Teste de Lógica & Linguagem de programação (C#)

**Instruções:**

1. **Leia atentamente o enunciado de cada exercício**.
2. Preencha as respostas dos exercícios logo após o final de cada enunciado, respectivamente. Caso prefira, envie os arquivos gerados pelos exercícios da IDE como anexo, identificando-os pelo nome do exercício;
3. Caso não saiba como responder, deixe em branco ou justifique qual foi a sua dificuldade, evite criar soluções que não consiga ter embasamento ou que não saiba explicar;
4. Utilize sempre a linguagem de programação C# nos exercícios, exceto se o enunciado pedir diferente;
5. Procure fazer “o básico bem feito”, não perca tempo com respostas mirabulosas, queremos avaliar a sua forma de raciocínio, clareza de código e capacidade de resolver problemas.
6. **Teste os exercícios antes de enviá-los.**

**QUESTÕES**

# 1) Prática - Primo

Um número primo é aquele que é divisível apenas por **um** e por **ele mesmo**.

Obs: O número 1 não é primo.

**🡺 Escreva uma função otimizada que, dado um número inteiro positivo, retorne true se o número for primo ou false caso contrário, com o menor número de iterações possível.**

Imprima o resultado em tela da seguinte forma:

*“O número* ***num*** *é primo. Número de iterações necessárias:* ***count****”*

*ou*

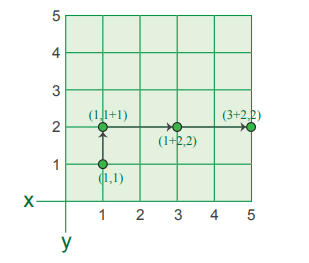
*“O número* ***num*** *não é primo. Número de iterações necessárias:* ***count****”*

# 2) Prática - Movimento do bot

Existe um bot localizado em um par de coordenadas inteiras, (x, y). Ele pode ser movido para um outro par de coordenadas. Embora o bot possa se mover quantas vezes quiser, ele só pode fazer os dois tipos de movimentos a seguir:

1. Da posição (x, y) para a posição (x + y, y).
2. Da posição (x, y) para a posição (x, x + y).

Por exemplo, se o bot começa em (1, 1), ele pode fazer a seguinte sequência de movimentos: (1, 1) → (1, 2) → (3, 2) → (5, 2). Observe que o movimento sempre será para cima ou para a direita.



**🡺 Escreva uma função que, dadas as coordenadas iniciais e finais, determine se o bot pode alcançar as coordenadas finais de acordo com as regras de movimento.**

Descrição da função:

A função deve retornar true se o bot puder atingir seu objetivo, caso contrário, retorne false.

A função tem o(s) seguinte(s) parâmetro(s):

x1: valor inteiro, coordenada x inicial

y1: valor inteiro, coordenada y inicial

x2: valor inteiro, coordenada x final

y2: valor inteiro, coordenada y final

# 3) Prática - GPS

Você é um desenvolvedor responsável pela manutenção de um software GPS famoso no mercado. Você recebeu a tarefa de calcular a melhor rota possível entre duas cidades distintas. Seu programa receberá informações relacionadas às cidades e estradas que os conectam entre si. Além disso, este famoso software de GPS pode se comunicar com satélites para receber dados em tempo real sobre o tráfego fluxo das cidades. Portanto, seu programa precisará considerar algum tempo gasto durante o tráfego intenso, quando tentando deixar algumas cidades.

Para simplificar o desenvolvimento de software, considere o seguinte:

* cada cidade será nomeada com apenas 1 letra minúscula (de a a z);
* cada estrada liga duas cidades, e o tráfego pode ir em ambas as direções;
  + exemplo: em uma estrada que liga as cidades aez, o tráfego pode ir de a a z e vice-versa;
* para cada estrada, será fornecido um número inteiro t representando o tempo necessário para o veículo atual viajar toda a estrada, em horas;
* cidades nomeadas com uma vogal (a, e, i, o ou u), estão atualmente com tráfego intenso de saída - isso causará um atraso de 5 horas na saída da cidade

Dada uma lista de cidades, todas as estradas que as conectam, uma cidade de partida e, finalmente, a cidade de destino, seu programa precisa produzir o tempo mínimo necessário para a viagem.

**Entrada**

A entrada começa com uma linha contendo um inteiro T indicando o número de casos de teste. Para cada caso de teste, a entrada acontecerá da seguinte forma:

* uma linha contendo um inteiro C, indicando o número de cidades;
* uma linha contendo os nomes de todas as cidades, que são nomeadas com uma letra minúscula, separadas por um espaço;
* uma linha contendo um inteiro R, indicando o número de estradas;
* R linhas, cada uma contendo os seguintes dados, separados por um espaço:
  + uma letra minúscula representando uma cidade, em uma das pontas da estrada; uma letra minúscula representando a cidade do outro lado da estrada;
  + um número inteiro t representando o tempo, em horas, necessário para percorrer toda a estrada (independentemente de a direção do tráfego);
* Finalmente, a última linha de um caso de teste conterá duas letras minúsculas, separadas por um espaço, indicando uma cidade de partida e uma cidade de destino.

**Resultado**

Um inteiro M indicando o tempo mínimo necessário para viajar da cidade de partida à cidade de destino.

**Observações:**

* **Utilize o arquivo enviado em anexo (EntradaGPS.txt) como entrada para o programa**.
* Use os recursos de Programação Orientada a Objeto (POO) da forma que julgar mais adequada para a solução do problema.
* Realize a implementação do método List<int> GpsFitSolution(String FilePath) cujo a entrada é caminho para um arquivo e a saida é uma lista de inteiros conforme detalhado na especificação de entrada e saida.

**Exemplo:**

Chart, diagram

Description automatically generated

**Entrada:**

3

4

z a b c

4

z a 1

z b 2

a c 2

b c 1

z c

4

z a b c

4

z a 1

z b 2

a c 2

b c 1

z a

4

z a b c

4

z a 1

z b 2

a c 2

b c 1

z b

**Saída:**

3

1

2

4) Prática - Supermercado

O supermercado Baixo Preço teve uma queda muito grande em suas vendas na pandemia e precisou aderir a outros métodos para se reerguer, e pensaram na criação de um e-Commerce com seus produtos. Você foi contratado por eles para desenvolver uma SPA (Single Page Application) em Angular 2+ para resolver esse problema.

**Requisitos necessários:**

* Necessário ter uma lista com todos os produtos.
* Ao selecionar um produto que não tem estoque, apresentar para o usuário um aviso: (“Produto sem estoque. Tente novamente mais tarde”, e um botão “OK” nesse aviso).
* Implementar uma parte que contém o carrinho de compras e ao adicionar esse produto da lista, já atualizar o carrinho.
* Finalizar a compra
* Código limpo e legível
* Não é necessário se preocupar com banco de dados, pode utilizar banco de dados em memória.

**Requisitos desejáveis:**

* Utilização de @mixins, @includes, diretivas em estilos.
* Testes unitários com injeção de dependências.
* EventEmitter.
* Exemplos de Async/Await

Exemplo (**Não é necessário fazer igual, use sua criatividade** 😊):

Graphical user interface, application

Description automatically generated