Curso Básico Programação Java

13/04/2017

Ìndice

- **Capítulo 5 - Operadores lógicos e if/else**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3
    4. Exercício 4
    5. Exercício 5
    6. Exercício 6
    7. Exercício 7
    8. Exercício 8

- **Capítulo 6 - Comandos if/else aninhados**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3
    4. Exercício 4
    5. Exercício 5
    6. Exercício 6
    7. Exercício 7
    8. Exercício 8
    9. Exercício 9
    10. Exercício 10
    11. Exercício 11
    12. Exercício 12
    13. Exercício 13
    14. Exercício 14
    15. Exercício 15
    16. Exercício 16

- **Capítulo 7 - Comando switch/case**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3
    4. Exercício 4
    5. Exercício 5

- **Capítulo 8 - Comando while**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3
    4. Exercício 4
    5. Exercício 5

- **Capítulo 9 - Comando do-while**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3
    4. Exercício 4
    5. Exercício 5

- **Capítulo 10 - Comando for**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3
    4. Exercício 4
    5. Exercício 5
    6. Exercício 6
    7. Exercício 7

- **Capítulo 11 - Laços Aninhados**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3
    4. Exercício 4
    5. Exercício 5

- **Capítulo 12 - Métodos Estáticos**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3
    4. Exercício 4
    5. Exercício 5
    6. Exercício 6

- **Capítulo 13 - Arrays Numéricas**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3
    4. Exercício 4
    5. Exercício 5
    6. Exercício 6

- **Capítulo 14 - Arrays de Strings**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3
    4. Exercício 4

- **Capítulo 15 - Arrays Bidimensionais**

    1. Exercício 1
    2. Exercício 2
    3. Exercício 3

- **Capítulo 16 - Manipulação de Strings**

    1. Exercício 1
    2. Exercício 2

Capítulo 1 - Conceitos Básicos

Cap. 1 - Exercício 1

Objetivo:

Construa uma aplicação em Java que imprima exatamente a seguinte frase: "Aprenda Java".

Passos:

Construa uma classe pública chamada **Exercicio1**
Essa classe possui o método main que deve imprimir a frase: "Aprenda Java"

Dicas:

Os principais comandos de impressão em Java são:
System.out.println("texto")
Imprime o texto em uma nova linha.

System.out.print("texto")
Imprime o texto na mesma linha.

Exemplo do comando de impressão:

```
1.    System.out.println("texto");
```

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiusculas e minusculas.

Cap. 1 - Exercício 2

Objetivo:

Construa uma aplicação em Java que imprima exatamente as palavras abaixo em duas linhas:

```
Aprenda
Java
```

Passos:

Construa uma classe pública chamada **Exercicio2**
Essa classe possui o método main que deve imprimir as palavaras: "Aprenda Java", uma em cada linha.

Dicas:

Use:
System.out.println() para imprimir o texto em uma nova linha.

Restrições:

Imprimir somente as informações solicitadas, respeitando os espaços em branco e as letras maiusculas e minusculas.

Capítulo 2 - Entrada de Dados e Variáveis

Cap. 2 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba uma palavra digitada pelo usuário e a imprima.

Passos:

Construa uma classe pública chamada **Exercicio1**
Essa classe possui o método main que deve receber a palavra digitada pelo usuário.
Após receber a palavra deve-se exibir a mesma ao usuário.

Dicas:

Para realizar a captura da palavra digitada utilize o camando abaixo:

```
1.    String recebe = JOptionPane.showInputDialog("Digite uma palavra: ");
```

Para exibir a palavra digitada ao usuário usar o comando apresentado no exercicio anterior.
Exemplo do comando de impressão:

```
1.    System.out.println("texto");
```

Cap. 2 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba o nome e o sobre nome do usuário.
Imprimir o nome e sobre em linhas separadas.

Passos:

Construa uma classe pública chamada **Exercicio2**
Essa classe possui o método main que deve receber os dados solicidatos: **nome** e **sobrenome**.
Após receber as informações solicitadas imprimir as mesmas em linhas separadas.

Dicas:

Receber os dados em duas variáveis **String**
Atenção: No AprendaJava não faça o import da do pacote **javax.swing**

Capítulo 3 - Operadores Aritméticos

Cap. 3 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba um numero digitado pelo usuário, subtraia por 10 e imprima o resultado.

Passos:

Construa uma classe pública chamada **Exercicio1**
Essa classe possui o método main que deve receber um numero digitado pelo usuário.
Subtrair o valor recebido por 10 e imprimir o resultado.

Dicas:

O comando **JOptionPane.showInputDialog** recebe valores no formato String.
Como trabalharemos com numeros devemos converter o valor recebido para numero, nesse caso converteremos para **double**, pois variáveis deste tipo aceitam valores do tipo reais.
Exemplo do comando para receber o valor digitado e convertê-lo para double:

```
1.    String recebe = JOptionPane.showInputDialog("Entre com o valor");
2.    double valor  = Double.parseDouble(recebe);
```

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 3 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba um numero real digitado pelo usuário e o converta para o seu valor oposto, ou seja, se for positivo converte para negativo e se for negativo converte para positivo. Imprimir o valor calculado.

Passos:

Construa uma classe pública chamada **Exercicio2**
Essa classe possui o método main que deve receber um numero digitado pelo usuário.
Utilize o operador **Menos Unário(-)** para realizar a conversão.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 3 - Exercício 3

Objetivo:

Construa uma aplicação em Java que receba um numero real digitado pelo usuário e imprima o dobro de seu valor.

Passos:

Construa uma classe pública chamada **Exercicio3**
Essa classe possui o método main que deve receber um numero digitado pelo usuário.
Calcular o dobro do valor digitado e imprimir o valor ao usuário.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 3 - Exercício 4

Objetivo:

Construa uma aplicação em Java que calcule a área de um quadrado.

Passos:

Construa uma classe pública chamada **Exercicio4**
Essa classe possui o método main que deve receber o valor de 1 lado do quadrado digitado pelo usuário.

Calcular a área do quadrado e imprimir somente o resultado ao usuário.

Dicas:

A fórmula para calcular a área de um quadrado é: **area = lado * lado**
Não se preocupe se o valor do lado digitado for negativo.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 3 - Exercício 5

Objetivo:

Construa uma aplicação em Java que calcule o perimetro de um quadrado.

Passos:

Construa uma classe pública chamada **Exercicio5**
Essa classe possui o método main que deve receber o valor de 1 lado do quadrado digitado pelo usuário.

Calcular o perimetro do quadrado e imprimir somente o resultado ao usuário.

Dicas:

O perímetro é calculado somando todos os lados do quadrado.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 3 - Exercício 6

Objetivo:

Construa uma aplicação em Java que calcule a área de um triângulo.

Passos:

Construa uma classe pública chamada **Exercicio6**
Essa classe possui o método main que deve receber o valor da base e da altura do triângulo digitado pelo usuário.
Calcular a área do triângulo e imprimir somente o resultado ao usuário.

Dica

A área de um triângulo é conhecida pela base vezes altura dividido por 2.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 3 - Exercício 7

Objetivo:

Construa uma aplicação em Java que calcule a área de um trapézio.

Passos:

Construa uma classe pública chamada **Exercicio7**
Essa classe possui o método main que deve receber os três valores para calcular a área de um trapézio.
Os dados necessários para calcular a área são: Base Maior, Base Menor e Altura.
Calcular a área do trapézio e imprimir somente o resultado ao usuário.

Dica:

Procure na WEB como calcular a área do trápezio.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.
A ordem de recebimento dos valores devem ser obrigatóriamente: Base Maior, Base Menor e Altura.

Cap. 3 - Exercício 8

Objetivo:

Construa uma aplicação em Java que receba 2 números e calcule o módulo do 1o. número pelo 2o. número.

Passos:

Construa uma classe pública chamada **Exercicio8**
Essa classe possui o método main que deve receber dois numeros digitados pelo usuário.
Utilizar o operador **Módulo (%)** para realizar o cálculo.

Dica:

Considerar que o segundo número digitado pelo usuário sempre será diferente de zero.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 3 - Exercício 9

Objetivo:

Construa uma aplicação que receba o salário de um funcionário e imprima o salário com reajuste de 30%.

Passos:

Construa uma classe pública chamada **Exercicio9**
Essa classe possui o método main que deve receber o salário digitado pelo usuário.
Calcular o reajuste de 30% e imprimir o salário atualizado.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 3 - Exercício 10

Objetivo:

Construa uma aplicação que receba o salário de um funcionário.
Após receber o salário calcule um bônus de 13% e impostos de 7.5%.
Finalmento calcular o salário liquido, considerando o salário, bônus e imposto.

Passos:

Construa uma classe pública chamada **Exercicio10**
Essa classe possui o método main que deve receber o salário digitado pelo usuário.
Calcular o bônus e impostos. Ambos são calculados sobre o salário recebido;
Calcular e imprimir o salário liquido do funcionário.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 3 - Exercício 11

Objetivo:

Construa uma aplicação em Java que calcule o volume em litros de um cilindro.

Passos:

Construa uma classe pública chamada **Exercicio11**
Essa classe possui o método main que deve receber o valor do raio e da altura do cilindro (em centímetros) digitado pelo usuário.
Calcular e imprimir o valor do volume em litros.

Dicas:

Procure na WEB como calcular o volume do cilindo em litros a partir do seu raio e altura.
Utilize a seguinte constante para o valor de PI:

```
1.    double PI = Math.PI;
```

Para elevar um número ao quadrado você pode utilizar o método **pow**:

```
1.    double quadrado = Math.pow(double numero, double elevado);
```

Em Java não é necessário fazer o import da classe **Math**

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.
Não esquecer de converter o valor para litros.
Ordem de entrada dos dados: **raio** e **altura**.

Cap. 3 - Exercício 12

Objetivo:

Construa uma aplicação em Java que receba 2 valores (salário e porcentagem de aumento) e mostre o salario atualizado.

Passos:

Construa uma classe pública chamada **Exercicio12**
Essa classe possui o método main que deve receber o salario e a porcentagem de aumento digitados pelo usuário.
Calcular e imprimir o salario atualizado.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.
Ordem de entrada dos dados:**salario** e **aumento**.
Se o usuário digitar para o percentual de aumento 87, considerar 87% de aumento.
Se o usuário digitar para o percentual de aumento 0.75, considerar 0.75% de aumento.

Cap. 3 - Exercício 13

Objetivo:

Construa uma aplicação em Java que receba 3 notas e mostre sua média aritmética e média ponderada.

Passos:

Construa uma classe pública chamada **Exercicio13**
Essa classe possui o método main que deve receber 3 notas digitadas pelo usuário.
Calcular a média aritmética
Calcular a média ponderada utilizando os seguintes pesos
nota 1 -> peso 1, nota 2 -> peso 2 e nota 3 -> peso 3.
Imprimir as médias calculadas, uma em cada linha.

Dicas:

Imprima cada resultado utilizando:

```
1.    System.out.println(resultado);
```

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.
Utilizar um **System.out.println(resultado);** para cada valor.
Ordem de entrada dos dados: **nota1**,**nota2** e **nota3**
Ordem de saída dos dados: **média aritmética** e **média ponderada**

Cap. 3 - Exercício 14

Objetivo:

Dadas as seguintes informações:
Medida A = 7 unidades.
Medida B = 5 unidades de A.
Medida C = 77 uninidas de B.
Construa uma aplicação que receba um valor em **unidades**, faça as conversões e mostre os resultados, em unidades, na ordem acima.

Passos:

Construa uma classe pública chamada **Exercicio14**
Essa classe possui o método main que deve receber um valor de unidades.
Calcular as conversões e imprimir o resultado de cada conversão EM LINHAS DISTINTAS.

Dicas:

Para a entrada de dados igual a 10 o esperado é: 70, 350 e 26950.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.
A ordem de saída deve ser: **Medida A**, **Medida B** e **Medida C**.

Cap. 3 - Exercício 15

Objetivo:

Construa uma aplicação que calcule a idade de uma pessoa em anos.
A aplicação deve receber o ano de nascimento e o ano atual.

Passos:

Construa uma classe pública chamada **Exercicio15**
Essa classe possui o método main que deve receber o ano de nascimento e o ano atual.
Nesse código usar somente tipos inteiros.
Calcular e exibir a idade.

Dica:

Esse calculo não é preciso, pois está considerando somente o ano de nascimento.
Exemplo:
Para o ano de nascimento=1970 e o ano atual=2000, a idade esperada será 30 anos.

