

### Texto para as questões 1) e 2):

Deseja-se implementar um programa em Java que, dado um conjunto de números inteiros, gera todos os múltiplos desses números entre 0 e 100. Também deve ser possível imprimir todos os valores gerados. Por exemplo, se o conjunto de números for {11, 49}, deve ser gerado o seguinte arranjo de arranjos:

```
múltiplos = { {0, 11, 22, ..., 99},    //múltiplos de 11, de 0 a 100
              {0, 49, 98}              } //múltiplos de 49, de 0 a 100
```

### Pergunta 1

2 pts

Assinale a alternativa que contenha um código que completa corretamente o método abaixo e gera o arranjo de arranjos no formato desejado.

```
static int[][] geraMultiplos(int[] conjunto) {
    int[][] multiplos = new int[conjunto.length][];
    // CÓDIGO A SER UTILIZADO
    return multiplos;
}
```

☐

```
for (int i = 0; i < conjunto.length; i++) {
    int base = conjunto[i];
    multiplos[i] = new int[base + 1];
    for (int j = 0; j < multiplos[i].length; j++) {
        multiplos[i][j] = 100/base;
    }
}
```

☐

```
for (int i = 0; i < conjunto.length; i++) {
    int base = conjunto[i];
    multiplos[i] = new int[base + 1];
    for (int j = 0; j < multiplos[i].length; j++) {
        multiplos[i][j] = j*base;
    }
}
```

☐

```
for (int base : conjunto) {
    multiplos[base] = new int[base];
    for (int j = 0; j < multiplos[base].length; j++) {
        multiplos[base][j] = base++;
    }
}
```



```
for (int i = 0; i < conjunto.length; i++) {
    int base = conjunto[i];
    multiplos[i] = new int[(100/base) + 1];
    for (int j = 0; j < multiplos[i].length; j++) {
        multiplos[i][j] = j*base;
    }
}
```



```
for (int i = 0; i < conjunto.length; i++) {
    int base = conjunto[i];
    multiplos[i] = new int[(100/base) + 1];
    for (int j = 0; j < multiplos[i].length; j++) {
        multiplos[i][j] = base++;
    }
}
```

## Pergunta 2

2 pts

Assinale a alternativa que implementa corretamente o método para impressão de um arranjo de arranjos genérico e que pode, então, ser utilizado para imprimir o arranjo *multiplos* criado no item anterior:



```
static void imprimeMatriz(int[][] matriz) {
    for (int[] linhas : matriz) {
        for (int val : linhas) {
            System.out.print(val + " ");
        }
        System.out.println("");
    }
}
```



```
static void imprimeMatriz(int[][] matriz) {
    for (int i = 0; i < matriz.length; i++) {
        for (int j = 0; j < matriz.length; j++) {
            System.out.print(matriz[i][j] + " ");
        }
        System.out.println("");
    }
}
```

☐

```
static void imprimeMatrizEr(int[][] matriz) {
    for (int[] linhas : matriz) {
        System.out.println(linhas + " ");
    }
}
```

☐

```
static void imprimeMatrizEr(int[][] matriz) {
    for (int[] linhas : matriz) {
        for (int i = 0; i < linhas.length; i++) {
            System.out.print(matriz[i] + " ");
        }
        System.out.println("");
    }
}
```

☐

```
static void imprimeMatriz(int[][] matriz) {
    int linhas = matriz.length;
    int colunas = matriz[0].length;
    for (int i = 0; i < linhas; i++) {
        for (int j = 0; j < colunas; j++) {
            System.out.print(matriz[i][j] + " ");
        }
        System.out.println("");
    }
}
```

### Pergunta 3

2 pts

Deseja-se implementar um programa em Java que, dada uma lista com os jogos de diversos jogadores e o resultado do sorteio, identifica quais deles acertaram a quadra. Por exemplo, se os jogos forem:

```
int[][] jogos = { {1, 2, 3, 4, 5, 6},
                  {1, 2, 5, 7, 9, 11},
                  {1, 3, 5, 6, 9, 10},
                  {2, 3, 4, 5, 6, 7},
```

```
        {1, 3, 6, 9, 12, 15}
    };
```

