

N.º Nome

Atenção: Não serão prestados esclarecimentos durante a prova. Leia pf as perguntas com atenção. As respostas devem ser assinaladas na tabela seguinte, usando **LETRAS MAIÚSCULAS**, sem rasuras. Cotação: 8 valores. Três respostas erradas, descontam uma certa. Ausência de resposta não desconta.

Respostas:

1	2	3	4	5	6	7	8	9	10	11	12
B	C	B	B	D	D	B	C	C	D	B	A

1. Considere o problema de dar um troco Q em moedas a um cliente numa mercearia portuguesa. As moedas disponíveis estão todas na caixa registadora da loja. Recorde que, os valores das moedas (em cêntimos) são 1, 2, 5, 10, 20, 50, 100 e 200. A pessoa que está na caixa sabe que, com as moedas de 1 e 2 cêntimos que tem, conseguiria formar Q , mas quer minimizar o número de moedas que entregará. Qual das afirmações seguintes é **falsa**?

- a) A solução ótima pode não incluir moedas do valor mais alto não superior a Q que tem na caixa.
- b) É conhecido que a estratégia *greedy* para “coin change” obtém a solução ótima deste problema.
- c) Se depois de dar o troco ainda tiver moedas de 20, não incluiu duas ou mais moedas de 10 no troco.
- d) Poderia aplicar um algoritmo de programação dinâmica para obter a solução ótima.

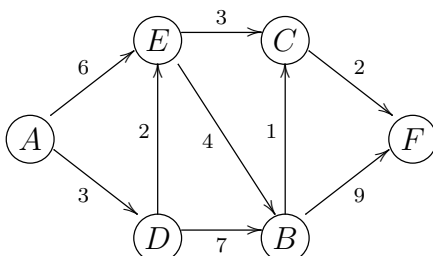
2. Qual das afirmações seguintes sobre grafos **não dirigidos**, com $n \geq 3$ nós e $m > 0$ ramos, é **incorreta**?

- a) Por aplicação de BFS (ou DFS) a partir de qualquer nó v podemos decidir se o grafo é conexo.
- b) Podemos verificar em tempo $O(n + m)$ se uma dada permutação dos nós define um ciclo de Hamilton.
- c) O algoritmo de Kosaraju-Sharir deve ser usado para identificar as componentes conexas.
- d) Existe um algoritmo com complexidade $O(n + m)$ para determinar os graus dos nós.

3. Queremos transferir até n ficheiros do computador para uma *pendrive* de capacidade de c megabytes. O ficheiro i tem tamanho de t_i megabytes. Na expectativa de maximizar o número de ficheiros transferidos, ordenámos os ficheiros por ordem crescente de tamanho, e transferimos os primeiros até atingir a capacidade da pendrive. Qual estratégia de desenho de algoritmos estamos a utilizar?

- a) Programação dinâmica b) Gananciosa c) Pesquisa com retrocesso d) Divisão-e-conquista

4. O projeto representado pela rede de atividades seguinte (modelo arco-atividade) **será concluído o mais cedo possível**. Não há partilha de recursos. É verdade que:



- a) A data de início mais próxima para CF é 8.
- b) Nenhuma tarefa estará a decorrer no instante 20.
- c) A data de início mais afastada para EC é 8.
- d) CF perde parte da sua folga total se EC não começar na sua data mais próxima.

N.º Nome

5. Considerando a matéria lecionada sobre complexidade de algoritmos e também sobre complexidade de problemas, é verdade que:

- a) determinar o caminho mais longo num DAG pesado é um problema NP-difícil (*NP-hard*).
- b) o algoritmo de Dijkstra pode ser aplicado a grafos com pesos negativos.
- c) é conhecido que existem problemas da classe NP que não são da classe P.
- d) a distância de edição (Levenshtein) entre duas *strings* pode ser calculada por um algoritmo polinomial.

6. Para obter o k -ésimo menor elemento num segmento $\text{arr}[\text{low}..\text{high}]$ de um *array* (que pode ser alterado) podemos ordená-lo e retornar $\text{arr}[\text{low}+k-1]$. A função *select* apresentada abaixo segue outra ideia. A função *partition* altera $\text{arr}[\text{low}..\text{high}]$, colocando os elementos menores do que o *pivot* nas posições $\text{low}..\text{pi}-1$, deixando o *pivot* na posição pi e os restantes elementos nas posições seguintes. O *pivot* é um elemento escolhido inicialmente em $\text{arr}[\text{low}..\text{high}]$, que pode ser $\text{arr}[\text{low}]$. Que estratégia de desenho de algoritmos implementa *select*?