Restrições:

Ordem de entrada dos dados: **ano de nascimento** e **ano atual**
Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.

Cap. 3 - Exercício 16

Objetivo:

Um trabalhador possui uma conta corrente em determinado banco, o qual cobra uma tarifa de 0.33% de cada operação de retirada efetuada.
Em determinada situação o trabalhador fez duas compras pagas com cheque.
Construa uma aplicação que receba o valor disponível na conta corrente e o valor de cada cheque descontado.
Calcule o saldo atual da conta após o desconto dos cheques.

Passos:

Construa uma classe pública chamada **Exercicio16**
Essa classe possui o método main que deve receber o saldo da conta e o valor de cada cheque.
Calcular e exibir o saldo atual.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilize variáveis do tipo **double**.
Ordem de entrada dos dados:**saldo**,**valor cheque1** e **valor cheque2**.

Cap. 3 - Exercício 17

Objetivo:

Um vendedor ganha 4.5% de comissão sobre as vendas além do salário base.
Construa uma aplicação que receba o valor do salário base e o valor das vendas desse vendedor.
Calcule e mostre o salário final.

Passos:

Construa uma classe pública chamada **Exercicio17**
Essa classe possui o método main que deve receber o salário base e o valor das vendas.
Calcular e exibir o salário final.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilize variáveis do tipo **double**.
Ordem de entrada dos dados:**Salário Base** e **Valor das Vendas**.

Cap. 3 - Exercício 18

Objetivo:

Um funcionário precisa pagar 2 contas em atraso.
Sabe-se que a multa é de 2% sobre cada conta e que ele acabou de receber seu salário.
Construa uma aplicação que receba o salário do funcionário e o valor de cada conta a ser paga.
Calcule quanto restará do salário.

Passos:

Construa uma classe pública chamada **Exercicio18**
Essa classe possui o método main que deve receber o salário do funcionario e o valor de cada conta a ser paga.
Calcular e exibir o restante do salário.

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilize variáveis do tipo **double**.
A ordem de entrada dos dados é:**salário**, **conta1** e **conta2**.

Capítulo 4 - Operadores Relacionais e if

Cap. 4 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba um número inteiro e imprima **MAIOR** se o número recebido for maior que 5.

Passos:

Construa uma classe pública chamada **Exercicio1**
Essa classe possui o método main que deve imprimir somente a palavra **MAIOR** (em maiúsculo) se o número digitado for maior do que 5.

Dicas:

Exemplo da estrutura do comando if:

```
1.    if (condição){
2.          comandos
3.    }
```

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiúsculas e minúsculas.

Cap. 4 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba um número inteiro e imprima **MENOR** (em maiúsculo) se ele for menor do que 0.

Passos:

Construa uma classe pública chamada **Exercicio2**
Essa classe possui o método main que deve imprimir somente a palavra **MENOR** se o número digitado for menor do que 0.

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiúsculas e minúsculas.

Cap. 4 - Exercício 3

Objetivo:

Construa uma aplicação em Java que receba um número inteiro e imprima **IGUAL** se for recebido for igual a 10.

Passos:

Construa uma classe pública chamada **Exercicio3**
Essa classe possui o método main que deve imprimir somente a palavra **IGUAL** (em maiúsculo) se o número digitado for igual a 10.

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiúsculas e minúsculas.

Cap. 4 - Exercício 4

Objetivo:

Construa uma aplicação em Java que receba um número inteiro e imprima **OK** se for maior ou igual a 15.

Passos:

Construa uma classe pública chamada **Exercicio4**
Essa classe possui o método main que deve imprimir somente a palavra **OK** (em maiúsculo) se o número digitado for maior ou igual a 15.

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiúsculas e minúsculas.

Cap. 4 - Exercício 5

Objetivo:

Construa uma aplicação em Java que receba um número inteiro e imprima **OK** se for menor ou igual a 20.

Passos:

Construa uma classe pública chamada **Exercicio5**
Essa classe possui o método main que deve imprimir somente a palavra **OK** (em maiúsculo) se o número digitado for menor ou igual a 20.

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiúsculas e minúsculas.

Cap. 4 - Exercício 6

Objetivo:

Construa uma aplicação em Java que receba um número real e imprima **POSITIVO** se for um número positivo (maior que zero).

Passos:

Construa uma classe pública chamada **Exercicio6**
Essa classe possui o método main que deve imprimir somente a palavra **POSITIVO** (em maiúsculo) se o número digitado for positivo.

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiúsculas e minúsculas.
Utilizar variáveis do tipo **double**.

Cap. 4 - Exercício 7

Objetivo:

Construa uma aplicação em Java que receba um número real e imprima **NEGATIVO** se for um número negativo.

Passos:

Construa uma classe pública chamada **Exercicio7**
Essa classe possui o método main que deve imprimir somente a palavra **NEGATIVO** (em maiúsculo) se o número digitado for negativo (menor que zero).

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiúsculas e minúsculas.
Utilizar variáveis do tipo **double**.

Cap. 4 - Exercício 8

Objetivo:

Construa uma aplicação em Java que receba dois número double e imprima **DIFERENTE** se os números recebidos forem diferentes.

Passos:

Construa uma classe pública chamada **Exercicio8**
Essa classe possui o método main que deve imprimir somente a palavra **DIFERENTE** (em maiúsculo) se os números recebidos forem diferentes.

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiusculas e minusculas.

Cap. 4 - Exercício 9

Objetivo:

Uma empresa está reajustando o salário de seus funcionários que ganham até R$900,00.
Para esses funcionários o reajuste será de 12%.
Construa uma aplicação em Java que receba o salário, calcule o reajuste e imprima o salário atualizado.

Passos:

Construa uma classe pública chamada **Exercicio9**
Essa classe possui o método main que deve receber o salário do funcionário.
Se o salário for menor do que R$ 900,00 faça o reajuste de 12% e imprima o salário atualizado.
Não imprimir informações se o salário for maior ou igual R$ 900,00.

Restrições:

Imprimir somente o valor do salário atualizado.
Utilizar variáveis do tipo **double**.
Considerar que o salário digitado sempre será maior que zero.

Cap. 4 - Exercício 10

Objetivo:

Construa uma aplicação em Java que receba dois números reais.
Se o 1º número for 0 mude seu valor para 1.
Divida o 2º número pelo 1º.
Se o resultado for negativo, converta-o para positivo.

Passos:

Construa uma classe pública chamada **Exercicio10**
Essa classe possui o método main que deve receber dois números reais.
Se o 1º número for 0 mude seu valor para 1.
Divida o 2º número pelo 1º.
Se o resultado for negativo, converta-o para positivo.
Imprimir o resultado final.

Restrições:

Imprimir somente o valor solicitado.
Utilizar variáveis do tipo **double**.

Cap. 4 - Exercício 11

Objetivo:

Um cliente faz uma compra em determinada loja.
A loja possui a seguinte política de descontos:
-Se o valor total da compra for maior do que R$ 500,00 o cliente ganha um desconto de 7%.
-Se a quantidade comprada for maior do que 10 o cliente ganha 5% de desconto.
Construa uma aplicação em Java que receba o valor de um produto, a quantidade comprada
e imprima o valor da compra seguindo a política de descontos da loja.

Passos:

Construa uma classe pública chamada **Exercicio11**
Essa classe possui o método main que deve receber:
-1º o valor do produto;
-2º a quantidade comprada (pode ser quebrada, como 1.1);
Calcular cada desconto e por último calcular e imprimir o resultado.

Dicas

Inicialize as variáveis.
Calcule cada desconto separadamente, baseado no valor original e por último cálcule o preço final.

Restrições:

Imprimir somente o valor.
Utilizar variáveis do tipo **double**.
A ordem de entrada de dados deve ser:**valor do produto** e **quantidade comprada**.

Cap. 4 - Exercício 12

Objetivo:

Construa uma aplicação em Java que receba a idade de 4 pessoas.
Calcule a média das idades.
Se a média for igual ou maior do que 18 imprima **MAIOR** (em maiúsculo).

Passos:

Construa uma classe pública chamada **Exercicio12**
Essa classe possui o método main que deve receber as 4 idades.
Realizar os cálculos necessários e imprimir o resultado.

Restrições:

Imprimir exatamente o resultado solicitado.
Utilizar variáveis do tipo **double**.

Cap. 4 - Exercício 13

Objetivo:

Construa uma aplicação em Java que receba a idade de 3 pessoas.
Calcule e imprima a média das idades válidas (maior ou igual a 18).

Passos:

Construa uma classe pública chamada **Exercicio13**
Essa classe possui o método main que deve receber as 3 idades.
Realizar os cálculos necessários.
Imprimir a média das idades válidas (maior ou igual a 18)
Se não existirem idades válias, não imprimir informações na tela.

Restrições:

Calcule a média somente das pessoas maiores de idade.
Imprima o resultado se pelo menos 1 pessoa tive idade maior ou igual a 18.
Utilizar variáveis do tipo **double**.
Imprimir somente o valor.

Dicas

Inicialize as variáveis. Você só pode fazer calculos com variáveis inicializadas ou com valores armazenados.

Exemplos

Idade 1 = **13**, Idade 2 = **15** e Idade 3 = **17**. Resultado = **Não é esperada nenhuma impressão**.
Idade 1 = **35**, Idade 2 = **35** e Idade 3 = **17**. Resultado = **35**.

Cap. 4 - Exercício 14

Objetivo:

Uma empresa está reajustando o salário de seus funcionários.
O reajuste será de 15% para os salários acima de R$750,00.
Os funcionários que ganham abaixo de R$1.000,00 (considerando o aumento anterior) ganham um Auxilio-Escola de R$ 120,00.
Construa uma aplicação Java que receba o salário, calcule o reajuste e mostre o salário final.

Passos:

Construa uma classe pública chamada **Exercicio14**
Essa classe possui o método main que deve receber o salário de um funcionário.
Calcular e imprimir o novo salário.

Dicas

Inicialize as variáveis.
Calcule 1º o aumento e depois o Auxilio-Escola.

Restrições:

Imprimir somente o valor.
Utilizar variáveis do tipo **double**.

Cap. 4 - Exercício 15

Objetivo:

Os vendedores de uma empresa que possuem um salário base de até R$700,00 (inclusive), ganham uma comissão de 4.5% sobre suas vendas.
Se o volume de vendas for maior do que R$15.000,00 ganham uma bonificação de 1.2% sobre as vendas.
Construa uma aplicação que receba o valor do salário base e o valor das vendas desse vendedor.
Calcule e mostre o salário final.

Passos:

Construa uma classe pública chamada **Exercicio15**
Essa classe possui o método main que deve receber o salário do funcionário e o valor total de suas vendas no mês.
Se o salário for de até R$ 700,00 (inclusive) calcular sua comissão.
Se o volume de vendas do funcionário for maior do que R$ 15.000,00 calcular sua bonificação.
Imprimir o salário atualizado do funcionário.

Restrições:

Imprimir somente o valor do salário atualizado.
Utilizar variáveis do tipo **double**.
A ordem de entrada de dados deve ser: **salário** e **vendas**.

Dicas

Inicialize as variáveis utilizadas.

Capítulo 5 - Operadores lógicos e if/else

Cap. 5 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba um número inteiro e imprima **positivo**, **negativo** ou **zero**, dependendo do valor digitado pelo usuário.

Passos:

Construa uma classe pública chamada **Exercicio1**
Essa classe possui o método main que deve imprimir somente a palavra **positivo**,**negativo** ou **zero**.

Dicas:

Lembre-se que dentro de um if/else podem existir outros if(s) e else(s).
Exemplo da estrutura do comando if:

```
1.    if (condição){
2.           comandos
3.       }else{
4.           comandos
5.       }
```

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiusculas e minusculas.
Imprimir tudo em minúsculo.

Cap. 5 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba um número inteiro e imprima **OK** se ele for maior do que 10 e menor do que 15.

Passos:

Construa uma classe pública chamada **Exercicio2**
Essa classe possui o método main que deve imprimir somente a palavra **OK** se o número digitado for maior do que 10 e menor do que 15.

Dicas:

Usar um operador lógico para construir a condição.

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiúsculas e minúsculas.

Cap. 5 - Exercício 3

Objetivo:

Construa uma aplicação em Java que receba 2 números reais e imprima o maior valor recebido.

