

Lista de Exercícios 3 – Programação Orientada a Objetos
Maikon Bueno

Java Básico

1. Desenvolva um programa que imprima a tabuada parametrizada: multiplicador e range da multiplicação (inicial e final). Exemplo: parâmetros: 4, 2, 5 → imprime: 4*2=8 | 4*3=12 | 4*4=16 | 4*5=20
2. Desenvolva um programa que leia 10 números e depois os imprima em ordem reversa em que foram inseridos
3. Declare dois arrays de 5 elementos com valores inteiros quaisquer. Imprima a soma desses arrays.
4. Desenvolva um programa que leia um número e utilize-o como largura para imprimir os seguintes formatos:

(a)

```
#
# #
# # #
# # # #
# # # # #
# # # # # #
# # # # # # #
# # # # # # # #
```

(b)

```
# # # # # # # #
# # # # # # #
# # # # # #
# # # # #
# # # #
# # #
# #
#
```

(c)

```
# # # # # # #
# # # # # #
# # # # #
# # # #
# # #
# #
#
```

(d)

```
#
# #
# # #
# # # #
# # # # #
# # # # # #
# # # # # # #
# # # # # # # #
```

(e)

```
# # # # # # #
#
#
#
#
#
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
```

(f)

```
# # # # # # #
#
#
#
#
#
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
```

(g)

```
# # # # # # #
#
#
#
#
#
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
```

(h)

```
# # # # # # #
#
#
#
#
#
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
```

(i)

```
# # # # # # #
# #
# #
# #
# #
# #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
# # # # # # #
```

5. Leia um parâmetro inteiro e reproduza a seguinte saída.

Para parâmetro = 9

*		1	2	3	4	5	6	7	8	9
1		1	2	3	4	5	6	7	8	9
2		2	4	6	8	10	12	14	16	18
3		3	6	9	12	15	18	21	24	27
4		4	8	12	16	20	24	28	32	36
5		5	10	15	20	25	30	35	40	45
6		6	12	18	24	30	36	42	48	54
7		7	14	21	28	35	42	49	56	63
8		8	16	24	32	40	48	56	64	72
9		9	18	27	36	45	54	63	72	81

6. Leia uma string e compute a porcentagem de vogais e consoantes do texto lido.
7. Leia uma string que corresponda a um valor binário. Verifique se o número binário está corretamente digitado e imprima o valor decimal correspondente.
8. Leia um valor inteiro N. Leia N nomes. Leia N sobrenomes. Imprima todas as combinações possíveis de nomes e sobrenomes.

9. Declare dois arrays de caracteres: o primeiro contendo consoantes e o segundo contendo vogais. Leia um número inteiro N do usuário. Imprima N palavras, contendo um número aleatório de sílabas aleatórias montadas a partir dos dois arrays.
Utilize: `int iAleatorio = (int) (Math.random() * 10 + 1);`
Onde: `random()` retorna um valor de 0 a 0.999...9 e `iAleatorio` varia de 0 a 10.
10. Crie um método que receba dois arrays do tipo `ArrayList<int>` e verifique se ambos são iguais retornando `true` ou `false`.

Classes Básico

11. Crie uma classe para representar um produto, com os atributos código, nome e valor. Implemente também o método `toString()`, retornando uma String formatada com os dados do objeto. Instancie, atribua valores e imprima o objeto final.
12. Crie uma classe chamada `VetorInteiro`. Crie os seguintes métodos
→ `add(int valor)` para adicionar novo inteiro
→ `get(int i)` que retorna elemento inteiro na posição i
→ `get(int i, int j)` que retorna um novo `VetorInteiro` contendo elementos da posição i a j
→ `add(VetorInteiro v)` para incluir os valores de 'v' ao vetor corrente
→ `reverse()` que retorne um novo `VetorInteiro`, com a ordem dos elementos invertida
13. Crie uma classe que representa um 'cliente'. Implemente diferentes tipos de construtores e instancie objetos utilizando-os.
14. Crie uma classe que representa hora (hora, minutos e segundos). Crie métodos para retornar a quantidade de segundos, minutos (`double`) e horas (`double`) em relação a 0:00h. Implemente também o método `toString()`, para imprimir a hora no formato adequado.
15. Crie uma classe que representa um Dicionário de inteiros. Essa classe deve armazenar inteiros indexados por strings. Implemente conforme o exemplo a seguir:

```
Dicionario d = new Dicionario( );
d.insert("numeros pares", 2);
d.insert("numeros pares", 4);
d.insert("numeros divisiveis por 10", 10);
d.insert("numeros divisiveis por 10", 20);
d.insert("numeros divisiveis por 10", 30);
d.print("numeros pares");
d.print("numeros quaisquer");
d.print("numeros divisiveis por 10");
ArrayList<int> l = d.get("numeros divisiveis por 10"); // retorna: [ 10, 20, 30 ]
```

Output:

