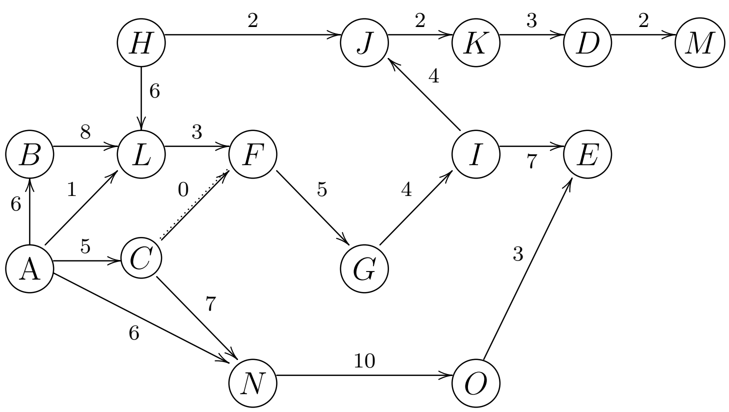
Folha 4

Aplicação dos algoritmos sobre grafos (pesquisa em largura/profundidade, planeamento de tarefas (CPM), algortmos de Prim, Dijkstra, Kruskal, e variante de Dijkstra para capacidade máxima a partir de uma origem. Estruturas para filas de prioridade (heapmin e heapmax) descritas nas aulas.

1. Um dado projeto envolve um certo número de tarefas. É conhecida a duração de cada tarefa. É conhecida a relação de precedência entre tarefas. Se a tarefa x **precede** a tarefa j então só se pode dar início à tarefa y depois de a tarefa i estar concluída. Não há partilha de recursos entre tarefas, podendo várias tarefas estar a decorrer em simultâneo, desde que sejam respeitadas as condições sobre precedências. Todas as tarefas do projeto têm de ser realizadas.

Considere a instância representada pela rede desenhada abaixo, segundo o modelo "arco-actividade".



Cada tarefa corresponde a um arco do grafo. As tarefas que têm fim num nó do grafo precedem as tarefas que têm início nesse nó. O valor em cada arco representa a duração da tarefa correspondente (em dias). A tarefa (C, F) com duração 0 é ficticia. Essa tarefa não faz parte do projecto mas estabelece precedências.

a) Insira um nó *fim* e ligue a esse nó todos os nós que têm grau de saída zero, por introdução de tarefas fictícias com duração 0.

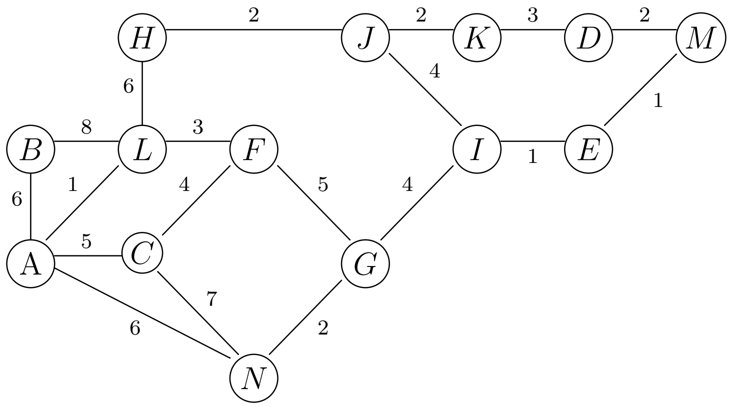
b) Por aplicação do algoritmo dado nas aulas, para obter o caminho mais longo num DAG, determine a duração mínima do projecto representado e a data de início mais próxima para cada tarefa. Durante a aplicação do algoritmo deve manter em cada nó uma estimativa da data de início mais próxima para as tarefas que têm início nesse nó (essa estimativa será 0 inicialmente mas vai sendo adiada à medida que o algoritmo vai propagando a informação sobre precedências). Use uma fila para suportar o conjunto de nós cujas precedências já estão tratadas e indicar a ordem de saída dos nós da fila. E caso de empate na entrada, coloque-os na fila por ordem crescente de identificador (ou seja, ordem alfabética).

c) Por análise para trás, marque cada nó com a data mais afastada em que seria possível concluir todas as tarefas que terminam nesse nó (sem atrasar o projecto). A seguir, determine a data mais afastada em que pode dar início a cada uma das tarefas.

d) Identifique as tarefas críticas.

e) Para cada tarefa indique: a folga total (diferença entre a sua data de início mais afastada e sua data de início mais próxima), a folga livre (folga que pode usar mas que permite às tarefas seguintes ainda terem início na sua data de início mais próxima).