E o sorteio for:

```
int[] sorteio = {1, 3, 5, 6};
```

Então o método deve retornar como resultado {true, false, true, false, false}, pois apenas o primeiro e o terceiro jogadores acertaram o jogo. Assinale a alternativa que contenha um código que completa corretamente o método abaixo, gerando o arranjo de booleanos desejado.

```
static boolean[] checkQuadra(int[][] jogos, int[] sorteio) {
    boolean[] ganhou = new boolean[jogos.length];
    // CÓDIGO A SER UTILIZADO
    return ganhou;
}
```



```
boolean[] acerto = {false, false, false, false};
for (int i = 0; i < ganhou.length; i++) {
    for (int palpite : jogos[i]) {
        if(palpite == sorteio[0]) {acerto[0] = true;}
        if(palpite == sorteio[1]) {acerto[1] = true;}
        if(palpite == sorteio[2]) {acerto[2] = true;}
        if(palpite == sorteio[3]) {acerto[3] = true;}
    }
    ganhou[i] = acerto[0] && acerto[1] && acerto[2] && acerto[3];
}
```



```
boolean[] acerto = {false, false, false, false};
for (int i = 0; i < ganhou.length; i++) {
    for (int palpite : jogos[i]) {
        if(palpite == sorteio[0]) {acerto[0] = true;}
        if(palpite == sorteio[1]) {acerto[1] = true;}
        if(palpite == sorteio[2]) {acerto[2] = true;}
        if(palpite == sorteio[3]) {acerto[3] = true;}
    }
    ganhou[i] = acerto[0] || acerto[1] || acerto[2] || acerto[3];
}
```



```
for (int i = 0; i < ganhou.length; i++) {
    boolean[] acerto = {true, true, true, true};
    for (int palpite : jogos[i]) {
        if(palpite != sorteio[0]) {acerto[0] = false;}
        if(palpite != sorteio[1]) {acerto[1] = false;}
        if(palpite != sorteio[2]) {acerto[2] = false;}
        if(palpite != sorteio[3]) {acerto[3] = false;}
    }
    ganhou[i] = acerto[0] && acerto[1] && acerto[2] && acerto[3];
}
```

☒

```
for (int i = 0; i < ganhou.length; i++) {
    boolean[] acerto = {false, false, false, false};
    for (int palpite : jogos[i]) {
        if(palpite == sorteio[0]) {acerto[0] = true;}
        if(palpite == sorteio[1]) {acerto[1] = true;}
        if(palpite == sorteio[2]) {acerto[2] = true;}
        if(palpite == sorteio[3]) {acerto[3] = true;}
    }
    ganhou[i] = acerto[0] && acerto[1] && acerto[2] && acerto[3];
}
```

☐

```
boolean[] acerto = {true, true, true, true };
for (int i = 0; i < ganhou.length; i++) {
    for (int palpite : jogos[i]) {
        if(palpite != sorteio[0]) {acerto[0] = false;}
        if(palpite != sorteio[1]) {acerto[1] = false;}
        if(palpite != sorteio[2]) {acerto[2] = false;}
        if(palpite != sorteio[3]) {acerto[3] = false;}
    }
    ganhou[i] = acerto[0] || acerto[1] || acerto[2] || acerto[3];
}
```

### Texto para as questões 4) e 5):

Deseja-se implementar o jogo de Batalha Naval usando Java. Para isso, deve ser criada a classe “Tabuleiro”, que tem a seguinte especificação:

- Essa classe deve conter 1 atributo de objeto (não de classe), o mar: trata-se de uma matriz quadrada cujo tamanho é passado como parâmetro no construtor da classe.
  - Os valores dessa matriz são inteiros: 0 significa que não há navio naquela coordenada; 1, que há um navio e que ele não foi abatido; 2, que havia um navio, mas ele foi abatido.
- Essa classe deve ter os seguintes métodos, além do construtor:
  - *colocarNavio*: recebe como parâmetros as coordenadas do navio a ser colocado no tabuleiro, e atualiza o tabuleiro de acordo.

- *retirarNavio*: recebe como parâmetros as coordenadas do navio a ser retirado no tabuleiro, e atualiza o tabuleiro de acordo.
- *contarNavios*: retorna o número total de posições no tabuleiro que ainda têm navios não abatidos.
- *atirar*: recebe como parâmetros as coordenadas de um tiro; se havia um navio naquela posição, atualiza o tabuleiro de acordo e retorna true para indicar que o tiro foi certo, caso contrário, retorna false para indicar que o tiro foi na água.

Assinale a alternativa que descreve corretamente uma possível implementação dessa classe, considerando as duas partes do código.