2 ,4

10, 20, 30

16. Crie uma classe chamada `MenuOpcoes`. Essa classe deve receber um vetor de String's que conterão as opções a serem impressas. Implemente uma função para imprimir o menu e outra para retornar o valor escolhido, conforme o exemplo a seguir:

```
MenuOpcoes m = newMenuOpcoes({"Primeira opcao","Segunda opcao","Terceira opcao"});
m.imprimir( );
System.out.println("A opção escolhida foi: " + m.lerOpcao( ) );
```

Output:

1 – Primeira opção

2 – Segunda opção

3 – Terceira opção

0 – Sair

Digite sua opção: 3

A opção escolhida foi: 3

17. Repita a saída do exercício anterior até que a opção escolhida seja 0 (zero).
18. Crie uma classe para representar um Curso (nome, descricao, horas, instrutor), de modo que seja possível instanciar um objeto passando no construtor todos os atributos ou também desse modo:

```
Curso cInformatica = new Curso ();  
cInformatica.setNome("Informatica basica").setDescricao("Curso de windows, word e powerpoint").setHoras(  
100).setInstrutor("Horácio Ludivico");
```

19. Crie uma classe para representar uma Matriz NxN. Implemente os métodos somar e subtrair que retornam uma nova instância do objeto Matriz a partir dos parâmetros recebidos.

```
Matriz m1 = new Matriz({{1,2},{3,4}});  
Matriz m2 = new Matriz({{1,1},{2,2}});  
Matriz mSoma = m1.soma(m2);  
Matriz m1Subm2 = m1.subtrai(m2)  
System.out.println(mSoma);  
System.out.println();  
System.out.println(m1Subm2);
```

Output:

```
2 3  
5 6  
  
0 1  
1 2
```

20. Crie uma classe Livro contendo os seguintes atributos: Titulo, Autores (pode ser mais que um → armazene em um array de strings), Editora, Ano de Publicação e Preço. Faça um programa que leia um inteiro N e depois leia N livros, incluindo autores distintos. Armazene os N livros em um ArrayList.
21. Acrescente ao exercício anterior a leitura de uma porcentagem que alterará o valor de todos os livros armazenados no array. Por fim, imprima todos os livros cadastrados.

Associação: Unária, Binária, Agregação e Composição

22. Implemente as classes Produto (nome, departamento, preço) e Departamento (nome), considerando que a classe Produto possui uma instância de um departamento ao qual ele pertence. Leia X Departamentos e armazene-os em um array. Leia Y produtos e armazene-os em um array. Imprima todos os produtos e seus departamentos.
23. Considere agora que, além de cada produto possuir uma instância de um departamento, cada departamento deve possuir uma coleção de seus produtos. Instancie-os de modo que o a mesma instância do produto inserido no array de produtos seja também inserido na coleção de seu departamento. Imprima todos os produtos e seus departamentos. Imprima todos os departamentos e seus produtos.
24. Considere uma loja que monta bicicletas customizadas, peça a peça. Nesse caso, tal loja possui diversos produtos a venda, tais como pneus, bancos e kit de transmissão. Além desses, bicicletas também são produtos e são formadas pela composição de outros produtos da loja. Implemente um modelo orientado a objeto para este caso.
25. Implemente as classes necessárias para realizar vendas de produtos: Produto (implementada anteriormente), Departamento (implementada anteriormente), Venda (id, cliente, total), ItemVenda (produto, quantidade, desconto, total). Implemente a leitura e armazenamento dos item anteriores: departamentos, produtos, vendas seus itens (utilize a classe Menu para

ajudar com as opções). Acrescente uma opção no menu para imprimir todas as vendas realizadas, incluindo seus produtos e departamento de cada produto.

Herança e Polimorfismo

26. Utilize a herança quando oportuno na implementação das seguintes classes: Caminhão, Veículo, Moto, VeiculoTerrestre, Navio, Lancha, VeiculoAquatico, Carro, Bicicleta. Acrescente atributos que julgar relevantes a cada classe e instancie todas.
27. Em um sistema existem os seguintes cadastros: Clientes (Pessoas Físicas ou Jurídicas), Fornecedores (Jurídicos), Vendedores, Gerentes e Funcionários em Geral. Identifique onde a herança pode ser utilizada para obter maior aproveitamento de código possível, agrupando métodos e atributos em comum. Crie as classes necessárias para implementar esse cadastro, bem como, identifique atributos de cada classe.
28. Implemente as seguintes classes: Animal, Réptil, Ave, Mamífero, Anfíbio, Sapo, Cachorro, Jacaré e Arara. Tais classes não devem possuir nenhum atributo, mas devem implementar o método quemSou() que deve retornar uma String com o nome da classe em questão. A hierarquia de classes, designada pela herança entre elas, deve ser encabeçada pela classe Animal. Crie uma classe adicional chamada TesteAnimais, que deve implementar um método que recebe um animal e imprime o nome de sua classe. Crie diversos objetos das várias classes e utilize este último método para verificar o resultado.