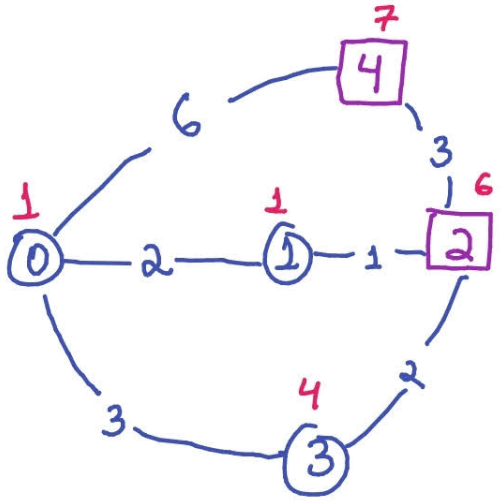
Trabalho 03 - Quest Explosiva

Data de entrega: 18/12/2022
Importante:

- **Não** olhe códigos de outros ou da internet. Também não mostre ou publique o seu.
- Em caso de plágio, fraude ou tentativa de burlar o sistema será aplicado nota 0 na disciplina aos envolvidos.
- Alguns alunos podem ser solicitados para explicar com detalhes a implementação.
- Passar em todos os testes do run.codes não é garantia de tirar a nota máxima. Sua nota ainda depende do cumprimento das especificações do trabalho, qualidade do código, clareza dos comentários, boas práticas de programação e entendimento da matéria demonstrada em possível reunião.
- Você deverá submeter, até a data de entrega, o seu código na plataforma run.codes. Pode submeter quantas vezes precisar, só será considerado a última.
- Seu trabalho deverá ser implementado em linguagem C. O seu código deverá rodar em 1 segundo para cada instância.
- Você receberá arquivos de um TAD (Tipo Abstrato de Dados) Grafo, um main.c e um algoritmo.c onde você deverá implementar seu algoritmo. Já que a TAD Grafo foi fornecida, você não deve implementar outra, e deve usar a fornecida. Você deverá alterar e submeter apenas o arquivo algoritmo.c.

Considere o seguinte jogo: Você é deixado num ponto de origem de um mapa. E você precisa alcançar um baú, dos vários que existem no mapa, que lhe dará alguns itens necessários para a sua sobrevivência. O seu objetivo é encontrar o baú que você consegue chegar mais rápido e o melhor caminho até ele. Mas é claro que existe um desafio extra, alguns pontos no mapa explodem à medida que o tempo passa.



Considere o exemplo à esquerda, o mapa é passado como um grafo, você sempre começa no vértice 0, nos vértices 2 e 4 marcados com quadrados, existem baús. Em vermelho, acima de cada vértice, tem o tempo que cada vértice vai explodir, por exemplo, o vértice 0 vai explodir no tempo 1, enquanto o vértice 4 vai explodir no tempo 7. No meio de cada aresta tem um número, que é o tempo que você leva para percorrer aquela aresta, por exemplo você poderia ir do vértice 0 diretamente para o vértice 4 através da aresta que leva tempo 6. Entretanto existem caminhos mais rápidos para chegar em algum baú. Você poderia tentar seguir o caminho 0 - 1 - 2, que levaria tempo 3, entretanto ao chegar no vértice 1 depois de viajar por 2 unidades de tempo o vértice 1 já teria explodido, e você morreria. Por outro lado, o caminho 0 - 3 - 2 é seguro, apesar de levar mais tempo, e esse é o melhor caminho a ser seguido.

Você receberá um grafo que representa o mapa. A entrada é dada pelo número *n* de vértices, depois pelo número *m* de arestas, seguido por uma linha com *n* valores que são os tempos de explosão de cada vértice, depois uma linho com *n* valores binárias indicando para cada vértice se contém um baú ou não. Depois seguem *m* linhas com os extremos de cada aresta e o tempo necessário para percorrê-la.

```
5 6
1 1 6 4 7
0 0 1 0 1
0 1 2
0 3 3
0 4 6
1 2 1
2 3 2
2 4 3
```

Você deverá implementar a função que recebe um vetor “caminho” (já pré alocado) e um apontador de inteiro “num_vertices_caminho”, que deverá conter os vértices a serem visitados e a quantidade de vértices desse caminho. No exemplo acima num_vertices_caminho deveria ser 3 e o vetor deveria conter nas 3 primeiras posições os valores 0, 3 e 2. Note que tanto o vértice origem como o vértice do baú devem aparecer no caminho. O programa main faz então uma verificação e imprime uma mensagem e o tempo da sua solução.

```
Sua solucao funciona e tem custo: 5
```