Passos:

Construa uma classe pública chamada **Exercicio3**
Essa classe possui o método main que deve receber 2 números e imprimir o maior valor recebido.
Se ambos os valores forem iguais, imprimir um dos valores recebido.

Restrições:

Imprimir somente a informação solicitada.

Cap. 5 - Exercício 4

Objetivo:

Construa uma aplicação em Java que receba a idade de uma pessoa e imprima **MAIOR** se maior ou igual à 18 anos ou **MENOR** se menor de 18 anos.

Passos:

Construa uma classe pública chamada **Exercicio4**
Essa classe possui o método main que deve receber 1 número e imprimir exatamente **MAIOR** ou **MENOR** dependendo da idade.
Usar um tipo inteiro para receber a idade.

Restrições:

Imprimir exatamente a informação solicitada.

Cap. 5 - Exercício 5

Objetivo:

Construa uma aplicação em Java que verifique a validade da seguinte senha: 2008.
Imprima a seguinte mensagem **PERMITIDO** ou **NEGADO**.

Passos:

Construa uma classe pública chamada **Exercicio5**
Essa classe possui o método main que deve receber 1 número inteiro e imprimir exatamente **PERMITIDO** ou **NEGADO**.

Restrições:

Imprimir exatamente a informação solicitada.
Imprimir as informações em maiúsculo.

Cap. 5 - Exercício 6

Objetivo:

A média de um aluno é calculada de acordo com os seguintes pesos:

Nota	Peso
Nota 1	1
Nota 2	2
Nota 3	4

Construa uma aplicação em Java que receba as 3 notas.
Calcule a média e imprima **APROVADO** ou **REPROVADO**, considerando a média 7 para aprovação.

Passos:

Construa uma classe pública chamada **Exercicio6**
Essa classe possui o método main que deve receber 3 notas.
Calcular a média ponderada e imprimir exatamente **APROVADO** ou **REPROVADO** de acordo com a média obtida.

Restrições:

Imprimir exatamente a informação solicitada.
Utilizar variáveis do tipo **double**.
Considerar todos os valores digitados maiores que zero.

Cap. 5 - Exercício 7

Objetivo:

Uma empresa está reajustando o salário de seus funcionários da seguinte forma:

Salario	Reajuste
até R$800,00 (inclusive)	25%
>maior que R$800,00	10%

Calcular e imprimir o o salário atualizado.

Passos:

Construa uma classe pública chamada **Exercicio7**
Essa classe possui o método main que deve receber o salário do funcionário.
Calcular e imprimir o salário reajustado.

Restrições:

Imprimir exatamente a informação solicitada.
Utilizar variáveis do tipo **double**.

Cap. 5 - Exercício 8

Objetivo:

Uma fabrica está dando um bônus de R$ 150,00 para os funcionários com salário abaixo de R$ 600,00. Calcule e imprima o salário final ou mostre a mensagem SEM REAJUSTE.

Passos:

Construa uma classe pública chamada **Exercicio8**
Essa classe possui o método main que deve receber o salário e imprime o salário final ou a mensagem **SEM REAJUSTE**.

Restrições:

Imprimir exatamente a informação solicitada, respeitando os maiúsculos e minúsculos.
Utilizar variáveis do tipo **double**.

Capítulo 6 - Comandos if/else aninhados

Cap. 6 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba 1 número inteiro e imprima o resultado de acordo com a tabela a seguir:

Condição	Mensagem
< 0	MENOR
= 0	IGUAL
> 0	MAIOR

Passos:

Construa uma classe pública chamada **Exercicio1**
Essa classe possui o método main que deve receber 1 número inteiro.
e imprimir exatamente a palavra da tabela acima.

Restrições:

Imprimir somente a informação solicitada, respeitando os espaços em branco e as letras maiusculas e minusculas.

Cap. 6 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba 3 números inteiros e imprima o menor.

Passos:

Construa uma classe pública chamada **Exercicio2**
Essa classe possui o método main que deve receber 3 números inteiros e imprimir o menor.
Exemplo1: Se receber os números 5, 10, 4 imprimir 4.
Exemplo2: Se receber os números 2, 2, 4 imprimir 2.
Exemplo3: Se receber os números 7, 7, 7 imprimir 7.

Restrições:

Imprimir somente a informação solicitada.

Cap. 6 - Exercício 3

Objetivo:

Construa uma aplicação em Java que receba o salário de um funcionário e o código correspondente ao cargo.
Imprima o valor do aumento e o salário final de acordo com a tabela a seguir:

Código Cargo	Aumento
1	10%
2	15%
3	25%

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber 1º o valor do salário e depois o código do cargo.
Calcular o aumento e imprimir 1º valor do aumento e depois o salário final.

Restrições:

Imprimir somente a informação solicitada.
Utilize para cada impressão o comando **System.out.println(texto);**
Utilize variáveis do tipo **double**.
Não se preocupe com códigos do cargo inválidos.
Ordem de entrada de dados: **salário** e **código do cargo**
Ordem da saída de dados: **aumento** e **salário final**

Cap. 6 - Exercício 4

Objetivo:

Uma empresa está reajustando o salário de seus funcionários da seguinte forma:

Salário	Reajuste
<= R$400,00	35%
> R$400,00 e <= R$700,00	25%
> R$700,00	20%

Calcular e imprimir o o salário atualizado.

Passos:

Construa uma classe pública chamada **Exercicio4**.
Essa classe possui o método main que deve receber o valor do salário do funcionário.
Calcular o aumento e imprimir o salário final.

Restrições:

Imprimir somente o salário.
Utilize variáveis do tipo **double**.
Considere que o salário digitado sempre será maior que zero.

Cap. 6 - Exercício 5

Objetivo:

Construa uma aplicação em Java que receba 3 notas.
Calcule a média aritmética e imprima a mensagem de acordo com a tabela a seguir:

Média	Mensagem
>= 0 e < 5	REPROVADO
>=5 e < 7	EXAME
>= 7 e <= 10	APROVADO

Passos:

Construa uma classe pública chamada **Exercicio5**.
Essa classe possui o método main que deve receber 3 notas.
Calcular a média e imprimir a mensagem referente a tabela acima.
Se a média for menor que 0 ou maior que 10 não imprimir nada.

Restrições:

Imprimir exatamente a mensagem da tabela.
Utilize variáveis do tipo **double**.

Cap. 6 - Exercício 6

Objetivo:

Construa uma aplicação em Java que receba 2 números e a opção do usuário de acordo com a tabela a seguir:

Opção	Cálculo
1	Soma dos números
2	Multiplicação
3	Divisão do maior pelo menor
4	Subtração do maior pelo menor

Imprima o resultado da operação.
Caso a opção seja inválida imprimir **ERRO**.

Passos:

Construa uma classe pública chamada **Exercicio6**.
Essa classe possui o método main que deve receber 3 números.
O 3º número é referente a operação.
Realizar o cálculo referente a opção selecionada e imprimir o resultado.
Na operação 3 se ambos os números forem iguais imprimir 1. Considerar que o divisor nunca será zero.
Na operação 4 se ambos os números forem iguais imprimir 0.

Restrições:

Imprimir somente o resultado do cálculo ou **ERRO** se a opção for inválida.
Todos os dados recebidos devem ser do tipo **double**.

Cap. 6 - Exercício 7

Objetivo:

Construa uma aplicação em Java que receba a idade de um atleta e imprima o nome de sua categoria em letras maiúsculas:

Idade	Categoria
7 a 10	MIRIM
11 a 12	INFANTIL
13 a 14	INFANTO-JUVENIL
15 a 17	JUVENIL
18 a 36	ADULTO
acima de 36	MASTER
abaixo de 7	SEM-CATEGORIA

Passos:

Construa uma classe pública chamada **Exercicio7**.
Essa classe possui o método main que deve receber a idade do atleta (anos completos), com o formato inteiro.
Verifique e imprima a categoria do atleta em letras maiúsculas.

Restrições:

Imprimir exatamente o nome da categoria como está na tabela acima, em letras maiúsculas e hífen.
Utilize variáveis do tipo **inteiro**.

Cap. 6 - Exercício 8

Objetivo:

Construa uma aplicação em Java que receba o código da região e o preço de um produto.
Calcule e imprima o valor do produto acrescido do imposto de sua região
e o nome da região em letras maiúsculas, de acordo com a tabela abaixo:

Código da Região	Região	Imposto
1	NORTE	2.33%
2	SUL	1.5%
3	LESTE	3.5%
4	OESTE	4.2%

Se o código digitado for inválido imprima **ERRO** em letras maiúsculas.

Passos:

Construa uma classe pública chamada **Exercicio8**.
Essa classe possui o método main que deve receber 1º o código da região e depois o preço do produto.
Verifique a região e calcule o imposto.
Imprima o valor do produto com o imposto e a região em letras maiúsculas.

Restrições:

Receber 1º o código da região e depois o valor do produto.
Imprima 1º o valor do produto e depois o nome da região em letras maiúsculas.
Para imprimir cada informação utilize um **System.out.println(informação);**
Se o código digitado for inválido imprima **ERRO** em letras maiúsculas.
Para o cálculo do preço utilize variáveis do tipo **double**.
Para o código utilize uma variável do tipo **inteiro**.

Cap. 6 - Exercício 9

Objetivo:

Construa uma aplicação em Java que receba 4 notas de um aluno, calcule e imprima a mensagem de acordo com sua média.

Passos:

Construa uma classe pública chamada **Exercicio9**
Essa classe possui o método main que deve receber 4 notas.
Calcule sua média aritmética.
Imprima a seguinte mensagem de acordo com sua média:

Média	Mensagem
Entre 9.0 e 10.0 (inclusive)	MB
Entre 7.0 (inclusive) e 9.0	B
Entre 0.0 (inclusive) e 7.0	R

Restrições:

Imprimir somente a informação solicitada e em letras maiúsculas.
Utilizar variáveis do tipo **double**.
Considerar que as notas digitadas como válidas(>=0 e <=10).

Cap. 6 - Exercício 10

Enunciado alterado em: 08/11/2008.

Objetivo:

Construa uma aplicação em Java que receba o salário de um empregado, calcule e imprima o novo salário de acordo com a tabela de reajuste abaixo:

Salário	Reajuste
Até R$600,00 (Inclusive)	10%
Acima de R$600,00 e R$ 1.100,00 (Inclusive)	7%
Acima de R$1.100,00	5%

Passos:

Construa uma classe pública chamada **Exercicio10**
Essa classe possui o método main que deve receber o salário do empregado.
Faça o cálculo do reajuste baseado na tabela acima.
Imprima o salario atualizado.

Restrições:

Imprimir somente o valor do salário.
Utilizar variáveis do tipo **double**.
Se o salário digitado for inválido (negativo) não imprimir informações na tela.

Cap. 6 - Exercício 11

Objetivo:

Construa uma aplicação em Java que resolva equações do 2º grau.
A aplicação deve receber 3 variáveis referentes aos valores de a, b e c (respectivamente).
Imprimir a mensagem ERRO ou o valor das raízes, dependendo do resultado encontrado.

Passos:

Construa uma classe pública chamada **Exercicio11**
Essa classe possui o método main que deve receber o valor de **a**, **b** e **c**, nessa ordem.
ax^2 + **b**x + **c**
"a" deve ser diferente de 0, caso contrário imprimir **ERRO**.
Delta = (b * b) - (4 * a * c);
Delta < 0 : Imprima a palavra **ERRO** em letras maiúsculas.
Delta = 0 : Existe uma raiz, x = -b/(2 * a)
Delta > 0 : Existe duas raízes:
x1 = (-b + raizQuadrada(Delta))/(2 * a);
x2 = (-b - raizQuadrada(Delta))/(2 * a);
Imprimir os resultados, 1º x1 e depois x2.

Restrições:

Se Delta < 0, imprimir em letras maiúsculas a palavra **ERRO**. <%--
Se a for igual a 0, imprimir em letras maiúsculas a palavra **ERRO**.--%>
Caso Delta >= 0 imprimir somente o(s) resultado(s).

Dicas:

Para calcular a raiz quadrada utilize:

```
1.     double raiz = Math.sqrt(double numero);
```

Caso você tenha dúvida sobre as formulas apresentadas, procure na WEB mais sobre equações do segundo grau.