2. Seja G o grafo seguinte em que os pesos associados aos ramos representam **distâncias**.



a) Aplique o algoritmo de Dijkstra para determinar um caminho mínimo de J para cada um dos restantes vértices do grafo. Para cada vértice $v \neq J$, deve indicar a distância mínima $\delta(J, v)$ e o vértice $prec[v]$ que precede v no caminho encontrado pelo algoritmo. Acrescente informação ao grafo que permita compreender como obteve o resultado.

b) Aplique o algoritmo de Prim para construir uma árvore de cobertura (i.e., árvore geradora) para G de peso total mínimo, partindo de J. Para cada nó $v \neq J$, deve indicar o vértice $pai[v]$ a que v ficou ligado nessa árvore. Acrescente informação ao grafo que permita compreender como obteve o resultado.

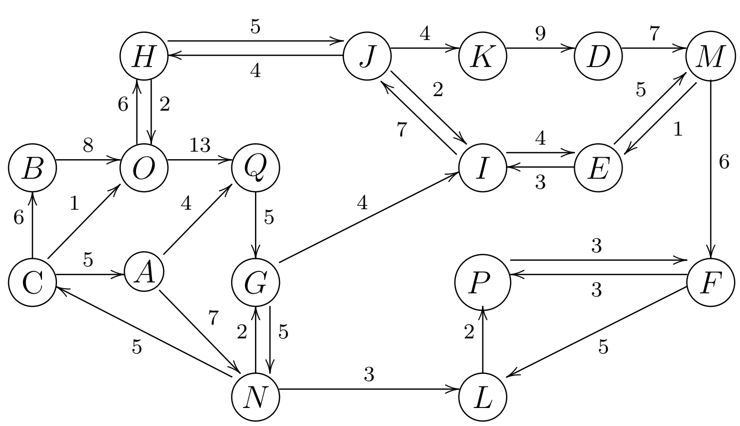
c) Justifique que o ramo $\{I, J\}$ não pertence a **nenhuma** árvore de cobertura de G com peso total mínimo.

d) Suponha que pretende determinar o caminho de comprimento mínimo entre um nó s e cada nó v de um grafo $\mathcal{G} = (V, E, d)$ não dirigido **conexo**, tal que $v \neq s$. Tem disponível uma função ALGOPRIM$(\mathcal{G}, pai, s)$ que lhe permite construir uma árvore de cobertura de $\mathcal{G}$ de peso mínimo, com raíz em s, dando como resultado o vetor $pai[v]$ que identifica o vértice a que v ficou ligado na árvore. Justifique que o algoritmo seguinte não resolve corretamente o problema.

```
CAMINHOSMINIMOS(s, G)
| ALGOPRIM(G, pai, s);
| Para cada v ∈ V \ {s} fazer
|     ESCREVECAMINHO(s, v, pai);
```

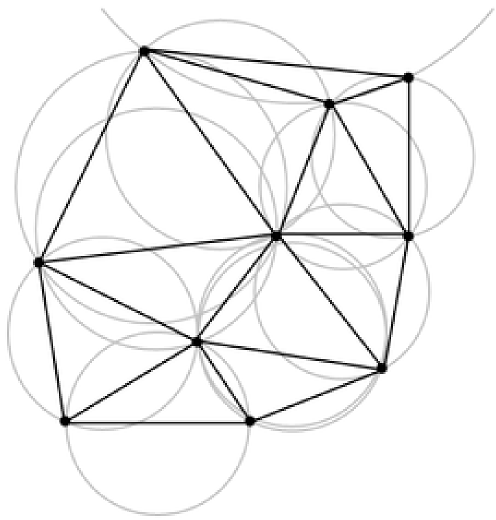
```
ESCREVECAMINHO(s, v, pai)
| Se v ≠ s então
|     ESCREVECAMINHO(s, pai[v], pai);
| escrever(v);
```

3. Na rede seguinte, os valores nos ramos representam **capacidades**. Por aplicação da variante dada do algoritmo de Dijkstra para este problema, determine um **caminho de capacidade máxima de** C **para** M. A capacidade do caminho é o mínimo das capacidades dos ramos que o constituem.



