Una nuova compagnia di trasporto aereo ha costruito il proprio orario e sta progettando una serie di servizi da fornire ai propri clienti per progettare i loro viaggi. L'orario della compagnia è dato da:

- Una lista A di ***n*** aeroporti in cui la compagnia fa scalo; per ogni aeroporto ***a*** lviene fornita l'indicazione del tempo minimo di coincidenza ***c(a)***, che è il tempo minimo necessario in quell'aeroporto per prendere una coincidenza;
- Un insieme F di ***m*** voli, dove per ogni volo ***f*** viene indicato:
  - l'areoporto di partenza ***s(f)***
  - l'areoporto di arrivo ***d(f)***
  - l'orario di partenza ***l(f)***
  - l'orario di arrivo ***a(f)***
  - il numero di posti disponibili sul volo ***p(f)***

1. Progettare una funzione **list_routes()** che, preso in input l'orario della compagnia, gli areoporti ***a*** e ***b***, un orario di partenza ***t*** ed un intervallo di tempo T, restituisce tutte le rotte che consentono di andare da ***a*** a ***b*** con un durata complessiva del viaggio non superiore a T e con orario di partenza successivo a t. Una rotta è costituita da una sequenza di voli e la sua durata è data dalla somma delle durate di tutti i voli più i tempi di attesa negli scali per le coincidenze. Ad ogni scalo bisogna considerare che non è possibile effettuare una coincidenza se tra l'orario di atterraggio di un volo ed il tempo di decollo del volo successivo intercorre un tempo inferiore a ***c(a).***

2. Progettare una funzione **find_route()** che, preso in input l'orario della compagnia, gli areoporti ***a*** e ***b***, ed un orario di partenza ***t***, trova la rotta che permette di arrivare da ***a*** a ***b*** nel minor tempo possibile, partendo ad un orario non precedente a t. (Come per l'esercizio precedente, bisogna tener conto del tempo minimo di coincidenza di ogni scalo).

3. Un volo consuma 60kg di gasolio per ogni ora di volo e prima del decollo la compagnia deve acquistare dal gesture dell'areoporto il gasolio necessario per il volo. Si assuma che ogni kg di gasolio costa 1€ e che la compagnia ha a disposizione un budget complessivo uguale a B per pagare il gasolio e che questo budget non consente di coprire tutti i voli previsti nell'orario. Gli amministratori della compagnia devono decidere quali voli far partire e quali cancellare. Progettare una funzione **select_flights()** che, preso in input l'orario della compagnia ed il budget B, seleziona quali voli far decollare in modo da massimizzare il numero complessivo di posti disponibili. Inoltre, la funzione deve restituire per ogni areoporto ***a*** quanti soldi devono essere assegnati al responsabile dello scalo per pagare il gasolio necessario per tutti i voli in partenza da ***a***.

   Analizzare la complessità di tempo di ciascuna delle tre funzioni proposte in funzione di ***n*** ed ***m***.

4. Scrivere una funzione **bipartite()** che, preso in input un grafo G non diretto, verifica se G è bipartito e restituisce una partizione (X, Y) dei vertici di G tale che tutti gli archi del grafo collegano un vertice di X ad un vertice di Y. Nel caso in cui il grafo non sia bipartito la funzione deve restituire None. Analizzare la complessità della funzione proposta.

La soluzione deve essere caricata sul sito del corso **entro le ore 23.59 di domenica 30 dicembre**.
Il materiale consegnato deve essere costituito da un unico file compresso contenente:

- Un file pdf con la documentazione del progetto
- Un progetto Python chiamato #gruppo_2_TdP contenente
  - il package Python TdP_collections (con tutte le implementazioni delle strutture dati di supporto, eventualmente modificate);
  - il package Python pkg_i contenente le classi e le funzioni per la soluzione dell'esercizio i (i = 1, 2, 3, 4);
  - script di testing per ciascun esercizio.

Per ogni gruppo deve essere caricato un solo file.