

- Contém vários métodos de funções matemáticas, por exemplo:
 - ***Math.sqrt(x)***: retorna o valor da raiz quadrada do número colocado entre os parênteses;
 - ***Math.random()***: retorna um número aleatório e real, entre 0 e 1;
 - ***Math.abs(x)***: retorna o valor absoluto do número colocado entre os parênteses;
 - ***Math.pow(x, y)***: retorna o valor do primeiro parâmetro, elevado ao segundo parâmetro.

A classe **Math** em Java

- Pelo fato da classe **Math** estar na **package java.lang**, não é necessário importá-la para ser utilizada.
- Exemplo:

```
public class Mathtest {  
  
    public static void main(String[] args){  
        System.out.println("'e' elevado ao quadrado = " + Math.exp(2));  
        System.out.println("2 elevado ao cubo = " + Math.pow(2, 3));  
    }  
}
```

A classe **Math** em Java

- Exemplos da classe **Math**:

```
public class constantes {  
  
    public static void main(String[] args){  
        System.out.println("O valor de pi é: " + Math.PI);  
        System.out.println("O valor de E é: " + Math.E);  
    }  
  
}  
  
public static float IMC(float peso, float altura){  
    return peso/(float)Math.pow(altura, 2);  
}  
  
public static float quadrado(float num){  
    return Math.pow(num,2);  
}
```

A classe **Math** em Java

- Exemplos da classe **Math**:

```
public class RaizDePi {  
  
    public static void main(String[] args){  
        System.out.println("A raiz quadrada de Pi é = "+ Math.sqrt( Math.PI ) );  
    }  
}  
  
public class logaritmos {  
  
    public static void main(String[] args){  
        System.out.println("O logaritmo natural de 10 é = "+ Math.log(10) );  
        System.out.println("O logaritmo natural de 'e' é = "+ Math.log( Math.E ) );  
    }  
}
```

A classe **Math** em Java

- Exemplos da classe **Math**:

```
public class Trigonometricas {  
  
    public static void main(String[] args){  
        System.out.println("O seno de 90 é = "+ Math.sin( (Math.PI)/2 ) );  
        System.out.println("O cosseno de 0 é = "+ Math.cos(0) );  
        System.out.println("A tangente de 45 é= "+ Math.tan( (Math.PI)/4 ));  
    }  
}
```


A classe **Math** em Java

- A classe **Math**:

Para calcular o módulo de um número 'numero' usamos: **Math.abs(numero)**

Para calcular o valor mínimo de dois números 'num1' e 'num2', usamos: **Math.min(num1,num2)**

Para calcular o valor máximo de dois números 'num1' e 'num2', usamos: **Math.max(num1,num2)**

Para arredondar um número 'numero' para cima, usamos: **Math.ceil(numero)**

Para arredondar um número 'numero' para baixo, usamos: **Math.floor(numero)**

Estes métodos, assim como todos os outros, recebem e retornam double. Caso deseje receber a passar outro tipo, use cast conforme foi explicado no método **pow()**.

Mais métodos matemáticos em Java

Acesse a documentação:

<http://docs.oracle.com/javase/6/docs/api/java/lang/Math.html>

Tópico

- Um pedaço da ***String***

Um Pedaco da **String**

- Para se obter uma parte de uma **String** deve ser utilizado o método ***substring(inicio, fim)*** , onde:
 - ***inicio*** é um inteiro que marca a posição do início do pedaço que se quer, sendo a primeira posição da **String** igual à 0;
 - ***fim*** é um inteiro que marca a posição do fim do pedaço que se quer, sempre acrescido de um (+1), por exemplo: ao se colocar o ***fim*** igual à 2, o método vai retornar até o caractere que está na posição 1 da **String**; ao se colocar o tamanho da **String** em ***fim***, vai retornar até o último caractere dela.

