1. (1pt) Qual das seguintes opções descreve corretamente a função principal da JVM?
   a) A JVM é responsável por compilar o código Java em instruções de baixo nível para a execução direta pelo hardware do computador.
   b) A JVM é um ambiente de execução que permite que o código Java seja executado de forma independente do sistema operacional e da plataforma de hardware.
   c) A JVM é uma biblioteca que fornece funcionalidades adicionais ao código Java, como suporte a banco de dados e interface gráfica.
   d) A JVM é um conjunto de ferramentas para depurar e testar aplicativos Java durante o processo de desenvolvimento.

2. (1pt) Qual das seguintes opções descreve corretamente a função da JRE?
   a) A JRE é um ambiente de execução que inclui a JVM e outras bibliotecas e ferramentas necessárias para executar aplicativos Java.
   b) A JRE é um ambiente de desenvolvimento completo que inclui o compilador Java e outras ferramentas de desenvolvimento.
   c) A JRE é uma biblioteca de código aberto que fornece funcionalidades adicionais ao código Java, como criptografia e compressão de dados.
   d) A JRE é uma ferramenta de análise estática de código que verifica a conformidade do código Java com as especificações e padrões da linguagem.

3. (4pts) A classe mostrada abaixo é denominada Ponto.java. Ela representa um ponto no plano cartesiano. Como pode ser visto, esta classe é composta por dois atributos e dois métodos: um construtor e um método estático que calcula a distância entre dois pontos passados como parâmetro.

```
import java.lang.Math;

public class Ponto
{
   public double x;
   public double y;

   public Ponto(double x, double y)
   {
       this.x = x;
       this.y = y;
   }

   public static double Distância(Ponto p1, Ponto p2)
   {
       double dist = Math.sqrt(Math.pow((p1.x - p2.x), 2) +
Math.pow((p1.y - p2.y), 2));
       return dist;
   }
}
```

a) (0,6pt) Na classe apresentada temos o uso de diversas variáveis: this.x, this.y, x, y, p1, p2 e dist. Classifique cada variável quanto ao seu escopo (de instância, local ou estática) e quanto ao seu tipo (primitivo ou referência).

this.x e this.y = instância e primitivo
x e y = local e primitivo
p1 e p2 = local e referências
dist = local e primitivo

b) (0,3pt) Um círculo pode ser desenhado num plano cartesiano a partir de duas informações: um ponto que determina seu centro e um valor positivo que indica o tamanho do seu raio, isto é, a distância do centro até os limites da circunferência. Observe a imagem abaixo. Crie uma classe Círculo contendo dois atributos: um ponto central (da classe Ponto) e um raio.


```
class Circulo {

    Ponto ponto;
    double raio;
}
```

c) (1,2pt) Explique se há a necessidade de se aplicar o encapsulamento em algum atributo. Se sim, crie os métodos de acesso (getter) e modificação (setter). Justifique a sua resposta.

Setter e getter no raio, ele deve ser double. Raio deve ser maior que 0.

d) (0,6pt) Crie um método construtor para essa classe que inicializa seus atributos internos de acordo com os parâmetros passados.

Método construtor deve estar coerente com os atributos da classe e utilizar o setter para definir o raio. O raio deve estar private.

e) (0,8pt) Crie dois métodos denominados calcularArea e calcularPerimetro. Esses métodos são chamados **através de uma instância** e retornam a área e o perímetro de um círculo, respectivamente. A área é dada por πr^2 e o perímetro é dado por $2\pi r$.

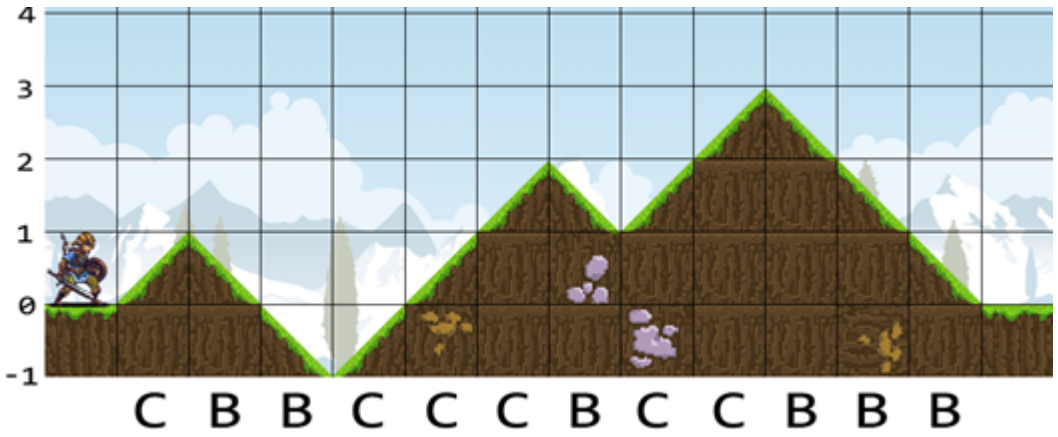
Ambos os métodos devem fazer o cálculo correto, não ter argumentos (usar o raio da instância) e retornar o resultado do tipo double. Foi aceito tanto Math.PI como 3.14 para o pi.