4. *(Este problema está relacionado com "Freckles":* `https://onlinejudge.org/external/100/10034.pdf` *)*

Dados n pontos no plano, em posição geral (o que neste caso significa que não há 4 ou mais pontos sobre a mesma circunferência nem estão todos em linha reta), pretendemos definir um conjunto de ligações de modo que qualquer ponto seja acessível de qualquer outro ponto e o comprimento total das ligações seja mínimo. É conhecido que, entre as $n(n-1)/2$ ligações que podiamos considerar, apenas um número $O(n)$ é relevante para a resolução do problema, sendo tais ligações relevantes arestas de uma *triangulação de Delaunay* determinada pelos pontos. (`https://en.wikipedia.org/wiki/Delaunay_triangulation`)



Dadas as coordenadas dos pontos, a determinação da triangulação de Delaunay é um problema geométrico interessante que pode ser resolvido por um algoritmo com complexidade temporal $O(n \log n)$, mas que está fora do âmbito do programa de CC2001.

Neste exercicio, queremos apenas determinar quais seriam as ligações a preservar na instância representada à esquerda, sendo dada já sua triangulação de Delaunay. Escolha um algoritmo adequado para resolver o problema e aplique-o à instância.

5. Recordar o problema "Bacalhaus Congelados" (Mooshak DAA2324_Vol2). Suponha que é dada a informação sobre a rede, como no enunciado do problema.

Usando pseudocódigo, escreva um algoritmo para resolução de cada um dos problemas seguintes. Em cada caso, indique dados para instâncias do problema e resolva-as por aplicação do algoritmo que apresentou.

a) Verificar se é possível efectuar o transporte, e em caso afirmativo, indicar um percurso.

b) Determinar um percurso que tenha comprimento mínimo, sendo o comprimento definido pelo número de troços do percurso.

c) Determinar um percurso que tenha custo mínimo, sendo esse custo dado pela soma dos custos dos troços que forem usados no percurso.

d) (*) Determinar os percursos melhores do ponto de vista do custo total e do número de ramos que envolvem, supondo que se procura minimizar simultaneamente esses dois parâmetros.

6. Seja q uma variável do tipo `HEAPMIN*` dado nas aulas (para C), com q $\rightarrow$ `sizeMax` $=$ q $\rightarrow$ `size` $= 16$. Admita que o conteúdo das posições 1 a 16 do *array* a da estrutura *heap de mínimo* apontada por q é:

2	3	5	15	4	7	12	22	17	13	23	21	9	30	28	50
3	7	4	10	1	12	15	5	8	14	2	9	13	16	11	6

a) Quais os valores de q $\rightarrow$ `pos_a`[15], q $\rightarrow$ `pos_a`[2], PARENT(9), LEFT(7), RIGHT(12), e q $\rightarrow$ `a`[7]`.vert`?

b) Indique a sequência de trocas sobre q $\rightarrow$ `a`[ ], q $\rightarrow$ `pos_a`[ ], q $\rightarrow$ `size` e q $\rightarrow$ `sizeMax` realizadas na chamada de `extractMin(q)`.

c) Sem considerar a alteração efetuada na alínea anterior, indique o estado de $\texttt{q} \to \texttt{a}[\,]$, $\texttt{q} \to \texttt{size}$, $\texttt{q} \to \texttt{sizeMax}$, e do *array* $\texttt{x}[\,]$, após a execução do bloco seguinte, se inicialmente $\texttt{i} = 0$.

```
while( ! heap_isEmpty(q) ) {
   x[i] = extractMin(q);
   i = i+1;
}
```

d) Sendo k o número de elementos na *heap*, qual é complexidade desse bloco?

7. Suponha que um vetor v tem nas posições 1 a 10 o conteúdo seguinte.

10	-3	4	2	7	5	-2	18	-10	5

a) Apresente os passos da construção de uma *heap de máximo*, a partir de v, segundo a estrutura HEAPMAX descrita nas aulas. Deve **seguir o código dado na implementação** DAA2324_DataStructures para a construção dessa heap.

b) Indique como mudam as estruturas quando se efetua *extract_max* e qual é o valor de retorno

c) Em seguida, efetue *increaseKey(9,17)*. Indique como são alteradas as estruturas de dados.

8. Suponha que um vetor v tem nas posições 1 a 10 o conteúdo seguinte.

10	-3	4	2	7	5	-2	18	-10	5

a) Apresente os passos da construção de uma *heap de mínimo*, a partir de v, segundo a estrutura HEAPMIN descrita nas aulas. Deve **seguir o código dado na implementação** DAA2324_DataStructures para a construção dessa heap.

b) A seguir, efetue duas operações de *extract_min*.

c) Depois, efetue *decreaseKey(4,-1)*, e indique como variam as estruturas.