Um Pedaco da **String**

- Exemplos:
 - **"012345".substring(0,6)** retorna **"012345"**;
 - **"012345".substring(0,2)** retorna **"01"**;
 - **"012345".substring(3,4)** retorna **"3"**;
 - **"012345".substring(3,3)** retorna **""**;
 - **"012345".substring(0,7)** retornará um erro de exceção (**stringOutOfBoundsException**), pois esta **String** tem início em 0 e termina em 5, e não em 6.

ECM251 – Linguagens de Programação I

Variáveis e Operadores

Tópico

- Entrada de dados via ***Scanner***

Entrada de dados via *Scanner*

- Para se obter a entrada de dados digitados via o console do Java, pode-se utilizar os métodos da classe ***Scanner***;
- Para isto, antes de mais nada, deve-se importá-la através de:

import java.util.Scanner;

- Depois, deve-se instanciar um objeto, por exemplo ***sc***:

Scanner sc = new Scanner(System.in);

- E, a seguir, invocar de seu objeto o método desejado, por exemplo ***nextInt()***:

sc.nextInt();

Entrada de dados via *Scanner*

- Exemplo:

```
import java.util.Scanner;

public class TestaDeclaracaoScanner {
    public static void main(String[] args) {
        //Lê a partir da linha de comando
        Scanner sc1 = new Scanner(System.in);
        String textoString = "Maria Silva";
        //Lê a partir de uma String
        Scanner sc2 = new Scanner(textoString);
    }
}
```

Entrada de dados via *Scanner*

- Exemplo:

```
import java.util.Scanner;
public class ExemplosScanner{
    public static void main(String args[]){
        String nome;
        int idade;
        double media;
        Scanner sc = new Scanner(System.in);
        System.out.println("Digite seu nome: ");
        nome = sc.nextLine();
        System.out.println("Digite sua idade: ");
        idade = sc.nextInt();
        System.out.println("Digite sua media: ");
        media = sc.nextDouble();
        System.out.println("O aluno " + nome + " tem " +
            idade + " anos e ficou com a media " + media);
    }
}
```


Entrada de dados via *Scanner*

- Exemplo:

```
import java.util.Scanner;

public class ContaTokens {
    public static void main(String[] args) {
        int i = 0;
        Scanner sc = new Scanner(System.in);
        System.out.print("Digite um texto:");
        while(sc.hasNext()){
            i++;
            System.out.println("Token: "+sc.next());
        }
        sc.close(); //Encerra o programa
    }
}
```

Entrada de dados via *Scanner*

- Exemplo:

:
:

```
Scanner sc = new Scanner(System.in);
```

```
float numF = sc.nextFloat();  
int num1 = sc.nextInt();  
byte byte1 = sc.nextByte();  
long lg1 = sc.nextLong();  
boolean b1 = sc.nextBoolean();  
double num2 = sc.nextDouble();  
String nome = sc.nextLine();
```

Entrada de dados via *Scanner*

- Exemplo:

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class LerArquivo {

    public static void lerTexto(String nomeArquivo){
        try {
            File arquivo = new File(nomeArquivo);
            Scanner sc = new Scanner(arquivo);
            while(sc.hasNext()){
                System.out.print(sc.nextLine());
            }
            sc.close();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        lerTexto("poema.txt");
    }
}