f) (0,5pt) Crie um método que indique se dois círculos se interceptam, ou seja, se eles se "encostam" em algum ponto. O método é um método estático que recebe como argumento dois Círculos e retorna a variável booleana true se os círculos se interceptam ou false, caso contrário. A imagem abaixo mostra um exemplo onde dois círculos estão se interceptando.

O método deve fazer o cálculo correto, ser estático, receber ambos os círculos como argumento e retornar uma variável booleana.

4. Link é um ávido alpinista e explorador do mundo. Ele rastreia suas escaladas meticulosamente, prestando muita atenção a pequenos detalhes como topografia. Em cada escalada ele faz anotações indicando todos os **passos** que deu: se era um passo para cima, C, ou se era um passo para baixo, B. Ao término de uma escalada, Link tinha em suas mãos uma sequência de C's e B's, denotando o percurso realizado. Além disso, as escaladas de Link sempre começam no nível do mar e cada passo para cima ou para baixo representa uma mudança de unidade na altitude.

<u>Por exemplo</u>: se o percurso que Link realizou foi tal que percurso=[CBBCCCBCCBBB], então ele alcançou três picos: o primeiro foi alcançado no primeiro passo e tem altitude 1, o segundo foi alcançado no sexto passo e tem altitude 2 e o terceiro foi alcançado no nono passo e tem altitude 3. Além disso, neste exemplo, ele retornou ao nível do mar ao final da escalada. A imagem abaixo ilustra graficamente o percurso que ele realizou.



De acordo com o que foi dito, escreva o seguinte programa:

a) (0,4pt) Crie uma classe chamada Percurso, que possui como atributo uma String para o percurso. Se durante a construção dos métodos da classe você sentir a necessidade de criar outros atributos, não há problemas.

```
class Percurso {
        private String caminho;
}
```

b) (0,8pt) Explique se há a necessidade de se aplicar o encapsulamento neste atributo. Se sim, crie os métodos de acesso (getter) e modificação (setter). Justifique a sua resposta.
Encapsulamento na String. O setter deve verificar se a String só contém Cs e Bs.

c) (0,4pt) Crie um método construtor para essa classe.
Construtor com setCaminho(caminho);

d) (0.4pt) Crie um método chamado nivelMar que retorna uma variável booleana indicando se Link retornou ou não ao nível do mar ao final do percurso.
O método deve ser chamado sem argumentos, fazer o cálculo correto da altura e retornar uma variável booleana.

e) (0.4pt) Crie um método chamado picoMaisAlto que retorne a altura do pico mais alto visitado.
O método deve ser chamado sem argumentos, fazer o cálculo correto da altura e retornar uma variável inteira.

f) (0.4pt) Crie um método chamado numeroPicos que retorne quantos picos foram visitados.
O método deve ser chamado sem argumentos, fazer o cálculo correto do número de picos e retornar uma variável inteira.

g) (0.4pt) Crie um método chamado passoPico que imprime na tela depois de quais passos o Link alcançou um pico.

O método deve ser chamado sem argumentos, fazer o cálculo correto dos picos e imprimir na tela o resultado.

h) (0.4pt) No programa principal, instancie um objeto dessa classe com o percurso dado pelo usuário. Se a string não respeitar o formato de percurso que foi descrito, isto é, apenas composta por C's e B's, o programa deverá pedir para que o usuário entre com uma nova string que seja válida. Utilize os métodos e/ou atributos da sua classe para fazer essa validação.
O método setter da classe deve ser utilizado em um loop para fazer essa verificação.

i) (0.4pt) Chame cada um dos métodos criados e imprima na tela o resultado correspondente.
Instanciar o objeto e chamar os métodos correspondentes.

Um exemplo de entrada e saída é mostrado abaixo:

Entrada:	**Saída:**
Entre com o percurso: CBBCCxBaCC O percurso digitado não é válido! Entre com o percurso: CBBCCCBCCBBB	Link, você retornou ao nível do mar e deu 12 passos. 3 picos foram visitados. O ponto de altitude mais alto alcançado foi 3. O 1º pico foi alcançado no passo 1. O 2° pico foi alcançado no passo 6. O 3º pico foi alcançado no passo 9.