Cap. 6 - Exercício 12

Objetivo:

Construa uma aplicação que receba a altura e o sexo de uma pessoa e que calcule e mostre o seu peso ideal, utilizando as seguintes fórmulas:
-Homens: (72.7 * h) - 58;
-Mulheres: (62.1 * h) - 44.7

Passos:

Construa uma classe pública chamada **Exercicio12**
Essa classe possui o método main que deve receber a altura (em metros) e o sexo da pessoa (M / H), nessa ordem.
Imprimir somente o valor do peso ideal da pessoa.

Restrições:

Receber as informações na ordem solicitada.
Receber o valor da altura em metros (tipo double) e para o sexo utilizar as siglas **M** para mulher e **H** para homem (tipo String), em letras maiúsculas.
Imprimir somente o valor.

Dicas:

Para comparar variáveis do tipo String com alguma palavra ou letra utilize o código abaixo:

```
if (variável.equals("palavra")) {
//instruções
}
```

Cap. 6 - Exercício 13

Objetivo:

Construa uma aplicação que receba o código do produto e a quantidade comprada, com variáveis do tipo int.
Calcule e mostre nessa ordem (Pulando de linha):
-O preço unitário do produto seguindo a Tabela 1;
-O preço total da nota;
-O valor do desconto, seguindo a Tabela 2;
-O preço da nota com o desconto.

Tabela 1	
Código	**Preço**
1 a 10	R$ 10,00
11 a 20	R$ 15,00
21 a 30	R$ 20,00
31 a 40	R$ 30,00

Tabela 2	
Preço total da nota	% de desconto
<= R$ 250,00	5 %
Entre R$ 250,00 e R$ 500,00	10 %
>= R$ 500,00	15 %

Passos:

Construa uma classe pública chamada **Exercicio13**
Essa classe possui o método main que deve receber 1º o código do produto e depois a quantidade comprada (Nessa ordem).
Realizar os cálculos necessários e imprimir os dados na ordem do enunciado.

Restrições:

Obedecer a ordem de entrada e saída de dados. Utilize variáveis do tipo **int** para o código do produto e quantidade, e variáveis **double** para o restante dos cálculos.

Dicas:

Inicialize as variáveis.

Cap. 6 - Exercício 14

Objetivo:

Construa uma aplicação em Java que receba 3 números inteiros e imprima-os em ordem crescente.

Passos:

Construa uma classe pública chamada **Exercicio14**
Essa classe possui o método main que deve receber 3 números, ordenar e imprimir em ordem crescente.

Restrições:

Imprimir exatamente a informação solicitada.
Imprimir cada número utilizando **System.out.println();**.

Cap. 6 - Exercício 15

Objetivo:

Construa uma aplicação em Java que simule o transporte de uma carga.
A aplicação deve receber 3 dados (nessa ordem):
1-Código da carga transportada (tipo inteiro);
2-Peso da carga em toneladas (tipo double);
3-Código do estado de origem da carga (tipo inteiro).
Tabelas:

Código da Carga	Preço por Kg
1 a 5	10
6 a 10	15
11 a 15	25

Código do Estado	Imposto
1	Isento
2	7%
3	15%
4	25%

Calcule e imprima (nessa ordem):
1-O preço da carga sem impostos;
2-O valor do imposto;
3-O valor da carga + impostos.

Passos:

Construa uma classe pública chamada **Exercicio15**
Essa classe possui o método main que deve receber na respectiva ordem:
1-Código da carga transportada (tipo inteiro);
2-Peso da carga em toneladas (tipo double);
3-Código do estado de origem da carga (tipo inteiro).

Faça os cálculos necessários utilizando as tabelas acima.
Imprima os resultados na seguinte ordem e uma informação em cada linha:
O preço da carga sem impostos;
O valor do imposto;
O valor da carga + impostos.

Restrições:

Obedecer a ordem da entrada de dados.
Obedecer a ordem da saída de dados.
Considere que o usuário vai digitar somente informações válidas.

Exemplos:

Entrada de dados:
Código carga: **1**
Peso em toneladas: **10**

Código estado: **2**
Saída de dados:
100000
7000
107000

Cap. 6 - Exercício 16

Objetivo:

Construa uma aplicação em Java que receba 2 valores:
-1º o preço;
-2º a categoria (1-limpeza, 2-alimentação, 3- vestuário).
Calcule o valor do aumento, usando a tabela a seguir:

Preço	Categoria	Aumento
<= R$ 25,00	1	5%
	2	8%
	3	10%
> R$ 25,00	1	12%
	2	15%
	3	18%

Imprimir na ordem abaixo:
-1º o novo preço com o aumento;
-2º a classificação, de acordo com a tabela.

Novo Preço	Classificação
<= R$ 50,00	BARATO
Entre R$ 50,00 e R$ 120,00	NORMAL
>= R$ 120,00	CARO

Passos:

Construa uma classe pública chamada Exercicio16
Essa classe possui o método main que deve receber na respectiva ordem:
1-O preço do produto;
2-A categoria (1-limpeza, 2-alimentação, 3- vestuário).
Faça os cálculos necessários utilizando as tabelas acima.
Imprima os resultados na seguinte ordem:
1-O novo preço do produto;
2-A classificação de acordo com a tabela, em letras maiúsculas.

Restrições:

Obedecer a ordem da entrada de dados.
Obedecer a ordem da saída de dados.
Utilizar um **System.out.println** para cada saida de dados.
Considere que o usuário vai digitar somente dados válidos.

Capítulo 7 - Comando switch/case

Cap. 7 - Exercício 1

Objetivo:

Utilizando a estrutura Switch/case, construa uma aplicação em Java que receba 2 números e a opção do usuário de acordo com a tabela a seguir:

Opção	Cálculo
1	Soma dos números
2	Multiplicação
3	Divisão do maior pelo menor
4	Subtração do maior pelo menor

Imprima o resultado da operação.
Caso a opção seja inválida imprimir **ERRO** em letras maiúsculas.

Passos:

Construa uma classe pública chamada **Exercicio1**.
Essa classe possui o método main que deve receber 3 números.
O 3º número é referente a operação.
Realizar o cálculo referente a opção selecionada e imprimir o resultado.

Restrições:

Imprimir somente o resultado do cálculo ou **ERRO** se a opção for inválida.
A ordem de entrada dos dados é: número1 (double), número2 (double) e operação (inteiro)
Na opção 3 e 4 verificar qual numero é o maior antes de efetuar o cálculo.
Na opção 3 considerar que o divisor sempre será diferente de zero.

Dica:

Para verificar qual número é maior, na opção 3 e 4, utilize **if**.

Exemplo da estrura do Switch

```
switch(variavel){
    case 1: instruções; break;
    case 2: instruções; break;
    case 3: instruções; break;
```

```
        default: instruções;
    }
```

Cap. 7 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba o valor de um produto e o código de seu aumento, de acordo com a tabela abaixo:

Código	Aumento em %
1	5
2	7
3	10
4	15
5	17

Calcule o aumento e imprima final do produto.
Caso a opção seja inválida imprimir **ERRO** em letras maiúsculas.

Passos:

Construa uma classe pública chamada **Exercicio2**.
Essa classe possui o método main que deve receber 1º o valor do produto e depois o código de seu aumento (nessa ordem!).
Calcular o aumento e imprimir o valor final do produto.

Restrições:

Imprimir somente o resultado do cálculo ou **ERRO** se a opção for inválida.

Cap. 7 - Exercício 3

Objetivo:

Construa uma aplicação em Java que implemente o jogo pedra-papel-tesoura.
Procure utilizar somente a estrutura SWITH-CASE.

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber 1º a opção do Jogador 1 e depois a opção do Jogador 2 (nessa ordem!).
De acordo com o resultado imprima **GANHOU**, **EMPATOU** ou **PERDEU** para o Jogador 1.
Dentro do SWITCH-CASE se alguma opção for inválida imprima **ERRO**.

Restrições:

Primeiro receba todos as entradas para depois fazer as verificações.
Considere **1** para Pedra, **2** para Papel e **3** para Tesoura.
Caso um outro valor seja identificado imprima **ERRO**.
Obs: pedra ganha de tesoura. tesoura ganha de papel e papel ganha de pedra.
Imprimir exatamente a informação solicitada, sem espaços em branco!

Dica:

Dentro de um **case** pode existir qualquer tipo de instrução, até outro comando **switch**.

Cap. 7 - Exercício 4

Objetivo:

Construa uma aplicação em Java que leia uma data na seguinte ordem e obedecendo o formato:
1-Dia da semana (um inteiro entre 1-7, inclusive 1 e 7);
2-Dia (um inteiro entre 1-31, inclusive 1 e 31);
3-Mês (um inteiro entre 1-12, inclusive 1 e 12);
4-Ano (inteiro entre 1900 e 9999, inclusive 1900 e 9999).
Converter para o formato especificado abaixo:

Dia da Semana	
1	Segunda-feira
2	Terca-feira
3	Quarta-feira
4	Quinta-feira
5	Sexta-feira
6	Sabado
7	Domingo

Dia do Mês	
1	Janeiro
2	Fevereiro
3	Marco
4	Abril
5	Maio
6	Junho
7	Julho
8	Agosto
9	Setembro
10	Outubro
11	Novembro
12	Dezembro

Caso algum dado seja inválido imprima **ERRO**.
Exemplo da tela de saída:
-Entre com o dia da semana(1-7): **4**
-Entre com o dia do mês (1-31): **23**
-Entre com o mês (1-12): **8**
-Entre com o ano: **1997**
Quinta-feira, dia 23 de Agosto de 1997

Passos:

1-Construa uma classe pública chamada **Exercicio4**.
2-Essa classe possui o método main que deve receber todas as entradas descritas acima e obedecendo a mesma ordem.
3-Utilize a estrutura SWITH-CASE para verificar o dia da semana e o mês.
4-Utilize a estrutura IF para verificar o dia e o ano.
5-Por último utilize o IF para imprimir a data ou a mensagem de erro.

Restrições:

-Primeiro receba todos as entradas para depois fazer as verificações.
-Imprima a data de forma que obedeça o padrão do exemplo acima.
-O dia da semana e o mês devem ter a 1ª letra maiúscula e sem acentos.

-Atenção aos espaços e virgulas.
-Não se preocupe nesse exercício com ano bissexto.

Dicas:

-Utilize uma String para armazenar a mensagem de erro, iniciando a variável com **null**.
Exemplo:
String erro = null;

-Utilize uma String para armazenar a data.
-Faça as verificações na mesma ordem da entrada de dados,
dessa forma você pode adicionar os dados na String obedecendo o formato pedido.
Exemplo:

```
String data;

data = "Terca-feira";
data = data + ", dia 5";
data = data + " de agosto";
data = data + " de 2001";
```

-No IF final se a String erro for igual a **null** imprima a data,caso contrário imprima a String erro.
Erro igual a **null** significa que o valor essa variável não armazena nenhuma informação.

```
if(erro == null)
   System.out.println(data);
else
   System.out.println(erro);
```

Cap. 7 - Exercício 5

Objetivo:

Construa uma aplicação em Java que calcula a área de um círculo, quadrado ou triângulo.

Passos:

Construa uma classe pública chamada **Exercicio5**.
Essa classe possui o método main que deve receber 1 número inteiro
referente ao cálculo da área de uma das seguintes figuras:

1-Circulo (receber o valor do raio (double), utilizar o valor de PI = 3.14);
2-Quadrado (receber o valor do lado (double));
3-Triângulo (receber o valor da base e da altura (tipos double)).

Exemplo de execução 1:
Calcular a área da figura: **1**
Digite o valor do raio: **1**
3.14

Exemplo de execução 2:
Calcular a área da figura: **2**
Digite o valor do lado: **4.5**
20.25

Exemplo de execução 3:
Calcular a área da figura: **3**
Digite o valor da base: **6**
Digite o valor da altura: **3**
9

Restrições:

Utilize variáveis dos tipos indicados no enunciado.
Caso a opção seja inválida imprima ERRO.
Para o exercício o valor PI deve ser obrigatoriamente **3.14**.
Imprimir somente o resultado.

Capítulo 8 - Comando while

Cap. 8 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba a quantidade de funcionários de uma empresa.
Com base na quantidade de funcionários receba o salário de cada um.
Calcule e imprima a média.

Passos:

Construa uma classe pública chamada **Exercicio1**.
Essa classe possui o método main que deve receber 1º a quantidade de funcionários (tipo int).
A partir da quantidade de funcionários, receber o salário de cada um (tipo double).
Calcular e imprimir a média.

Restrições:

Imprimir somente o resultado do cálculo.

