1. (2.0pt) Fomos contratados para definir um planejamento de tarefas para uma equipe para as próximas n semanas. Para cada semana existem 3 tarefas possíveis: A , B e C. A realização da tarefa A na semana i gera um lucro de a_i; a realização da tarefa B na semana i gera um lucro de b_i e a realização da tarefa C na semana i gera um lucro de c_i. Sabe-se também que, para $i = 2, ..., n$, a tarefa C só pode ser realizada na semana i se a tarefa B for realizada na semana $i - 1$.

a) Seja $OPT(i)$ o rendimento máximo que pode ser obtido nas i primeiras semanas. Encontre uma equação de recorrência para $OPT(i)$.

b) Escreva um algoritmo polinomial e não recursivo para computar as tarefas que devem ser realizadas a cada semana de modo a maximizar o rendimento total.

Questão 2 (3.0pt) Considere a equação de recorrência que define o número de Lupinutesky: Se $i > 1$ ou $j > 1$ então

$$Lup(i, j) = 3Lup(i - 2, j) + \max_{k=1,...,j-1}\{k + Lup(i, k)\}.$$

Se $i \leq 1$ ou $j \leq 1$, então

$$Lup(i, j) = 1$$

.

a)(2.0pt) Desenvolva o PSEUDO-CÓDIGO de um algoritmo polinomial e recursivo para calcular $Lup(n, n)$.

b)(1.0pt) Analise a complexidade do algoritmo proposto no item (a) em função de n .

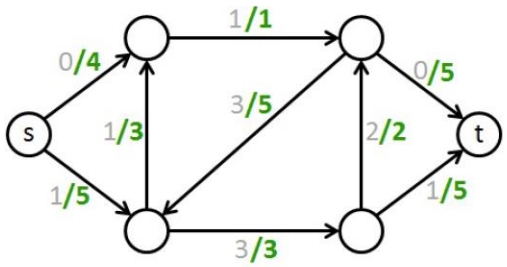
You are going on a long trip. You start on the road at mile post 0. Along the way there are n hotels, at mile posts $a_1 < a_2 < \cdots < a_n$, where each a_i is measured from the starting point. The only places you are allowed to stop are at these hotels, but you can choose which of the hotels you stop at. You must stop at the final hotel (at distance a_n), which is your destination.

You'd ideally like to travel 200 miles a day, but this may not be possible (depending on the spacing of the hotels). If you travel x miles during a day, the *penalty* for that day is $(200 - x)^2$. You want to plan your trip so as to minimize the total penalty—that is, the sum, over all travel days, of the daily penalties.

Give an efficient algorithm that determines the optimal sequence of hotels at which to stop.

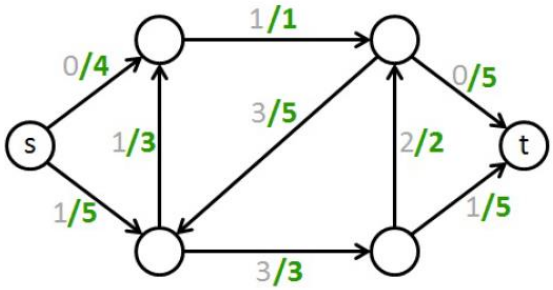
Questao: Considere o fluxo abaixo (capacidades em verde, fluxo em cinza). Decomponha esse fluxo em fluxos em caminhos e fluxos em ciclos

Ou seja, voce deve retornar uma colecao de fluxos f1, f2, ... fk tal que: 1) Para cada aresta *e*, se somarmos of fluxos f1(e)+f2(e)+...+fk(e) obtemos o fluxo na aresta *e* no grafo abaixo; 2) Cada fi passa fluxo somente em **um caminho** ou em **um ciclo**.



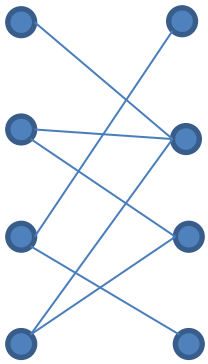
3. Use the Ford-Fulkerson algorithm to find an s,t-flow $0 \leq f \leq c$ of maximum value in the graph below (the values in green are the capacities and the values in grey are the current flow). Use the current flow as starting point. In each iteration, show the residual network and path used for augmentation.

   In addition, argue that the flow you found has maximum value by looking at s,t-cuts in the graph.



Questao: Lembre que um *matching* em um grafo é um subconjunto de arestas que contem no maximo uma aresta terminando em cada vertice.

Considere o grafo abaixo. Prove que ele nao tem um *matching* que case todos os nós. Para isso, reduza o problema de matching máximo ao problema de fluxo máximo e use um certificado do valor do fluxo máximo para provar o resultado desejado.



Given two strings $x = x_1x_2 \cdots x_n$ and $y = y_1y_2 \cdots y_m$, we wish to find the length of their *longest common substring*, that is, the largest k for which there are indices i and j with $x_ix_{i+1} \cdots x_{i+k-1} = y_jy_{j+1} \cdots y_{j+k-1}$. Show how to do this in time $O(mn)$.