UNIVERSIDADE ESTADUAL DA PARAÍBA
CENTRO DE CIÊNCIA E TECNOLOGIAS
CURSO DE CIÊNCIA DA COMPUTAÇÃO
DISCIPLINA: LINGUAGEM DE PROGRAMAÇÃO II
ALUNO: LUIZ CARLOS DE LEMOS JÚNIOR

1ª VERIFICAÇÃO DE APRENDIZAGEM

1. Como funciona o processo de compilação e execução de um programa em Java? **(1,0 ponto)**

   Em Java, o processo de compilação e execução se dá em duas etapas, usando o conceito de Máquina Virtual Java (JVM). Primeiro, é gerado um código intermediário durante a compilação, chamado de Bytecode. Só então no momento da execução é que a JVM traduz o Bytecode para a linguagem que o computador possa entender.

2. Como funcionam as conversões de tipos em Java? **(1,0 ponto)**

   Em Java, as conversões ocorrem de duas maneiras: ampliadoras e redutoras. As ampliadoras (promoções ou conversões implícitas) ocorrem de forma automática, ou seja, a variável de menor tamanho é convertida (promovida) para o tipo da de maior tamanho no uso da expressão. Essa conversão é automática, pois a JVM entende que deve se adaptar para que não haja perda de informação na execução da operação. Já as redutoras (coerção ou conversão explícita) precisam de um cast explícito, ou seja, o programador deve indicar (forçar) a variável definida a mudar de tipo para uma de tamanho menor.

3. Este exercício deve ser respondido com relação ao funcionamento do comando switch..case. O que acontece se o usuário digitar o número 1? O número 2? O número 3? Qual deveria ser a saída esperada? Como corrigir o código para ele funcionar de acordo com a saída esperada?

   Ao digitar 1, aparece no terminal todos os casos do programa:

   1

   2

   3

   Número diferente de 1, 2 e 3.

   Ao digitar 2, aparece no terminal todos os casos do programa a partir do caso 2:

   2

   3

   Número diferente de 1, 2 e 3.

   Ao digitar 3, aparece no terminal o restante do programa a partir do caso 3:

   3

   Número diferente de 1, 2 e 3.

Ao digitar qualquer número diferente de 1, 2 ou 3, aparece no terminal apenas o caso padrão:

Número diferente de 1, 2 e 3.

Isso ocorre porque há uma falha no código, não foi inserido o break em cada caso. Assim, ao entrar num caso, o programa segue exibindo todos os demais após ele.

A saída esperada seria mostrar apenas o número digitado para os três primeiros casos (1, 2 ou 3) ou a mensagem padrão para números diferentes desses.

```
import java.util.Scanner;

public class main {

	public static void main(String[] args) {

		Scanner sc = new Scanner(System.in);
		int op;

		System.out.println("Digite um número: ");
		op = sc.nextInt();

		switch (op) {
		    case 1:
			System.out.println("1");

		    case 2:
			System.out.println("2");

		    case 3:
			System.out.println("3");

		    default:
			System.out.println("Número diferente de 1, 2 e 3.");
		}
	}
}
```

Para corrigir o código e a saída apresentar o valor esperado, devemos introduzir o break em cada um dos casos do switch. Após a correção, o código ficará assim:

```
package prova1;

import java.util.Scanner;

public class Questao3 {

	public static void main(String[] args) {

		Scanner sc = new Scanner(System.in);
		int op;

		System.out.println("Digite um número: ");
		op = sc.nextInt();

		switch (op) {
		    case 1:
			System.out.println("1");
		    break;

		    case 2:
			System.out.println("2");
		    break;

		    case 3:
			System.out.println("3");
		    break;

		    default:
			System.out.println("Número diferente de 1, 2 e 3.");
		}
	}
}
```

4. Escreva um programa em Java que deve verificar e exibir na tela a quantidade de números pares e de números ímpares entre os números de 0 a 100. Também deve fazer o somatório dos valores pares e dos valores ímpares, exibindo ao final estes resultados na tela. **(2,0 pontos)**