Exemplo da estrutura while

```
while (condição){
instruções;
}
```

Cap. 8 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba o salário de 2 funcionários.
O funcionário 1 aplicará seu salário integralmente na poupança,
que está rendendo 1.5% ao mês.
O funcionário 2 aplicará seu salário integralmente no fundo de renda fixa,
que está rendendo 4% ao mês.
Calcule e mostre a quantidade de meses necessários para que o valor
pertencente ao funcionário 2 ultrapasse o valor do funcionário 1.

Passos:

Construa uma classe pública chamada **Exercicio2**.
Essa classe possui o método main que deve receber 1º o salário do funcionário 1 (tipo double)
e depois o salário do funcionário 2 (tipo double).
Utilize o laço WHILE para calcular a quantidade de meses.
Imprima somente o resultado

Restrições:

Imprimir somente o resultado do cálculo. Utilize variáveis do tipo double.

Exemplo

Salário do funcionário 1: **500**
Salário do funcionário 2: **200**
38

Cap. 8 - Exercício 3

Objetivo:

Construa uma aplicação em Java que leia dois números inteiros,
o primeiro é o valor inicial de um contador
e o segundo é o valor final do contador.
Usando o comando WHILE escreva na tela uma contagem que comece
no primeiro número lido, escreva os números seguintes colocando
apenas um número em cada nova linha da tela, até chegar ao
valor final indicado.

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber 1º o valor inicial
e depois o valor final.
Utilize a estrutura WHILE para imprimir a sequência numérica.
Imprima 1 número por vez.

Restrições:

Imprimir 1 número por vez.

Exemplo

Entre com o número inicial: **5**
Entre com o número final: **9**
5
6
7
8
9

Exemplo da tela de saída: Entre com o número inicial: 5 Entre com o número final: 9 5 6 7 8 9

Cap. 8 - Exercício 4

Objetivo:

Construa uma aplicação em Java que calcule todos os números inteiros divisíveis por um certo valor inteiro indicado pelo usuário (o resto da divisão por este número deve ser igual a 0), compreendidos em um intervalo também especificado pelo usuário.
O usuário deve entrar com um primeiro valor correspondente ao divisor e
após ele vai fornecer o valor inicial do intervalo, seguido do valor final deste intervalo.

Passos:

Construa uma classe pública chamada **Exercicio4**.
Essa classe possui o método main que deve receber 1º o valor do divisor,
depois o valor do inicio do intervalo(inclusive) e depois o valor final do intervalo(inclusive).
Verifique se o divisor = 0, se sim imprima **"ERRO"**, se não
utilize a estrutura WHILE para imprimir a sequência numérica.
Imprima 1 número por vez.

Restrições:

Imprimir 1 número por vez.
Caso o divisor seja = 0 imprimir exatamente a palavra **ERRO**,
em letras maiúsculas e sem espaços.

Exemplo

Entre com o valor do divisor: **3**
Início do intervalo: **18**
Final do intervalo: **29**
18
21
24
27

Dicas

Para verificar se um número possui resto utilize a seguinte expressão:
numero % divisor;
Se o valor da expressão for = 0 significa que a divisão foi perfeita, sem restos.

Cap. 8 - Exercício 5

Objetivo:

Construa uma aplicação em Java que recebe e soma vários números inteiros.
Quando receber o número 0 imprima o resultado e encerre a aplicação.

Passos:

Construa uma classe pública chamada **Exercicio5**.
Essa classe possui o método main que deve receber números inteiros
até que o número 0 seja recebido.
Imprima o resultado da soma e encerre a aplicação.

Restrições:

Imprimir somente o resultado.

Capítulo 9 - Comando do-while

Cap. 9 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba a quantidade de notas de um aluno.
Com base na quantidade de notas receba o valor de cada uma.
Calcule e imprima a média.

Passos:

Construa uma classe pública chamada **Exercicio1**.
Essa classe possui o método main que deve receber 1º a quantidade de notas do aluno.
A partir da quantidade de notas, utilize o laço DO/WHILE para receber o valor de cada nota.
Calcular e imprimir a média.

Restrições:

Imprimir somente o resultado do cálculo.

Exemplo da estrutura while

```
do{
instruções;
}while (condição);
```

Cap. 9 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba uma sequência de números inteiros até que o número 0 seja recebido.
Ao final imprimir a soma dos números.

Passos:

Construa uma classe pública chamada **Exercicio2**.
Essa classe possui o método main que deve receber os números utilizando o laço DO/WHILE.
Receba 1 número por vez até que seja recebido o número 0.
Imprimir a soma dos números.

Restrições:

Imprimir somente o resultado do cálculo.

Cap. 9 - Exercício 3

Objetivo:

Construa uma aplicação em Java que recebe e soma vários
números inteiros enquanto não digitarem 0.
Some os números pares e ímpares separados.
Imprima a soma dos pares e depois a soma dos ímpares.

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber números inteiros
até que o número 0 seja digitado.
Some os números pares e ímpares separados.
Ao final imprima primeiro a soma dos pares e depois a soma dos ímpares.

Restrições:

Imprimir somente o resultado.

Dica:

Para verificar se um número é par utilize a seguinte expressão:
numero % 2;
Se o valor da expressão for = 0 significa que o número é par.

Cap. 9 - Exercício 4

Objetivo:

Construa uma aplicação em Java que recebe uma quantidade
indeterminada de números inteiros e conte quantos deles
estão nos seguintes intervalos:
[1-25], [26-50], [51-75] e [76-100].
A entrada de dados deverá terminar quando for lido o número 0.

Passos:

Construa uma classe pública chamada **Exercicio4**.
Essa classe possui o método main que deve receber números inteiros
até que o número 0 seja digitado.
Some os números de acordo com os intervalos.
Ao final imprima a quantidade de números de cada intervalo
obedecendo a ordem dada.

Restrições:

Imprimir cada um dos 4 resultados utilizando 1 **System.out.println();**

Cap. 9 - Exercício 5

Objetivo:

Construa uma aplicação em Java que receba o valor do salário mínimo
e uma sequência de números contendo 1º a quantidade de quilowatts gasta
por consumidor e 2º o tipo de consumidor:
1-Residencial;
2-Comercial;
3-Industrial.
Calcule e mostre:
-O preço do quilowatt, considere o valor sendo igual a 1/8 do salário mínimo;
-O valor a ser pago por cada consumidor com base no imposto da tabela abaixo:

Tipo	Imposto (%)
1	5
2	10
3	15

Sabendo-se que o valor a ser pago = (quantidade KW * preço KW) + (impostos).
-A quantidade de consumidores que pagam entre R$500,00 e R$ 1.000,00.
Finalize a aplicação quando a quantidade de quilowatts for igual a 0.

Passos:

Construa uma classe pública chamada **Exercicio5**.
Essa classe possui o método main que deve receber o valor do salário mínimo,
calcular e imprimir o valor do quilowatt.
Depois utilize o laço Do-While para receber os outros valores, calcular e imprimir
o total gasto de cada consumidor.
Quando o usuário digitar **0** para o valor do quilowatt saia do laço e
imprima a quantidade de consumidores que pagam entre R$500,00 e R$1.000,00.

Restrições:

Imprimir cada resultado utilizando **System.out.println();**

Capítulo 10 - Comando for

Cap. 10 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba um número inteiro inicial e final
e utilizando o laço FOR calcule a soma de todos os números inteiros desse intervalo.

Passos:

Construa uma classe pública chamada **Exercicio1**.
Essa classe possui o método main que deve receber 1 número inteiro inicial e depois
um número inteiro final.
Utilize o laço FOR para o cálculo.
Ao final imprima somente o resultado.

Restrições:

Imprimir somente o resultado do cálculo.

Exemplo:

Número inicial: **5**
Número final: **10**
Resultado: 5+6+7+8+9+10= **45**

Exemplo da estrutura for

```
for (int x = 0; x < 10; x++){
instruções;
}
```

Cap. 10 - Exercício 2

Objetivo:

Construa uma aplicação em Java que calcule o fatorial de um número.

Passos:

Construa uma classe pública chamada **Exercicio2**.
Essa classe possui o método main que deve receber 1 número inteiro
para o cálculo do fatorial.
Utilize o laço FOR para o cálculo.
Ao final imprima somente o resultado.
Não se preocupe com dados inválidos.

Restrições:

Imprimir somente o resultado do cálculo.

Dicas:

O fatorial de 4 é: 1*2*3*4 = 24
O fatorial de 9 é: 1*2*3*4*5*6*7*8*9=362880

Exemplo da estrutura for

```
for (int x = 0; x < 10; x++){
instruções;
}
```

Cap. 10 - Exercício 3

Objetivo:

Com base no exercicio anterior, construa uma aplicação em JAVA
que receba um número inteiro e calcule seu fatorial.
Antes de efetuar o cálculo verifique se o número é válido. (Números maiores ou igual a zero são válidos
)

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber 1 número inteiro
para o cálculo do fatorial.
Verifique se o número é válido, se não for imprima a palavra **ERRO**.
Se for válido utilize o laço FOR para o cálculo.
Ao final imprima somente o resultado.

Dicas

- Se digitar 4 imprime: 24
- Se digitar 0 imprime: 1
- Se digitar -10 imprime: ERRO

Restrições:

Imprimir somente o resultado do cálculo ou a palavra **ERRO** em letras maiúsculas.

Cap. 10 - Exercício 4

Objetivo:

Construa uma aplicação em JAVA que receba um número inteiro,
calcule e imprima todos os seus divisores inteiros.

Passos:

Construa uma classe pública chamada **Exercicio4**.
Essa classe possui o método main que deve receber 1 número inteiro
para o cálculo dos divisores.
Utilize o laço FOR para o cálculo.
O 1º número inteiro divisivel é o número 1 e o último é o próprio número.

Restrições:

Imprimir somente o resultado do cálculo.
Imprima cada número em uma linha.

Dicas:

Para verificar se um número é divisivel a expressão **numero%divisor = 0**
deve ser verdadeira.

Exemplo:

Número: **20**
Divisores: **1**
2
4
5
10
20

Cap. 10 - Exercício 5

Objetivo:

Construa uma aplicação em Java que calcule o salário atual de um funcionário.
Sabe-se que o funcionário foi contratado em 2000 com o salário inicial digitado pelo usuário.
Em 2001 o aumento foi de 1% e nos anos seguintes o aumento foi o dobro do percentual do anterior.
Calcule e imprima o salário de 2008.

Passos:

Construa uma classe pública chamada **Exercicio5**.
Essa classe possui o método main que deve receber o salário do funcionário.
Utilize o laço FOR para o cálculo dos aumentos de 2001 até 2008 (inclusive).
Ao final imprima somente o valor do salário atualizado.

Restrições:

Utilize variáveis do tipo **double**.

Cap. 10 - Exercício 6

Objetivo:

Construa uma aplicação em Java que receba (nessa ordem) o sexo (M ou F),
a altura (em cm) e o peso de 10 pessoas.
Calcule e imprima nessa ordem
1)O maior peso;
2)O menor peso;
3)A média das alturas;
4)O número de homens;
5)O sexo da pessoa mais pesada.

Passos:

Construa uma classe pública chamada **Exercicio6**.
Essa classe possui o método main que deve utilizar o laço FOR
para receber as 3 informações das 10 pessoas.
Faça os cálculos necessários e imprima as informações acima
na mesma ordem.

Restrições:

O peso e altura devem ser do tipo **double**.
Para o sexo masculino utilize **M** e para o feminino **F**,
em letras maiúsculas.
Imprimir cada informação utilizando um **System.out.println(informação);**
Não é necessário verificar se as infomações de entrada são válidas.

Dicas:

Para verificar a String sexo utilize a expressão **sexo.equals("M")**.
Nesse caso a expressão retorna **true** se a String sexo é igual a letra M.

Cap. 10 - Exercício 7

Objetivo:

Construa uma aplicação em Java que receba 3 notas de 5 alunos, calcule e mostre nessa ordem:
1)A média aritmética das notas de cada aluno;
2)A mensagem da tabela:

Média	Mensagem
< 5	REPROVADO
>= 5 e < 7	EXAME
>= 7	APROVADO

3)O total de alunos reprovados;
4)O total de alunos de exame;
5)O total de alunos aprovados;
6)A média da classe.

Passos:

Construa uma classe pública chamada **Exercicio7**.
Essa classe possui o método main que deve utilizar o laço FOR para receber as 3 notas dos 5 alunos.
Ao receber as 3 notas, calcule e imprima a média do aluno e a mensagem correspondente a média, em letras maiúsculas.
Finalizado o laço imprima a quantidade de alunos reprovados, de exame, aprovados e a média dos alunos.