/* poema.txt deve ser um arquivo texto localizado na mesma pasta da classe LerArquivo,
 * contendo o texto: "A conduta é um espelho no qual todos exibem sua imagem" - Johann Goethe
 */
```

ECM251 – Linguagens de Programação I

Variáveis e Operadores

Principais métodos da classe *Scanner*

Método	Descrição
close()	Fecha o escaneamento de leitura.
findInLine()	Encontra a próxima ocorrência de um padrão ignorando máscaras ou strings ignorando delimitadores.
hasNext()	Retorna um valor booleano verdadeiro (true) se o objeto Scanner tem mais dados de entrada.
hasNextXYZ()	Retorna um valor booleano como verdadeiro (true) se a próxima entrada a qual XYZ pode ser interceptada como Boolean, Byte, Short, Int, Long, Float ou Double.
match()	Retorna o resultado da pesquisa do último objeto Scanner atual.
next()	Procura e retorna a próxima informação do objeto Scanner que satisfazer uma condição.
nextBigDecimal(), nextBigInteger()	Varre a próxima entrada como BigDecimal ou BigInteger.
nextXYZ()	Varre a próxima entrada a qual XYZ pode ser interceptado como boolean, byte, short, int, long, float ou double.
nextLine()	Mostra a linha atual do objeto Scanner e avança para a próxima linha.
radix()	Retorna o índice atual do objeto Scanner.
remove()	Essa operação não é suportada pela implementação de um Iterator.
skip()	Salta para a próxima pesquisa de um padrão especificado ignorando delimitadores.
string()	Retorna uma string que é uma representação do objeto Scanner.

ECM251 – Linguagens de Programação I

Variáveis e Operadores

Tópico

- Saída de dados via ***System.out***

Saída de dados via ***System.out***

- O objeto ***System.out*** é a saída padrão, que permite exibir as ***Strings*** no console (terminal) de comando quando o aplicativo de Java é executado;
- Dentro desse objeto existem métodos para gerar saídas de ***Strings***, entre eles ***println***, ***print*** e o ***printf***.

O método ***System.out.println()***

- A instrução ***System.out.println()*** gera uma saída do texto entre aspas, ou ***String***, criando uma nova linha e posicionando o cursor na linha abaixo, o que é identificado pela terminação “***ln***”.

```
public class Texto_Simples {  
    public static void main(String[] args) {  
        System.out.println("Seu texto é inserido aqui, entre aspas");  
    }  
}
```


O método *System.out.print()*

- O método ***print***, se for observado não possui o “*ln*”, por isso exibe uma ***String*** sem criar uma nova linha, deixando o cursor na mesma linha.

```
public class Texto_Simples_print {  
    public static void main(String[] args) {  
        System.out.print("José");  
        System.out.print("Silva Moraes");  
    }  
}
```


O caractere de escape

- O caractere de escape pode ser considerado um caractere especial, permitindo inserir uma nova linha dentro dos métodos ***print*** e ***println*** do objeto ***System.out***;
- ‘\n’ não é impresso, pois o Java identifica que é uma sequência de escape, formada pela barra invertida e um caractere de escape, dentro de uma ***String*** de caracteres;
- A sequência de escape ‘\n’ é representada por um caractere de nova linha, o ‘n’, fazendo que o cursor de saída da tela mova-se para o começo de uma nova linha.

O caractere de escape

- A tabela a seguir apresenta algumas sequências de escapes mais utilizadas:

Sequência de escape	Descrição
\n	Nova linha. Posiciona o cursor de tela no início da próxima linha
\t	Tabulação horizontal. Move o cursor de tela para a próxima parada de tabulação.
\r	Posiciona o cursor da tela no início da linha atual - não avança para a próxima linha. Qualquer saída de caracteres gerada depois de algum retorno já gerado é sobrescrito os caracteres anteriores gerados na linha atual.
\\	Barras invertidas. Utilizada para imprimir um caractere de barra invertida.
\"	Aspas duplas. Utilizada para imprimir um caractere de aspas duplas. Exemplo, <code>System.out.println(\"\"aspas\"");</code> exibe “aspas”

Variáveis e Operadores

O caractere de escape

- Exemplo:

```
public class Texto_sequencia_caractere {  
    public static void main(String[] args) {  
        System.out.print("Antônio \n Vieira \n dos\n Santo\n ");  
    }  
}
```

O método ***System.out.printf()***

- O argumento do método ***printf*** é uma ***String*** de formato que pode consistir em texto fixo e especificadores de formato;
- A letra “f” no final da palavra “***print***” significa “***formatted***” ou seja, exibe os dados formatados;
- Os especificadores de formato são como marcadores de lugares para um valor, especificando o tipo da saída dos dados que iniciam com um sinal de porcentagem (%), seguido por um caractere representando seu tipo de dado.

O método *System.out.printf()*

- No exemplo a seguir, o especificador de formato **%s**, que é um marcador de lugar para uma **String**, se for especificado um número no lugar irá gerar um erro:

```
public class Texto_printf{  
    public static void main(String[] args) {  
        System.out.printf("%s\n %s\n", "Marcela", "Nogueira");  
    }  
}
```

O método ***System.out.printf()***

- No tabela abaixo são apresentados alguns dos especificadores de formatos mais utilizados:

%d	representa números inteiros
%f	representa números floats
%2f	representa números doubles
%b	representa valores booleanos
%c	representa valores char

O método *System.out.printf()*

- Exemplo:

```
public class Testa_Especificador {  
    public static void main(String[] args) {  
        int num1 = 10;  
        int num2 = 30;  
        System.out.printf("Soma das variáveis num1 e num 2 = %d", (num1 + num2));  
    }  
}
```


Exercícios

1. Escreva um programa para uma loja que imprima os dados de uma compra em uma Nota Fiscal. Quando o programa é executado, deve-se fornecer, via teclado, os seguintes dados:
 - Nome do comprador;
 - Produto adquirido;
 - Quantidade adquirida;
 - Preço unitário do produto.



Exercícios

2. Escreva um programa para uma loja que imprima os dados de uma compra em uma Nota Fiscal. Quando o programa é executado, deve-se fornecer, via arquivo, os seguintes dados:
- Nome do comprador;
 - Produto adquirido;
 - Quantidade adquirida;
 - Preço unitário do produto.



Exercícios

- Extras, propostos pelo professor em aula, utilizando os conceitos abordados neste material...



ECM251 – Linguagens de Programação I

Variáveis e Operadores

Bibliografia Básica

- MILETTO, Evandro M.; BERTAGNOLLI, Silvia de Castro. Desenvolvimento de software II: introdução ao desenvolvimento web com HTML, CSS, javascript e PHP (Tekne). Porto Alegre: Bookman, 2014. E-book. Referência Minha Biblioteca:
<https://integrada.minhabiblioteca.com.br/#/books/9788582601969>
- WINDER, Russel; GRAHAM, Roberts. Desenvolvendo Software em Java, 3ª edição. Rio de Janeiro: LTC, 2009. E-book. Referência Minha Biblioteca:
<https://integrada.minhabiblioteca.com.br/#/books/978-85-216-1994-9>
- DEITEL, Paul; DEITEL, Harvey. Java: how to program early objects. Hoboken, N. J: Pearson, c2018. 1234 p. ISBN 9780134743356.

Continua...

Bibliografia Básica (continuação)

- HORSTMANN, Cay S; CORNELL, Gary. Core Java. SCHAFRANSKI, Carlos (Trad.), FURMANKIEWICZ, Edson (Trad.). 8. ed. São Paulo: Pearson, 2010. v. 1. 383 p. ISBN 9788576053576.
- LIANG, Y. Daniel. Introduction to Java: programming and data structures comprehensive version. 11. ed. New York: Pearson, c2015. 1210 p. ISBN 9780134670942.
- TURINI, Rodrigo. Desbravando Java e orientação a objetos: um guia para o iniciante da linguagem. São Paulo: Casa do Código, [2017]. 222 p. (Caelum).

Bibliografia Complementar

- HORSTMANN, Cay. Conceitos de Computação com Java. Porto Alegre: Bookman, 2009. E-book. Referência Minha Biblioteca:
<https://integrada.minhabiblioteca.com.br/#/books/9788577804078>
- MACHADO, Rodrigo P.; FRANCO, Márcia H. I.; BERTAGNOLLI, Silvia de Castro. Desenvolvimento de software III: programação de sistemas web orientada a objetos em java (Tekne). Porto Alegre: Bookman, 2016. E-book. Referência Minha Biblioteca:
<https://integrada.minhabiblioteca.com.br/#/books/9788582603710>
- BARRY, Paul. Use a cabeça! Python. Rio de Janeiro: Alta Books, 2012. 458 p.
ISBN 9788576087434.

Continua...

ECM251 – Linguagens de Programação I

Variáveis e Operadores

Bibliografia Complementar (continuação)

- LECHETA, Ricardo R. Web Services RESTful: aprenda a criar Web Services RESTfulem Java na nuvem do Google. São Paulo: Novatec, c2015. 431 p.
ISBN 9788575224540.
- SILVA, Maurício Samy. JQuery: a biblioteca do programador. 3. ed. rev. e ampl. São Paulo: Novatec, 2014. 544 p.
ISBN 9788575223871.
- SUMMERFIELD, Mark. Programação em Python 3: uma introdução completa à linguagem Phython. Rio de Janeiro: Alta Books, 2012. 506 p.
ISBN 9788576083849.

Continua...

ECM251 – Linguagens de Programação I

Variáveis e Operadores

Bibliografia Complementar (continuação)

- YING, Bai. Practical database programming with Java. New Jersey: John Wiley & Sons, c2011. 918 p.
- ZAKAS, Nicholas C. The principles of object-oriented JavaScript. San Francisco, CA: No Starch Press, c2014. 97 p. ISBN 9781593275402.

ECM251 – Linguagens de Programação I

Aula 03 – L1/1, L2/1 e L3/1

FIM

ECM251 – Linguagens de Programação I

Aula 03 – L1/2, L2/2 e L3/2

Engenharia da Computação – 3ª série

Variáveis, Operadores e Expressões Aritméticas ***(L1/2, L2/2 e L3/2)***

2025

ECM251 – Linguagens de Programação I

Aula 03 – L1/2, L2/2 e L3/2

Horário

Terça-feira: 2 x 2 aulas/semana

- L1/1 (07h40min-09h20min): *Prof. Calvetti*;
- L1/2 (09h30min-11h10min): *Prof. Calvetti*;
- L2/1 (07h40min-09h20min): *Prof. Menezes*;
- L2/2 (11h20min-13h00min): *Prof. Calvetti*;
- L3/1 (09h30min-11h10min): *Prof. Evandro*;
- L3/2 (11h20min-13h00min): *Prof. Evandro*.

Problema

Crie um algoritmo para calcular o valor do seu IMC – Índice de Massa Corporal, com base em seu peso e sua altura.

A fórmula é $IMC = \text{peso} / \text{altura} * \text{altura}$, onde o peso deve estar em kg e altura em metros.

- Use o tipo de dados real ao invés de inteiro para a declaração das variáveis;
- Os números reais usam como marcador decimal o ponto ao invés da vírgula, logo, se sua altura for um metro e setenta centímetros, digite 1.70 e não 1,70.



ECM251 – Linguagens de Programação I

Variáveis e Operadores

Resolvido

```
import javax.swing.JOptionPane;