   O programa implementa um contador de 0 a 100, de modo que termos 101 números, sendo 51 pares (incluindo o 0) e 50 (ímpares). A cada passo, ele testa se o número é divisível por 2, em caso afirmativo, conta-se como par e soma-se ao valor anterior. A mesma lógica é utilizada para os ímpares. Eis o código:

```
package prova1;

public class Questao4 {

	public static void main(String[] args) {
		int cont=0, somapar=0, somaimpar=0, par=0, impar=0;

		for(cont = 0; cont <= 100; cont++) {
			if(cont%2==0) {
				somapar = somapar + cont;
				par++;
			}
			else {
				somaimpar = somaimpar + cont;
				impar++;
			}
		}
		System.out.println("Número de pares de 0 a 100: "+par);
		System.out.println("Soma dos pares de 0 a 100: "+somapar);
		System.out.println("Número de impares de 0 a 100: "+impar);
		System.out.println("Soma dos impares de 0 a 100: : "+somaimpar);
	}
}
```

5. Faça um programa em Java que imprima a sequência numérica abaixo utilizando um laço de repetição. **(2,0 pontos)**

```
1
2 1
3 2 1
4 3 2 1
5 4 3 2 1
6 5 4 3 2 1
7 6 5 4 3 2 1
8 7 6 5 4 3 2 1
9 8 7 6 5 4 3 2 1
10 9 8 7 6 5 4 3 2 1
```

Para este programa, temos dois laços de repetição encadeados, simulando o problema conhecido como impressão de uma matriz. Porém, aqui os valores serão os próprios contadores dos laços externos e interno, de modo que o laço interno recebe o valor externo, imprime-o sem saltar a linha decrementando seu conteúdo. Após o término dessa etapa, o laço externo incrementa, mudando o valor do interno e repetindo o processo com um novo valor máximo, até que este atinja o pré-determinado valor de 10.

```
package prova1;

public class Questao5 {

	public static void main(String[] args) {
		for(int cont1=1; cont1 <= 10; cont1++) {
			for(int cont2 = cont1; cont2 >= 1 ; cont2--) {
				System.out.print(" "+cont2);
			}
			System.out.println(" ");
		}
	}
}
```

6. Compare e contraste a instrução de uma única seleção if e a instrução de repetição while. Qual a semelhança dessas duas instruções? Qual é a diferença? **(2,0 pontos)**

Observando a estrutura das instruções if e while:

```
if(condição){                    while(condição){
        comandos;                        comandos;
}                                }
```

Podemos observar que ambas apresentam a mesma estrutura, ou seja, serão realizadas um conjunto de comandos se a condição de sua execução for satisfeita. Porém, o if só será executado uma única vez, e seu teste servirá como decisão para sua única execução. Já no while, a rotina será executada infinitas vezes enquanto a condição for satisfeita. Para isso, adota-se um contador atrelado a condição de execução a fim de determinar uma quantidade definida de laços a srem executados.

7. Escreva um programa que recebe um ano como entrada e verifica se ele é ou não bissexto. Ao final, o programa deve exibir essa informação na tela. **(1,0 ponto)**

Para a resolução deste problema, usa-se o algoritmo de determinação de ano bissexto, em seguida, foi implementado um código baseado num conjunto de ifs e elses como forma de testar as condições do algoritmo.

Para que um ano seja bissexto deve-se:

Verificar se o número é divisível por 4, se não for, não é bissexto. Se for, deve testá-lo novamente se é divisível por 100. Nesta etapa, se o número divisível por 4 não for divisível por 100, então o ano é bissexto, do contrário, segue-se a uma nova etapa. Na etapa final, o número que antes foi divisível por 4 e por 100, agora deve ser divisível por 400 para ser de um ano bissexto. Em caso negativo, não será um valor de ano bissexto. Eis o código:

```
package prova1;

import java.util.Scanner;

public class Questao7 {

        public static void main(String[] args) {

                Scanner sc = new Scanner(System.in);
            int ano;

            System.out.println("Informe o ano: ");
            ano = sc.nextInt();

            if(ano%4 == 0) {
                if(ano%100 == 0) {
                        if(ano%400 == 0) {
                                System.out.println("O ano "+ano+" é bissexto");
                        }
                        else {
                                System.out.println("O ano "+ano+" não é bissexto");
                        }
                }
                else {
                        System.out.println("O ano "+ano+" é bissexto");
                }
            }
            else {
                System.out.println("O ano "+ano+" não é bissexto");
            }
        }
}
```