Restrições:

As notas devem ser do tipo double.
Imprimir a média de cada aluno assim que receber as 3 notas, assim como a mensagem correspondente.
As mensagens devem ser sempre em letras maiúsculas e sem espaços em branco.
Siga a ordem de saída apresentada.
Imprimir cada informação utilizando um **System.out.println(informação);**

Dicas:

Utilize apenas 1 laço FOR para receber as notas dos 5 alunos.
Receba as 3 notas e imprima a média e a mensagem em cada volta do laço.
Finalizado o laço imprima o restante das informações.

Capítulo 11 - Laços Aninhados

Cap. 11 - Exercício 1

Objetivo:

Construa uma aplicação em Java que leia um número. Esse número indica quantos valores inteiros e positivos devem ser lidos.
Para cada número lido imprima o seu fatorial.

Passos:

Construa uma classe pública chamada **Exercicio1**.
Essa classe possui o método main que deve receber um número.
Esse número indica quantos números devem ser recebidos.
Utilize o laço FOR para receber esses números.
Dentro do laço FOR utilize mais um outro laço FOR para calcular o fatorial.
Imprima o resultado e receba o próximo número.

Restrições:

Imprimir somente o resultado do cálculo.

Cap. 11 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba um número indeterminado de pares de valor [num1, num2], todos inteiros, um par de cada vez.
Para cada par digitado calcule e mostre a soma de todos os números inteiros entre eles(inclusive).
A digitação dos pares termina quando num1 for maior ou igual a num2.

Passos:

Construa uma classe pública chamada **Exercicio2**.
Essa classe possui o método main que deve receber o par de números.
Utilize o laço While para verificar se o par de números é válido.
Utilize o laço FOR para calcular a soma dos números.
Ao final do laço FOR imprima a soma do par.
Em seguida receba mais um par de dados.

Restrições:

Imprimir somente o resultado do cálculo.

Dicas:

Exemplo de Execução:
Se Num1=10 e Num2=11, Imprime 21 (10+11).
Se Num1=1 e Num2=5, Imprime 15 (1+2+3+4+5).

Cap. 11 - Exercício 3

Objetivo:

Construa uma aplicação em Java que leia um número indeterminado de valores para Num, todos inteiros e positivos, um de cada vez.
Se Num for par, verifique quantos divisores possui.
Se Num for ímpar, calcule a soma dos números inteiros de 1 até Num (Num não deve entrar nos cálculos).
Imprima o resultado e finalize a entrada de dados quando Num = 0 ou negativo.

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber um número.
Utilize o laço While para verificar se o número é válido.
Utilize o laço FOR para calcular a soma dos números ímpares ou
para contar quantos divisores possui o número par.
Imprima o resultado e receba o próximo número.

Restrições:

Imprimir somente o resultado do cálculo.
Utilize a função módulo para verificar se o número é divisível.

Cap. 11 - Exercício 4

Objetivo:

Em um campeonato de vôlei de praia existem 6 duplas.
Construa uma aplicação que receba, nessa ordem: a idade, peso e altura(em cm) de cada um dos jogadores.
Calcule e mostre (nessa ordem):
1-As médias das idades de cada dupla;
2-A quantidade de jogadores com idade inferior a 18 anos;
3-A média das alturas de todos os jogadores;
4-A porcentagem de jogadores com mais de 80Kg.

Passos:

Construa uma classe pública chamada **Exercicio4**.
Essa classe possui o método main.
Utilize dois laços FOR, um dentro do outro.
O FOR externo percorre as 6 duplas e o interno recebe os dados de cada jogador da dupla.
1º receba a idade, depois o peso e por último a altura.
Após receber os dados de cada dupla imprima a média das idades.
Por fim imprima a quantidade de jogadores com idade inferior a 18 anos, a média das alturas de todos os jogadores e a porcentagem de jogadores com mais de 80Kg.

Restrições:

Imprimir somente o resultado do cálculo.
Seguir a ordem de entrada e saída de dados.

Cap. 11 - Exercício 5

Objetivo:

Uma empresa contratou 5 funcionários temporários.
De acordo com o valor das vendas mensais, os funcionários adquirem pontos que determinarão seus salários ao final de cada mês.
Sabe-se que esses funcionários trabalharão nos meses de novembro a janeiro do ano seguinte.
Construa uma aplicação em Java que:
1) Leia as pontuações dos três meses de cada funcionário;
2) Calcule e imprima a pontuação total de cada funcionário nos três meses;
3) Calcule e imprima a média das pontuações de cada funcionário nos três meses;
4) Ao final determine e imprima a maior pontuação atingida entre todos os funcionários nos três meses.

Passos:

Construa uma classe pública chamada **Exercicio5**.
Essa classe possui o método main.
Utilize dois laços FOR, um dentro do outro.
O FOR externo percorre os 5 funcionários e o interno recebe os dados de cada mês do funcionário.
Imprima 1º a pontuação total de cada funcionário, depois a média e por último (fora do laço) a maior pontuação encontrada.

Restrições:

Imprimir somente o resultado do cálculo.
Seguir a ordem de saída de dados.

Capítulo 12 - Métodos Estáticos

Cap. 12 - Exercício 1

Objetivo:

Construa uma aplicação em Java que imprima a frase **Aprenda Java**
utilizando o método **imprimir()**, que deverá ser criado.

Passos:

Construa uma classe pública chamada **Exercicio1**.
Essa classe possui o método main que deve chamar o método **imprimir()**.
O método **public static void imprimir()**, a ser criado deve imprimir exatamente a frase
Aprenda Java, respeitando as letras maiúsculas e espaços.

Restrições:

Criar o método solicitado como sendo do tipo void.
Imprimir exatamente a frase solicitada utilizando o método indicado.

Cap. 12 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba um número inteiro, calcule e mostre seu fatorial.
Crie um método chamado **fatorial(int)** para auxiliar o cálculo.

Passos:

Construa uma classe pública chamada **Exercicio2**.
Essa classe possui o método main que deve receber um número inteiro e chamar
o método **fatorial(int)** passando o número recebido.
O método **fatorial(int)** deve calcular e retornar o resultado.
O método principal deve imprimir o valor retornado.

Restrições:

Criar o método solicitado com o valor de retorno do tipo int.
O parâmetro do método deve ser do tipo int.
O método também deve ser publico e estático.
Imprimir no método main somente o resultado do método fatorial.
Utilize variáveis do tipo **int**.

Cap. 12 - Exercício 3

Objetivo:

Construa uma aplicação em Java que receba um número inteiro e imprima **false** se for negativo ou **true** se positivo (>= 0).
Crie um método chamado **verifica(int)** para auxiliar o cálculo.

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber um número inteiro e chamar o método **verifica(int)** passando o número recebido.
O método **public static boolean verifica(int)** deve identificar se o número é negativo (retornar false) ou >= 0 (retornar true).
O método principal deve imprimir o valor retornado.

Restrições:

Criar o método solicitado com o retorno do tipo boolean.
O parâmetro do método deve ser do tipo int.
Imprimir no método main somente o resultado do método verifica.

Cap. 12 - Exercício 4

Alterado em 19/01/2009.
Alterado em 20/01/2009.
Alterado em 11/05/2016.

Objetivo:

Construa uma aplicação em Java que receba dois números inteiros e imprima a soma de todos os número, inclusive dos digitados.
O 1º número deve ser menor ou igual ao 2º número digitado.
Crie um método chamado somar(int, int) para realizar o cálculo.

Passos:

Construa uma classe pública chamada **Exercicio4**.
Essa classe possui o método main.
Crie o método **public static int somar(int, int)** para somar todos os números do intervalo, inclusive os digitados, com o tipo de retorno do tipo int.
No método main receba 2 números inteiros. Repita esse processo até que o 1º número seja menor ou igual ao 2º, utilize o laço do..while para a repetição.
Somente quando a condição acima for verdadeira chame o método **somar(int, int)** e imprima o resultado retornado por ele.

Restrições:

Criar o método solicitado com o retorno do tipo int.
Os 2 parâmetros de entrada do método devem ser do tipo int.
Imprimir no método main somente o resultado do método somar.
Faça a validação da entrada de dados.
Utilize variáveis do tipo **int**.

Cap. 12 - Exercício 5

Objetivo:

Construa uma aplicação em Java que receba 3 notas de um aluno e uma letra como parâmetros e chame o método calcula(double, double, double, String).
Se a letra for A o metodo calcula a média aritmética das notas,
se for P calcula a média ponderada com pesos 5, 3 e 2.
A média calculada deve ser devolvida ao main para ser impressa.

Passos:

Construa uma classe pública chamada **Exercicio5**.
Essa classe possui o método main.
Receba 3 números double referente as notas e uma letra referente ao tipo de média.
Chame o método **calcula(double, double, double, String)**.
O método **public static double calcula(double n1, double n2, double n3, String op)** deve ser criado e seu tipo de retorno deve ser um double.
O método deve calcular a média Aritmética se a letra for **A** ou Ponderada se a letra for **P**.
Imprima o resultado no método main.

Restrições:

Criar o método solicitado de acordo com os parâmetros indicados e com o retorno do tipo double.
A letra (A ou P) ndicando o tipo de cálculo deve ser maiúscula.
Para o tipo de cálculo considere que o usuário sempre digitará entradas válidas.
Imprimir no método main somente o resultado do método calcula().
Utilize o método **equals** para verificar a letra digitada.

Cap. 12 - Exercício 6

Objetivo:

Construa uma aplicação em Java que receba 3 notas de um aluno e chame o método media(double, double, double).
O método deve calcular a média ponderada com os seguintes pesos:
nota1 peso 1, nota2 peso 2 e nota 3 peso4.
Se a média for < 5 retorne **REPROVADO**, se for >= 5 e < 7 retorne **EXAME** e se for >= 7 retorne **APROVADO**.

Passos:

Construa uma classe pública chamada **Exercicio6**.
Essa classe possui o método main.
Receba 3 números double referente as notas.
Chame o método **media(double, double, double)**.
O método **public static String media(double n1, double n2, double n3)** deve ser criado e seu tipo de retorno deve ser String.
O método deve calcular a média Ponderada das notas de acordo com os pesos acima.
Retorne a situação do aluno de acordo com as condições acima.
A situação do aluno deve ser impressa no método main.

Restrições:

Criar o método solicitado de acordo com os parâmetros indicados e com o retorno do tipo String.
Para o tipo de cálculo considere que o usuário sempre digitará entradas válidas.
Imprimir no método main somente o resultado do método media(), que deve ser sempre em letras maiúsculas.

Capítulo 13 - Arrays Numéricas

Cap. 13 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba um vetor com 10 números inteiros.
Verifique a existência de números iguais a 15, mostrando os índices em que eles apareceram.

Passos:

Construa uma classe pública chamada **Exercicio1**.
Essa classe possui o método main que deve carregar um vetor com 10 números inteiros.
Verifique a existência de números iguais a 15, imprimindo seus respectivos índices no vetor.

Restrições:

O índice de um vetor começa sempre pela posição 0.

Dicas:

Utilize o laço FOR para receber os dados e para efetuar os cálculos.

Exemplo

Vetor X	15	3	5	7	9	11	13	15	17	19

Impressão do resultado:
0
7

Cap. 13 - Exercício 2

Alterado em 22/01/2009.
Alterado em 27/01/2009.

Objetivo:

Construa uma aplicação em Java que receba e armazene 9 números inteiros em um vetor, calcule e mostre os números primos e seus respectivos índices.

Passos:

Construa uma classe pública chamada **Exercicio2**.
Essa classe possui o método main que deve receber 9 números inteiros e armazenar em um vetor.
Calcule e imprima os números primos do vetor e também seus seus respectivos índices.

Restrições:

Utilize 1 System.out.println() para imprimir cada informação.
O indice de um vetor começa sempre pela posição 0.

Dicas:

Utilize o laço FOR para receber os dados e para efetuar os cálculos.

Exemplo

Valores de entrada	1	2	3	4	5	6	7	8	9
Índices	0	1	2	3	4	5	6	7	8

Dessa forma:
2 é primo e o índice no vetor é 1.
3 é primo e o índice no vetor é 2.
5 é primo e o índice no vetor é 4.
7 é primo e o índice no vetor é 6.