public class Imc {
    public static void main(String[] args) {
        //lendo os dados
        String sPeso = JOptionPane.showInputDialog(
            "Digite seu peso em kilogramas:");
        String sAltura = JOptionPane.showInputDialog(
            "Digite sua altura em metros:");
        //convertendo para reais
        double peso = Double.parseDouble(sPeso);
        double altura = Double.parseDouble(sAltura);
        //calculando o imc
        double imc = peso / (altura * altura);
        //apresentando o resultado
        JOptionPane.showMessageDialog(null, "IMC = " + imc);
    }
}
```

Exercícios

1. Crie um algoritmo para calcular o número de dias que você está vivo, com base em sua idade, que deverá ser digitada.



Exercícios

2. Crie um algoritmo para calcular a área de um retângulo, com base nas medidas de sua base e de sua altura.



Exercícios

3. Crie um algoritmo que leia um número inteiro e o eleve ao quadrado usando a classe **Math**, onde a base e o expoente são números reais.



Exercícios

4. Crie um algoritmo capaz de ler três *Strings* quaisquer, digitadas por meio do console, e apresente a soma do comprimento destas ***Strings***.



Exercícios

5. Crie um algoritmo capaz de ler dez palavras quaisquer, gravadas em um arquivo texto, e as apresente no console em ordem inversa do arquivo.



Exercícios

6. Crie um algoritmo que calcule o valor de uma dívida, submetida à juros compostos:

$$\text{ValorFinal} = \text{ValorInicial} * (1 + J / 100) ^ N$$

E seja capaz de responder, se você deve para o cartão de crédito R\$ 100,00, a uma taxa de juros de 10%, quanto deverá ser pago depois de 8 meses.

Onde:

- J representa os juros (em %); e
- N representa o número de meses.



Exercícios

7. Entrar com um único número inteiro, com 5 dígitos, e imprimir o algoritmo correspondente à casa da dezena.



Exercícios

8. Entrar com um ângulo em graus e imprimir seu seno, cosseno, tangente, secante, cossecante e cotangente.



Exercícios

9. Entrar com um número e imprimir o seu logaritmo na base 10.



Exercícios

10. Entrar com um número e a base em que se deseja calcular o logaritmo desse número. Após isto, calcular tal logaritmo e imprimir o resultado.



Exercícios

11. Crie um algoritmo que embaralhe mensagens fazendo o seguinte: leia três frases, separe cada uma delas ao meio. Então junte nesta ordem: primeira metade da segunda, segunda metade da terceira, segunda metade da segunda, primeira metade da primeira, primeira metade da terceira, segunda metade da primeira. Concatene então as três frases originais e imprima o resultado. Na linha de baixo, escreva a frase embaralhada e compare o resultado.



Exercícios

12. Entre com uma data em uma variável do tipo inteiro no formato ***ddmmaa*** e imprimir dia, mês e ano separados.



Exercícios

13. Entre com uma data em uma variável do tipo ***String*** no formato ***dd/mm/aa*** e imprimir dia, mês e ano separados.



Exercícios

14. Escrever um algoritmo que leia três números reais ***a***, ***b*** e ***c***, calcule e escreva o resultado da expressão:

$$x = 2 * ((a - c) / 8) - b * 5.$$



Exercícios

15. Crie um algoritmo para calcular a área de um círculo, com base no seu raio ($\text{Área} = \text{PI} * \text{raio} * \text{raio}$, onde $\text{PI} = 3.14159$). Use variáveis reais.



Exercícios

16. Ler dois números inteiros e imprimir dividendo, divisor, quociente e resto.



Exercícios

17. Entrar com um número e imprimir o número, seu quadrado e sua raiz quadrada.



Exercícios

18. Criar um algoritmo que calcule e imprime a área de um triângulo.



Exercícios

19. Criar um algoritmo que calcule e imprime a área de um losango.



Exercícios

20. Criar um programa capaz de calcular o 3º lado de um triângulo, digitados os outros dois lados e o ângulo entre eles esses dois lados.



Atividade

- Individualmente, resolver os exercícios propostos e apresentar à sala, explicando-a, na próxima aula L1/2 e L2/2, a solução daquele solicitado pelo professor.

Bibliografia (apoio)

- LOPES, ANITA. GARCIA, GUTO. Introdução à Programação: 500 algoritmos resolvidos. Rio de Janeiro: Elsevier, 2002.
- DEITEL, P. DEITEL, H. Java: como programar. 8 Ed. São Paulo: Prentice-Hall (Pearson), 2010.

ECM251 – Linguagens de Programação I

Aula 03 – L1/2, L2/2 e L3/2

FIM