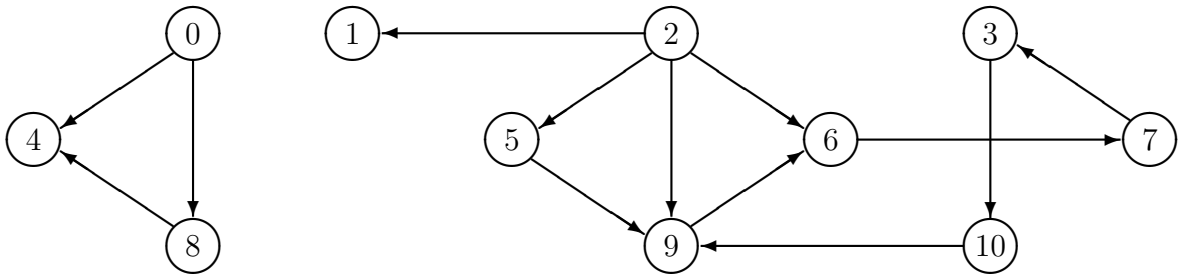
1º Teste de Análise e Desenho de Algoritmos

Departamento de Informática
Universidade Nova de Lisboa
18 de Abril de 2015

1. Suponha que se executa o algoritmo *isAcyclic* (que testa se um grafo orientado é acíclico) com o grafo G esquematizado na figura.



Assuma que o saco *ready* tem disciplina FIFO (*first-in, first-out*) e que os métodos *nodes* e *outAdjacentNodes* iteram sempre os vértices por ordem crescente. Por exemplo, $G.outAdjacentNodes(2)$ produz os vértices 1, 5, 6 e 9 (por esta ordem). Indique:

   (a) [1 valor] o resultado do algoritmo (o valor retornado pelo método *isAcyclic*);

   (b) [2 valores] os vértices que são removidos do saco pela ordem com que são removidos;

   (c) [1 valor] o maior número de vértices presentes simultaneamente no saco (ou seja, o maior valor de *ready.size*()).

2. Dados:

   um grafo orientado G e dois vértices distintos, o e d,

   pretende-se descobrir se há (algum) caminho de o para d em G.

   Para exemplificar, no grafo da Pergunta 1, há caminho de 2 para 10 (por exemplo, 2 6 7 3 10), mas não há caminho de 1 para 7.

   (a) [4 valores] Apresente uma função booleana (em pseudo-código) que recebe um grafo orientado G e dois vértices, o e d, e retorna *true* se, e só se, existir algum caminho de o para d em G. Assuma (sem testar) que os vértices dados são distintos.

   (b) No cálculo da complexidade temporal do seu algoritmo, tanto no pior caso, como no melhor, considere que o número de arcos do grafo é muito superior ao número de vértices.

      (i) [1 valor] Qual é a complexidade temporal do seu algoritmo, no pior caso? Justifique e indique uma situação em que o pior caso ocorre.

      (ii) [1 valor] Qual é a complexidade temporal do seu algoritmo, no melhor caso? Justifique e indique uma situação em que o melhor caso ocorre.

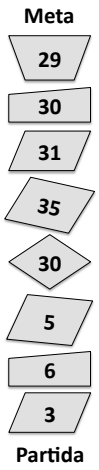
3. [6 valores] Considere a seguinte função recursiva $f(i,j)$, onde i e j são números inteiros não negativos. (A expressão $\lfloor x/y \rfloor$ denota a divisão inteira de x por y.)

$$f(i,j) = \begin{cases} j, & \text{se } i = 0 \text{ e } j \geq 0; \\ i+1, & \text{se } i \geq 1 \text{ e } j = 0; \\ \max\limits_{0 \leq k \leq \lfloor j/2 \rfloor} \Big(f(i,k) \times f(i-1, j-k) \Big), & \text{se } i \geq 1 \text{ e } j \geq 1. \end{cases}$$

Apresente um algoritmo, desenhado segundo a técnica da programação dinâmica, que, dados dois números inteiros positivos, m e n, calcula o valor de $f(m,n)$. Estude (justificando) as complexidades temporal e espacial do seu algoritmo, no pior caso.

4. [4 valores] As crianças jogam ao *SaltaLajes* numa sequência de lajes. Antes de cada jogo, um adulto escreve em cada laje um número com giz (como se ilustra na figura). Depois, cada criança faz uma sequência de saltos sobre as lajes, começando na partida (antes da primeira laje) e terminando na meta (depois da última laje), obtendo uma pontuação. Ganha quem obtiver a maior pontuação.

Para efetuar a sequência de saltos, a criança coloca-se na partida. No primeiro salto, a criança só pode colocar o pé direito numa laje; no segundo salto, só pode colocar o pé esquerdo; e assim sucessivamente, alternando o pé que toca no chão, até chegar à meta. Todos os saltos são na direção da meta. Cada salto pode ter *comprimento* 1, 2 ou 3, ou seja, a criança pode saltar para a primeira, a segunda ou a terceira lajes a seguir ao sítio onde se encontra. A sequência de saltos é pontuada somando os números nas lajes tocadas com o pé direito e subtraindo os números nas lajes tocadas com o pé esquerdo.



Por exemplo, uma criança que salte para as lajes com os números:

- **3** (primeiro salto, de comprimento 1, com o pé direito),
- **5** (segundo salto, de comprimento 2, com o pé esquerdo) e
- **31** (terceiro salto, de comprimento 3, com o pé direito)

e depois salte para a meta (quarto salto, de comprimento 3, com o pé esquerdo) obtém a pontuação $\mathbf{3 - 5 + 31} = 29$.

Outra criança, que efetue os quatro primeiros saltos para as lajes com os números **6**, **5**, **35** e **29** e o quinto salto para a meta, obtém a pontuação $\mathbf{6 - 5 + 35 - 29} = 7$. Qual é a maior pontuação que uma criança pode obter?

Apresente **uma função recursiva** que, com base:

- num inteiro positivo C e
- numa sequência $S = (x_1, x_2, \ldots, x_n)$ de inteiros positivos, com $n > C$,

calcula a maior pontuação que uma criança pode obter a jogar ao *SaltaLajes*, quando o comprimento máximo de cada salto é C e a sequência de lajes tem os números $(x_1, x_2, \ldots, x_n)$. No exemplo, $C = 3$ e $S = (3, 6, 5, 30, 35, 31, 30, 29)$. Indique claramente o que representa cada uma das variáveis que utilizar e explicite a chamada inicial (a chamada que resolve o problema).