1 - Escreva uma função que imprima os números de 1 a 100 e, ao lado dos números múltiplos de 3 imprima “BUZZ”, ao lado dos múltiplos de 5 imprima “BIZZ” e ao lado dos múltiplos de 3 e 5 imprima “BUZZBIZZ”.

**public** **void** questao1() {

**for** (**int** i = 1; i <= 100; i++) {

System.***out***.print(i);

**if** (i % 3 == 0) {

System.***out***.print("BUZZ");

}

**if** (i % 5 == 0) {

System.***out***.print("BIZZ");

}

System.***out***.println("");

}

}

2 - Implemente a função **somaDeQuadrados** que retorna a soma de quadrados de uma lista.

**EXEMPLO:**

somaDeQuadrados([1,2,3,4,5])

retorna o número 55.

**public** Integer somaDeQuadrados(List<Integer> listaNumeros) {

**int** total = 0;

**for** (Integer numero : listaNumeros) {

total += (numero \* numero);

}

**return** total;

}

3 - A sequencia de Fibonacci é definida pela relação:

F*n* = F*n*−1 + F*n*−2, onde F1 = 1 e F2 = 1.

Assim, os primeiros 12 elementos são:

F1 = 1  
F2 = 1  
F3 = 2  
F4 = 3  
F5 = 5  
F6 = 8  
F7 = 13  
F8 = 21  
F9 = 34  
F10 = 55  
F11 = 89  
F12 = 144

Escreva uma função que retorna o primeiro elemento da sequência de Fibonacci com 5 dígitos.

**public** Integer questao3 () {

List<Integer> fibonacci = **new** ArrayList<Integer>();

fibonacci.add(1);

fibonacci.add(1);

**int** contador = 1;

**do** {

contador++;

fibonacci.add(fibonacci.get(contador - 1) + fibonacci.get(contador - 2));

} **while** (fibonacci.get(contador) <= 9999);

**return** fibonacci.get(contador);

}

4 - Estrutura de dados:

Arvore {

int id;

Arvore[] filhos;

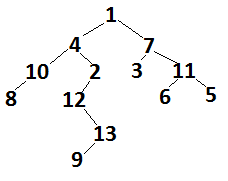
}

Escrever uma função que recebe uma árvore e um id e retorna um vetor com os ids do caminho desde o nó raiz até o nó passado como parâmetro.

int[] caminho\_arvore(Arvore a, int n);

Exemplos:

Arvore a:



caminho\_arvore(a, 9) retorna [1,4,2,12,13,9]

caminho\_arvore(a,1) retorna [1]

caminho\_arvore(a, 17) retorna []

caminho\_arvore(a, 4) retorna [1,4]

5 - Utilizando a função:

function acumular(combiner, nullValue, list)

{

if (list.length == 0) {

return nullValue;

}

var primeiro = list.removePrimeiro();

return combiner(primeiro, acumular (combiner, nullValue, list));

}

Implemente a função **somaDeQuadrados** que retorna a soma de quadrados de uma lista.

somaDeQuadrados([1,2,3,4,5])

retorna o número 55.

Neste caso a função acumular deve ser utilizada. A variável “combiner” é um “ponteiro para função”. A implementação da função “combiner” faz parte da solução.

6 - O termo n da sequência de “números triângulo” é dado pela fórmula:

Portanto os primeiros números da sequência são:

1, 3, 6, 10, 15, 21, 28, 36, 45, 55, ...

Convertendo cada letra de uma palavra no número correspondendo a posição no alfabeto e adicionando os valores, nós obtemos o valor de uma palavra. Por exemplo, o valor da palavra SKY é 19 + 11 + 25 = 55.

Se o valor da palavra é um “número triangulo”, então a palavra é chamada “palavra triângulo”.

Escreva uma função que retorna a posição da sequência dado uma palavra ou -1 se a palavra não for uma palavra triângulo.

int palavraTriangulo(string palavra);

palavraTriangulo(“SKY”) retorna 10.

palavraTriangulo(“ASDF”) retorna -1

**public** Integer retornarPosicaoPalavraTriangulo(String palavra) {

**int** somaPosicaoLetra = 0;

**char**[] letras = palavra.toUpperCase().toCharArray();

**for** (**char** letra : letras) {

somaPosicaoLetra += ((**int**) letra) - 64;

}

**int** aux = 1;

**int** numeroTrangulo = (aux \* (aux + 1) ) / 2;

**while**(somaPosicaoLetra > numeroTrangulo){

aux++;

numeroTrangulo = (aux \* (aux + 1)) / 2;

}

**if** (somaPosicaoLetra == numeroTrangulo)

{

**return** aux;

}

**return** -1;

}