Resultado Esperado (Ordem: Primo e Índice)
2
1
3
2
5
4
7

```
6
```

Cap. 13 - Exercício 3

Objetivo:

Construa uma aplicação em Java receba um vetor com 10 números inteiros digitados pelo usuário.
Calcule e imprima os números maiores do que 25 e suas posições.
Imprimir a mensagem "VAZIO" se não existir nenhum número.

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber um vetor com 10 números inteiros digitados pelo usuário.
Imprima SOMENTE os números maiores do que 25 e suas posições (Cada um em uma linha).
Se não existir nenhum número imprima **VAZIO** em letras maiúsculas.

Restrições:

A palavra VAZIO deve ser impressa em letras maiúsculas.

Exemplo

Entrada: [10, 15, 20, 25, 30, 35, 40, 45, 50, 55]
Saída:

```
30
4
35
5
40
6
45
7
50
8
55
9
```

Cap. 13 - Exercício 4

Alterado em 23/01/2009.

Objetivo:

Construa uma aplicação em Java que receba 2 vetores (X e Y) com 10 números inteiros cada um.
Calcule o vetor resultante da diferença entre X e Y.
O vetor resultante deve conter todos os elementos de X que não existam em Y.
Imprima o resultado.

Passos:

Construa uma classe pública chamada **Exercicio4**.
Essa classe possui o método main que deve receber 2 vetores (X e Y) com 10 números inteiros cada.
Receber primeiro todos os valores para o vetor X.
Após receber todos os valores para X, receber todos os valores para o vetor Y.
Calcule o vetor resultante da diferença entre X e Y e imprima o resultado.

Restrições:

O indice de um vetor começa sempre pela posição 0.
Inicialize os vetores com 0.
Receber todos os elementos de X e depois todos os elementos de Y.

Exemplo

Vetor X	3	5	4	2	1	6	8	7	11	9

Vetor Y	2	1	5	12	3	0	-1	4	7	6

Vetor Resultante	8	11	9	0	0	0	0	0	0	0

Impressão do vetor resultante:
8
11
9
0
0
0
0
0
0
0
0
0

Dicas:

Utilize o laço FOR para receber os dados e para efetuar os cálculos.
Quando a intenção é imprimir somente os valores do vetor,
pode-se usar o laço FOR da seguinte maneira:

```
for (int valor : vetor)
System.out.println(valor);
```

Dessa forma o laço for percorre o vetor inteiro passando seu conteúdo
para a variável valor, que nesse caso é impressa.

Cap. 13 - Exercício 5

Alterado em 28/01/2009

Objetivo:

Construa uma aplicação em Java que receba 2 vetores com 5 números inteiros cada.
Faça a intercalação dos 2 vetores em um vetor resultante e imprima seus valores.
Ordene o vetor de forma decrescente e imprima seus valores.

Passos:

Construa uma classe pública chamada **Exercicio5**.
Essa classe possui o método main que deve receber 5 números inteiros referentes ao vetor1 e 5 números inteiros referentes ao vetor2.
Receber os vetores separadamente, primeiro o vetor1 e depois o vetor2.
Faça a intercalação dos 2 vetores em um 3º vetor.
Imprima seus valores.
Faça a ordenação decrescente e imprima os valores.

Restrições:

O indice de um vetor começa sempre pela posição 0.
Receber os vetores separadamente, primeiro o vetor1 e depois o vetor2.

Dicas:

Utilize o laço FOR para receber os dados e para efetuar os cálculos.

Exemplo

Vetor 1	1	3	5	7	9

Vetor 2	2	4	6	8	10

Vetor Resultante	1	2	3	4	5	6	7	8	9	10

Impressão do vetor resultante:
1
2
3
4
5
6
7
8

```
9
10
Impressão do vetor ordenado:
10
9
8
7
6
5
4
3
2
1
```

Cap. 13 - Exercício 6

Alterado em 27/01/2009.

Objetivo:

Construa uma aplicação em Java que receba 2 vetores (X e Y) com 6 números inteiros cada um.
Calcule o vetor resultante da união de X e Y.
O vetor resultante deve conter todos os elementos de X e os elementos de Y que não estejam em X.

Passos:

Construa uma classe pública chamada **Exercicio6**.
Essa classe possui o método main que deve receber 2 vetores (X e Y) com 6 números inteiros cada.
Receber primeiro o vetor X e depois o vetor Y.
Calcule o vetor resultante da união de X e Y e imprima o resultado.

Restrições:

O indice de um vetor começa sempre pela posição 0.
Inicialize os vetores com 0.
Receber primeiro o vetor X e depois o vetor Y.

Dicas:

Utilize o laço FOR para receber os dados e para efetuar os cálculos.

Exemplo

Vetor X	1	2	3	4	5	6

Vetor Y	4	5	6	7	8	9

Vetor Resultante	1	2	3	4	5	6	7	8	9	0	0	0

Impressão do vetor resultante:
1
2
3
4
5
6
7
8
9
0

```
0
0
```

Capítulo 14 - Arrays de Strings

Cap. 14 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba 5 nomes e armazene em um vetor.
Troque o 1º nome pelo último e o 2º pelo penúltimo.
Imprima o resultado, cada nome em uma linha.

Passos:

Construa uma classe pública chamada **Exercicio1**.
Essa classe possui o método main que deve receber 5 nomes e armazenar em um vetor.
Troque a posição dos nomes, o 1º pelo último e o 2º pelo penúltimo.
Imprima o vetor resultante, cada nome em uma linha.

Restrições:

Utilize 1 System.out.println() para imprimir cada nome.

Exemplo

Entrada: Felipe, Luiz, Camila, Anderson, Flavio
Resultado:
Flavio
Anderson
Camila
Luiz
Felipe

Cap. 14 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba uma data no formato numérico
e imprima a data por extenso.

Indice do vetor	Dia da Semana
[0]	Segunda-feira
[1]	Terça-feira
[2]	Quarta-feira
[3]	Quinta-feira
[4]	Sexta-feira
[5]	Sábado
[6]	Domingo

Indice do vetor	Mês
[0]	janeiro
[1]	fevereiro
[2]	março
[3]	abril
[4]	maio
[5]	junho
[6]	julho
[7]	agosto
[8]	setembro
[9]	outubro
[10]	novembro
[11]	dezembro

Passos:

Construa uma classe pública chamada **Exercicio2**.
Essa classe possui o método main que deve receber os dados numéricos na seguinte ordem:
-O dia do mês;
-O dia da semana [0-6];
-O mês [0-11];
-O ano.
Imprima a data resultante.

Exemplo

Entrada:
Dia do mês: 30
Dia da semana: 5
Mês: 7
Ano: 2008
Resultado: Sábado, 30 de agosto de 2008

Restrições:

As entradas de dados não precisam ter validações de dados.
O dia da semana e mês devem ser exatamente iguais aos da tabela.
A frase resultante deve seguir o mesmo padrão do exemplo.

Dicas:

Inicialize o vetor dia da semana e mês com seus valores.

Cap. 14 - Exercício 3

Objetivo:

Construa uma aplicação em Java que receba uma frase e uma letra digitados pelo usuário.
Converta a frase para letras minúsculas, crie um vetor e jogue cada palavra em uma posição.
Imprima somente as palavras que tenham a letra digitada pelo usuário.

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber uma frase e uma letra digitados pelo usuário.
Converta a frase para letras minúsculas.
Crie um vetor de Strings e coloque cada palavra da frase em uma posição do vetor.
Imprima somente as palavras que contenham a letra digitada inicialmente.

Exemplo

Entrada:
Frase: O Rato Roeu A Roupa Do Rei De Roma
Letra: a
Saída:
rato
a
roupa
roma

Restrições:

Imprima cada palavra em uma linha.

Dicas:

Utilize os métodos toLowerCase() para conveter em letras minúsculas,
split() para separar cada palavra e
contains() para procurar a letra.

Cap. 14 - Exercício 4

Objetivo:

Construa uma aplicação em Java para corrigir provas de multipla escolha.
Cada prova contém 10 questões e cada questão vale 1 ponto.
Receba as respostas de 5 alunos e imprima suas respectivas notas.
Por último imprima a quantidade de alunos aprovados, sabendo-se que a nota mínima é 6.

Passos:

Construa uma classe pública chamada **Exercicio4**.
Essa classe possui o método main que deve receber o gabarito de respostas digitadas pelo usuário e armazenar em um vetor.
Não use o método **split** nesse exercício. Faça a entrada de cada dado individualmente.
Após receba as respostas de cada aluno e imprima sua nota.
Faça isso com cada 1 dos 5 alunos.
Ao final imprima a quantidade de alunos aprovados, sabendo-se que a nota mínima é 6.

Exemplo

Entrada Gabarito: [a,b,c,a,b,c,a,b,c,a]
Entrada Aluno1: [a,b,c,a,b,c,a,b,c,a]
Saída nota: [10]
Entrada Aluno2: [a,a,a,a,a,a,a,a,a,a]
Saída nota: [4]
Entrada Aluno3: [c,a,b,c,a,b,c,a,b,c]
Saída nota: [0]
Entrada Aluno4: [a,b,b,a,b,b,a,b,b,c]
Saída nota: [6]
Entrada Aluno5: [a,b,b,a,b,b,a,a,b,c]
Saída nota: [5]
Saída Total Aprovados: [2]

Restrições:

Imprima cada saída em uma linha.
Não use o método **split** nesse exercício.
Armazene o gabarito em um vetor.

Capítulo 15 - Arrays Bidimensionais

Cap. 15 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba a temperatura média de cada mês e imprima o mês mais frio e o mês mais quente.

Passos:

Construa uma classe pública chamada **Exercicio1**.
Essa classe possui o método main que deve carregar uma matriz de 2 dimensões do tipo String.
A 1ª coluna deve conter os meses, que devem ser armazenados com os nomes de cada mês de acordo com a tabela abaixo (NÃO digitar essas informações!).
A 2ª coluna deve conter a temperatura média que deve ser digitada pelo usuário.
Calcule e imprima o mês mais frio e o mês mais quente.

Indice vetor	Mês	Temperatura
0	Janeiro	-
1	Fevereiro	-
2	Março	-
3	Abril	-
4	Maio	-
5	Junho	-
6	Julho	-
7	Agosto	-
8	Setembro	-
9	Outubro	-
10	Novembro	-
11	Dezembro	-

Restrições:

O indice de um vetor começa sempre pela posição 0.
As temperaturas devem ser do tipo **double**.
Imprima somente o mês, da mesma forma como está na tabela acima.
Imprima cada mês em uma linha.

Dicas:

Utilize o laço FOR para receber os dados e para efetuar os cálculos.
Converta os dados das temperaturas de String para double para fazer os cálculos.

Exemplo

Entrada: 32, 33.5, 31, 30, 28, 27, 26, 23, 23.6, 26, 29, 30

Saída:
Agosto
Fevereiro

Cap. 15 - Exercício 2

Objetivo:

Construa uma aplicação em Java que carregue um vetor com 5 modelos de carros junto com seus consumos.
Armazene os dados em uma matriz.
Calcule e imprima o carro mais econômico e o modelo e quantos litros são gastos para percorrer 1.000km para cada carro.

Passos:

Construa uma classe pública chamada **Exercicio2**.
Essa classe possui o método main que deve receber o modelo e consumo de 5 carros.
Armazene os dados em uma matriz de 2 dimensões.
A 1ª coluna deve conter os carros e a 2ª coluna deve conter o consumo.
Calcule e imprima o carro mais econômico e o modelo e o consumo de cada carro.

Restrições:

O indice de um vetor começa sempre pela posição 0.
Os consumos devem ser do tipo **double**.
Imprima cada dado em uma linha.

Dicas:

Utilize o laço FOR para receber os dados e para efetuar os cálculos.

Exemplo

Entrada:
Modelo: [fusca]
Consumo: [5]
Modelo: [gol]
Consumo: [19]
Modelo: [uno]
Consumo: [18]
Modelo: [corsa]
Consumo: [15]
Modelo: [santana]
Consumo: [11]
Saída:
Modelo mais econômico: [gol]
Modelo: [fusca]
Consumo em litros (1.000km): [200]
Modelo: [gol]
Consumo em litros (1.000km): [52.63]

```
Modelo: [uno]
Consumo em litros (1.000km): [55.55]
Modelo: [corsa]
Consumo em litros (1.000km): [66.66]
Modelo: [santana]
Consumo em litros (1.000km): [90.9]
```

Cap. 15 - Exercício 3

Objetivo:

Construa uma aplicação em Java que carregue uma matriz do tipo 5x3.
A 1ª coluna receberá os nomes dos 5 funcionários.
A 2ª coluna receberá os salários dos 5 funcionários e a 3ª coluna
receberá a quantidade de anos que cada funcionário trabalha na empresa.
Mostre primeiro os funcionários que não receberão aumento.
Depois mostre o nome e o novo salário dos funcionários que receberão aumento.