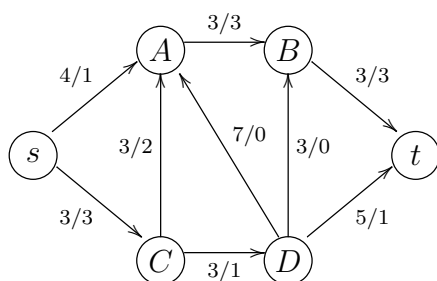
```
select(arr[], low, high, k) {           // assumes low <= high and 1 <= k <= high-low+1
    if (k == 1) return arr[low];
    pi = partition(arr, low, high);
    m = pi-low+1;                       // number of elements in arr[low..pi]
    if (k == m) return arr[pi];
    else if (k < m) return select(arr, low, pi-1, k);
    else return select(arr, pi+1, high, k-m);
}
```

- a) Programação dinâmica b) Gananciosa c) Pesquisa com retrocesso d) Divisão-e-conquista

7. Considere o problema “*minimum cardinality vertex cover*” que, dado um grafo não dirigido $G = (V, E)$, visa determinar $S^* \subseteq V$, com $|S^*|$ mínimo e tal que qualquer ramo de E tem algum extremo em S^* , sendo $|S^*|$ o número de elementos de S^* . Face à matéria lecionada, é verdade que:

- a) Não existe e não pode existir um algoritmo polinomial que resolva todas as instâncias.
- b) Existe um algoritmo polinomial que determina uma cobertura S com $|S| \leq 2|S^*|$.
- c) O problema de decisão associado não pertence NP.
- d) Para obter S^* , podemos seleccionar sucessivamente o nó que cobre mais ramos ainda não cobertos.

8. Considere a rede de fluxo seguinte, onde c/f são pares capacidade/fluxo, e s e t são a origem e destino. O algoritmo de Edmonds-Karp encontra o caminho (s, A, C, D, t) para aumento de fluxo e:



- a) aumenta o fluxo de 5 (cinco) unidades.
- b) aumenta o fluxo de 1 (uma) unidade.
- c) reduz o fluxo no ramo CA para zero.
- d) aumenta $|f|$ de 3 (três) unidades.

N.º

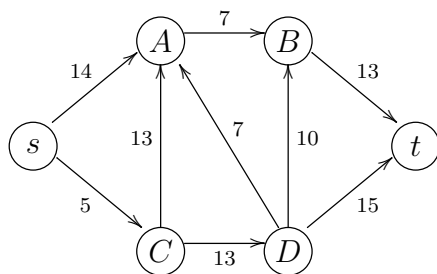
Nome

9. Um texto utiliza apenas os caracteres *sp* (espaço), *A*, *E*, *R*, e *V*, os quais ocorrem o número de vezes indicado na tabela. O código de cada caracter é uma sequência em binário. Qual é o custo mínimo de uma codificação do texto, utilizando um sistema de codificação de **tamanho fixo**?

caracter	sp	A	E	R	V
ocorrências	1	3	1	2	1

- a) 18 b) 21 c) 24 d) 27

10. Considere a rede de fluxo seguinte. Os valores indicados nos ramos são capacidades. O caminho de *s* para *t* com capacidade máxima:

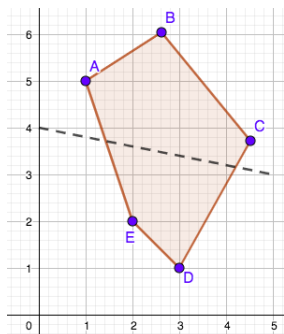


- a) passa em *D* e tem capacidade 15.
b) tem capacidade 45.
c) passa em *C*.
d) tem capacidade 7.

11. Considerar um problema de colocação de candidatos em empresas, que têm preferências iguais entre elas mas os candidatos não (e podem não incluir algumas empresas). A seriação acordada pelas empresas e as listas das preferências dos candidatos estão ordenadas estritamente. Queremos uma solução em que nenhum candidato seja ultrapassado por outro com classificação inferior. Para resolver o problema, devemos aplicar:

- a) pesquisa com retrocesso (*backtracking*).
b) pesquisa sequencial, colocando os candidatos pela ordem definida pelas empresas, sempre atribuindo ao candidato a sua primeira preferência, nas que ainda têm vagas, se existir alguma.
c) a extensão do algoritmo de Gale-Shapley, orientada pelos candidatos, partindo de qualquer candidato.
d) o algoritmo de Edmonds-Karp para obter um emparelhamento de cardinal máximo num grafo bipartido.

12. Considere um problema de **minimização de** $z = x + 5y$, com restrições lineares sobre (x, y) que determinam o polígono apresentado. Os pontos *A*, *E*, e *D* têm coordenadas inteiras, *B* tem $y = 6$ e *C* não tem coordenadas inteiras. A reta a tracejado é curva de nível da **função objetivo**. É verdade que:



- a) Se se arrancar o Simplex com a solução básica admissível correspondente a *B*, duas variáveis de desvio serão candidatas a entrar para a base.
b) Se se arrancar o Simplex, partindo de *A*, obtém-se a solução ótima numa única iteração (troca de base).
c) Se se procurar soluções ótimas com x e y inteiros, o ótimo da relaxação linear não é solução.
d) O valor de z da solução ótima linear excede o da solução ótima inteira.