#### Pergunta 4

2 pts

Os atributos e o construtor, bem como a localização dos métodos.



```
public class Tabuleiro {

    int[][] mar;

    public Tabuleiro(int tamanho) {
        this.mar = new int[tamanho][tamanho];
        for (int i = 0; i < tamanho; i++) {
            for (int j = 0; j < tamanho; j++) {
                this.mar[i][j] = 0;
            }
        }
    }

    //Os métodos vêm aqui
}
```



```
public class Tabuleiro {

    int[][] mar;

    public Tabuleiro(int tamanho) {
        this.mar = new int[tamanho][tamanho];
        for (int i = 0; i < tamanho; i++) {
            for (int j = 0; j < tamanho; j++) {
                this.mar[i][j] = 0;
            }
        }
    }

    //Os métodos vêm aqui
}
```

```
public class Tabuleiro {

    int[][] mar;

    public Tabuleiro(int tamanho) {
        this.mar = new int[tamanho][tamanho];
        for (int valor : mar) {
            valor = 0;
        }
    }

    //Os métodos vêm aqui
}
```

```
public class Tabuleiro {

    int[][] mar;

    public Tabuleiro(int tamanho) {
        this.mar = new int[tamanho][tamanho];
        for (int i = 0; i < tamanho; i++) {
            for (int j = 0; j < tamanho; j++) {
                this.mar[i][j] = i;
            }
        }
    }

    //Os métodos vêm aqui
}
```

```
public class Tabuleiro {

    double[][] mar;

    public Tabuleiro(double tamanho) {
        this.mar = new double[tamanho][tamanho];
        for (int i = 0; i < tamanho; i++) {
            for (int j = 0; j < tamanho; j++) {
                this.mar[i][j] = 0.0;
            }
        }
    }

    //Os métodos vêm aqui
}
```

## Os métodos da classe.

```
public void colocarNavio(int x, int y) {
    this.mar[x][y] = 0;
}

public void retirarNavio(int x, int y) {
    this.mar[x][y] = 1;
}

public int contarNavios() {
    int contador = 0;
    for (int[] linha : mar) {
        for (int valor : linha) {
            if (valor != 0) { contador++; }
        }
    }
    return contador;
}

public boolean atirar(int x, int y) {
    boolean acertou = (this.mar[x][y] == 1);
    if (acertou) {
        this.mar[x][y] = 2;
    }
    return acertou;
}
```






```
public void colocarNavio(int x, int y) {
    this.mar[x][y] = 1;
}

public void retirarNavio(int x, int y) {
    this.mar[x][y] = 0;
}

public int contarNavios() {
    int contador = 0;
    for (int[] linha : mar) {
        for (int valor : linha) {
            if (valor == 1) { contador++; }
        }
    }
    return contador;
}

public boolean atirar(int x, int y) {
    if (this.mar[x][y] == 1) {
        this.mar[x][y] = 2;
        return true;
    }
    return false;
}
```



```
public void colocarNavio(int x, int y) {
    this.mar[x][y]++;
}

public void retirarNavio(int x, int y) {
    this.mar[x][y]--;
}

public int contarNavios() {
    return this.mar.length;
}

public boolean atirar(int x, int y) {
    if (this.mar[x][y] == 1) {
        this.mar[x][y] = 2;
    }
    return (this.mar[x][y] == 2);
}
```




```
public void colocarNavio(int x, int y) {
    this.mar[x][y] = 1;
}

public void retirarNavio(int x, int y) {
    this.mar[x][y] = 0;
}

public int contarNavios() {
    int contador = 0;
    for (int i = 0; i < tamanho; i++) {
        for (int j = 0; j < tamanho; j++) {
            contador = contador + this.mar[i][j];
        }
    }
    return contador;
}

public boolean atirar(int x, int y) {
    if (this.mar[x][y] == 1) {
        this.mar[x][y] = 2;
    }
    return (this.mar[x][y] == 1);
}
```



```
public void colocarNavio(int x, int y) {
    this.mar[x][y] = 1;
}

public void retirarNavio(int x, int y) {
    this.mar[x][y] = 0;
}

public int contarNavios() {
    return this.mar.length;
}

public boolean atirar(int x, int y) {
    return (this.mar[x][y] == 1);
    if (this.mar[x][y] == 1) {
        this.mar[x][y] = 2;
    }
}
```