Condições para o aumento:

Condição	Aumento
Tempo de serviço > 5 e salário < 500	35%
Tempo de serviço > 5	25%
Salário < 500	15%

Os funcionários que terão direito ao aumento são aqueles que possuem tempo de
serviço superior a 5 anos ou salário inferior a R$ 500,00.
Os funcionários que satisfazerem as 2 condições acima (tempo de serviço e salário)
o aumento será de 35%.
Os funcionários que satisfazerem apenas a condição de tempo de serviço, o aumento será de 25%.
Os funcionários que satisfazerem penas a condição de salário, o aumento será de 15%.

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber a matriz do tipo 5x3,
com os dados digitados pelo usuário.
1º receba o nome do funcionário, depois o salário e por fim o tempo de serviço.
Imprima 1º o nome dos funcionários que não receberão aumento.
Depois calcule e imprima o nome e o novo salário de cada funcionário de acordo
com as condições do enunciado.

Restrições:

Imprima cada dado em uma linha.
Para o salário utilize variáveis do tipo **double**.

Exemplo

Entrada:
Nome: Funcionário1, Salário: 500, Tempo de serviço: 5
Nome: Funcionário2, Salário: 500.01, Tempo de serviço: 5
Nome: Funcionário3, Salário: 499.99, Tempo de serviço: 4
Nome: Funcionário4, Salário: 800, Tempo de serviço: 4

Nome: Funcionário5, Salário: 800, Tempo de serviço: 6
Saída:
Funcionário1
Funcionário2
Funcionário4
Funcionário3
574.9885
Funcionário5
1000.0

Capítulo 16 - Manipulação de Strings

Cap. 16 - Exercício 1

Objetivo:

Construa uma aplicação em Java que receba uma palavra e compare com a palavra **teste**.
Se for igual imprima **IGUAL**, se for diferente imprima **DIFERENTE**.

Passos:

Construa uma classe pública chamada **Exercicio1**
Essa classe possui o método main que deve receber uma palavra digitada pelo usuário.
Compare com a palavra **teste** e imprima **IGUAL** ou **DIFERENTE**.

Restrições:

Imprimir extamente a palavra solicitada em letras maiúsculas.

Dicas:

Utilize o método equals().

Cap. 16 - Exercício 2

Objetivo:

Construa uma aplicação em Java que receba uma palavra digitada pelo usuário.
Imprima cada letra da palavra em uma linha.

Passos:

Construa uma classe pública chamada **Exercicio2**
Essa classe possui o método main que deve receber uma palavra digitada pelo usuário.
Imprima cada letra da palavra em uma linha.

Restrições:

Imprimir cada letra em uma linha.

Dicas:

Utilize o laço for para imprimir cada letra.
O método length() retorna a quantidade de letras da palavra.
O método charAt(x) retornar a letra que se localiza na posição x.

Exemplo:

Entrada: Aprenda Java
Saída: A
p
r
e
n
d
a

J
a
v
a

Cap. 16 - Exercício 3

Objetivo:

Construa uma aplicação em Java que receba uma url digitada pelo usuário.
Verifique se o prefixo e o sufixo da URL são válidos.
O prefixo deve ser = **http://** e o sufixo = **com.br**.
Imprima **VALIDO** ou **INVALIDO**.

Passos:

Construa uma classe pública chamada **Exercicio3**.
Essa classe possui o método main que deve receber um url digitada pelo usuário.
Verifique a validade da URL e imprima **VALIDO** ou **INVALIDO**.

Restrições:

Imprimir a palavra em letras maiúsculas e sem acento.

Dicas:

O método startsWith(x) retorna true se a palavra contém a String x como prefixo.
O método endsWith(x) retornar true se a palavra contém a String x como sufixo.

Cap. 16 - Exercício 4

Objetivo:

Construa uma aplicação em Java que receba uma frase digitada pelo usuário
Inverta a frase e imprima.

Passos:

Construa uma classe pública chamada **Exercicio4**.
Essa classe possui o método main que deve receber uma frase digitada pelo usuário.
Inverta a frase e imprima.

Dicas:

Utilize uma String para receber cada letra da frase, para depois imprimi-la.
O laço for ajudará a percorrer cada letra da palavra.
O método charAt(x) retorna um char, ele devolve o caracter da posição **x** e para armazená-lo na String acima é necessário converte-lo, para isso utilize o comando **String.valueOf**.

Exemplo:

Entrada: Aprenda Java
Saída: avaJ adnerpA

Cap. 16 - Exercício 5

Objetivo:

Construa uma aplicação em Java que receba uma frase e uma palavra digitada pelo usuário.
Procure na frase a existência da palavra, se existir imprima sua posição, se não imprima a frase **NAO ENCONTRADO**, sem acento e em letras maiúsculas.

Passos:

Construa uma classe pública chamada **Exercicio5**.
Essa classe possui o método main que deve receber uma frase e uma palavra digitada pelo usuário.
Procure a existência da palavra na frase e imprima sua posição, se não imprima **NAO enocntrado**.

Ordem de Entrada:

Receber primeiro a frase e depois a palavra.

Dicas:

A mensagem **NAO ENCONTRADO** deve ser impressa em letras maiúsculas e sem acento.
Utilize o método indexOf() para procurar a palavra na frase.

Exemplo:

Frase: Aprenda Java
Palavra: a
Saída: 6

Cap. 16 - Exercício 6

Objetivo:

Construa uma aplicação em Java que receba uma frase digitada pelo usuário.
Elimine todos os espaços em branco antes e depois da frase e imprima em letras minúsculas.

Passos:

Construa uma classe pública chamada **Exercicio6**.
Essa classe possui o método main que deve receber uma frase digitada pelo usuário.
Elimine os espaços em branco antes e depois da frase.
Converta para letras minúsculas.
Imprima o resultado.

Dicas:

Utilize o método trim() e o método toLowerCase().

Exemplo:

Frase: [Aprenda Java]
Saída: [aprenda java]

Cap. 16 - Exercício 7

Objetivo:

Construa uma aplicação em Java que receba uma frase digitada pelo usuário.
Troque todos os espaços em branco por "-" e elimine o 1º e o último caracter.
Imprima o resultado.

Passos:

Construa uma classe pública chamada **Exercicio7**.
Essa classe possui o método main que deve receber uma frase digitada pelo usuário.
Troque todos os espaços em branco por "-" e elimine o 1º e o último caracter.
Imprima o resultado.

Dicas:

Utilize o método replace() para trocar os caracteres e o substring() para eliminar os caracteres.

Exemplo:

Frase: [cafe com leite]
Saída: [afe-com-leit]

Cap. 16 - Exercício 8

Objetivo:

Construa uma aplicação em Java que receba uma frase digitada pelo usuário.
Considere que as frases sempre terão 3 palavras.
Troque a 1ª palavra pela 3ª palavra.
Imprima o resultado.

Passos:

Construa uma classe pública chamada **Exercicio8**.
Essa classe possui o método main que deve receber uma frase digitada pelo usuário.
Troque a 1ª palavra pela 3ª palavra.
Imprima o resultado.

Dicas:

Utilize o laço for para percorrer todos os caracteres da frase.
Identifique cada palavra pelo espaço em branco.

Exemplo:

Frase: [cafe com leite]
Saída: [leite com cafe]

Capítulo 17 - Classe Math

Cap. 17 - Exercício 1

Objetivo:

Construa uma aplicação em Java que calcule o perímetro de um círculo.

Passos:

Construa uma classe pública chamada **Exercicio1**
Essa classe possui o método main que deve receber o valor do raio digitado pelo usuário.
Calcular e imprimir o valor do perímetro.

Dicas:

Utilize a seguinte função para calcular o valor de PI:

```
1.    double PI = Math.PI;
```

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 17 - Exercício 2

Objetivo:

Construa uma aplicação em Java que calcule a área de um circulo.

Passos:

Construa uma classe pública chamada **Exercicio2**
Essa classe possui o método main que deve receber o valor do raio digitado pelo usuário.
Calcular e imprimir o valor da área.

Dicas:

Para elevar um número ao quadrado você pode utilizar:

```
1.    double quadrado = Math.pow(double numero, double elevado);
```

Utilize a seguinte função para calcular o valor de PI:

```
1.    double PI = Math.PI;
```

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.

Cap. 17 - Exercício 3

Objetivo:

Construa uma aplicação em Java que calcule a hipotenusa de um triângulo retângulo utilizando o Teorema de Pitágoras.

Passos:

Construa uma classe pública chamada **Exercicio3**
Essa classe possui o método main que deve receber o valor do cateto1 e cateto2 digitado pelo usuário.
Calcular e imprimir o valor da hipotenuza.

Dicas:

Para elevar um número ao quadrado você pode utilizar:

```
1.    double quadrado = Math.pow(double numero, double elevado);
```

Para calcular a raiz quadrada utilize:

```
1.    double raiz = Math.sqrt(double numero);
```

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.
Cálculo baseado no Teorema de Pitágoras.

Cap. 17 - Exercício 4

Objetivo:

Construa uma aplicação em Java que calcule o volume em litros de um cilindro.

Passos:

Construa uma classe pública chamada **Exercicio4**
Essa classe possui o método main que deve receber o valor do raio e da altura do cilindro (em cm) digitado pelo usuário, obrigatoriamente nessa ordem (raio, altura).
Calcular e imprimir o valor do volume.

Dicas:

Utilize a seguinte função para calcular o valor de PI:

```
1.    double PI = Math.PI;
```

Para elevar um número ao quadrado você pode utilizar:

```
1.    double quadrado = Math.pow(double numero, double elevado);
```

Restrições:

Imprimir somente o valor calculado, NÃO imprimir qualquer outra informação.
Utilizar variáveis do tipo **double**.
Não esquecer de converter o valor para litros.

Capítulo 18 - Testes Finais

Cap. 18 - Exercício 1

Objetivo:

Faça uma aplicação Java que receba uma seqüência de caracteres (String), processe a mesma o mostre o resultado solicitado

Passos:

Construa uma classe pública chamada **TesteFinal1**
Essa classe possui somente o método main que deve receber um String.
Da seqüência de caracteres recebidos, a aplicação deve considerar somente os caracteres 0 e 1.
Após receber a seqüência de caracteres, a aplicação deve avaliar individualmente essa seqüência da DIREITA para a ESQUERDA para poder calcular o resultado final que deve acumular os valores segundo essa fórmula:

```
resultadoFinal =  O primeiro  0 ou 1 válido vezes 1 +
                              O segundo 0 ou 1 válido vezes 2 +
                              O terceiro   0 ou 1 válido vezes 3 +
                              O quarto    0 ou 1 válido vezes  4 +
                              O n              0 ou 1 válido vezes n;
```

Exemplos:

```
Veja os exemplos abaixo:

Entrada: 0
Processamento: (0*1)=0
Resultado a ser impresso: 0.0

Entrada: 1
Processamento: (1*1)=1
Resultado a ser impresso : 1.0

Entrada: 11
Processamento:  (1*1) + (1*2)=3
Resultado a ser impresso: 3.0

Entrada: 10
Processamento:  (0*1) + (1*2)=2
Resultado a ser impresso: 2.0

Entrada: 101
Processamento: (1*1) + (0*2) + (1*3)=4.0
Resultado a ser impresso: 4.0
```

```
Entrada:  1121A1
Processamento: ( 1*1)+(1*2)+(1*3)+(1*4)=10.
Resultado a ser impresso: 10.0

Entrada:  1021X1
Processamento: ( 1*1)+(1*2)+(0*3)+(1*4)=7.
Resultado a ser impresso: 7.0

Entrada: 23456asd54gfs
Processamento:  nenhum
Resultado a ser impresso: 0.0
```

Restrições:

O usuário deve digitar somente uma entrada de dados.
Considerar somente os caracteres válidos para o processamento.
Imprimir somente o valor resultado final.
Faça a avaliação da direita para a esquerda da entrada de dados.
Use uma variável double para armazenar o resultado final.