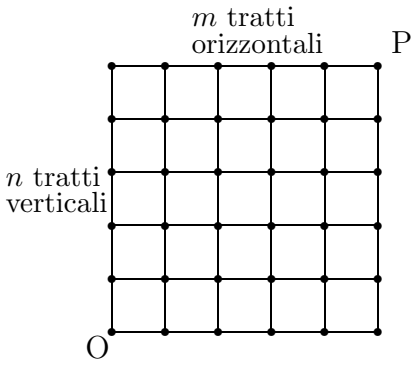
Indice

1 Scheda n. 1s: il problema “Manhattan”

Considera una mappa stradale come quella rappresentata nella figura sottostante, che ricorda le strade del quartiere di New York “Manhattan”.



I punti rappresentano gli incroci, mentre i tratti orizzontali e verticali, tutti della stessa lunghezza L, rappresentano le strade che uniscono questi incroci. Osserva che gli incroci O e P riportati nella figura, rappresentano i due vertici opposti di un rettangolo di base mL ed altezza nL. Vogliamo conoscere:

1. qual’è, se esiste, la minima lunghezza di un percorso che, rimanendo all’interno del rettangolo, parta dal punto P e giunga al punto O;
2. se tale minima lunghezza esiste, quanti sono i possibili percorsi distinti di minima lunghezza.

Esercizi da svogersi a casa

- Riflettere sulla questione dell’iniettività e suriettività della funzione $P \to \mathrm{d}_{PO}$.
- Analizzando dei semplici casi, ad esempio $P(1,2)$, $P(2,2)$ e $P(2,3)$,
  - cosa si può dire sull’unicità dei cammini minimi?
  - provare a dare una qualche descrizione (caratterizzazione) dei cammini minimi;
  - secondo la vostra intuizione[1], dato il punto di partenza $P(m,n)$, quanto vale d_{PO}?

[1] Si considerino i valori d_{PO} per i punti P più vicini ad O.

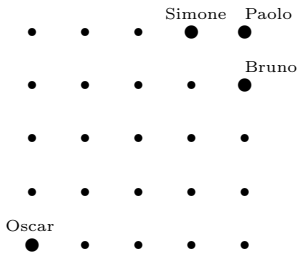
2 Scheda n. 2s: simulazione dell'algoritmo ricorsivo

Lo scopo del gioco-simulazione che qui viene descritto è mostrare che applicando le regole dell'algoritmo ricorsivo è possibile in un tempo finito trovare il numero di percorsi minimi tra due incroci di una piccola rete stradale.

Materiali necessari

- almeno 70 biglie raggruppate in tre o quattro cestelli;
- un cartellone e materiale di cancelleria;
- 25 blocknotes, altrettante penne e ciotole capienti (una ciotola deve poter contenere tutte le settanta bilgie);
- un orologio per misurare la durata del gioco.

Preparazione del gioco
In un spazio liberato dai banchi, si dispongono 25 sedie a formare una griglia di 5 righe e 5 colonne, e ciascun discente occupa un posto a sedere. Ogni discente deve avere penna e blocknotes a portata di mano. Sulla lavagna si rappresenta uno schema con nome e posizione di ciascun discente, analogo a quello riportato nella seguente figura.



Il discente che occupa il posto nell'angolo in alto a destra (nell'esempio è Paolo) rappresenta l'incrocio di partenza dei percorsi, mentre quello seduto in basso e a sinistra (nell'esempio è Oscar), ne rappresenta la meta. Tutti gli altri discenti rappresentano gli altri 23 incroci del rettangolo di rete stradale così individuato. I cestini di biglie devono essere disposti vicino alla sedie della riga di estrema sinistra e della colonna più in basso.

Lo scopo del gioco è trovare il numero di prcorsi minimi tra l'incrocio di partenza e quello di arrivo, utilizzando le regole dell'algoritmo ricorsivo con cui è stato risolto il problema "Manhattan".

Le regole del gioco

1. gli incroci si dividono in incroci base, cioè gli incroci nella riga e nella colonna di Oscar, e incroci ricorsivi, che sono tutti gli altri casi;

2. ogni rappresentante di incrocio ricorsivo può richiedere all'incrocio adiacente nella colonna a sinistra e a quello adiacente nella riga in basso, che consegnino un numero di biglie pari al rispettivo numero percorsi minimi che li congiungono alla meta. Una volta che queste due richieste sono state soddisfatte, la somma dei due gruppi di biglie, in virtù della regola ricorsiva, equivale al numero di percorsi minimi che congiungono l'incrocio considerato con la meta;

3. ogni rappresentante di incrocio base, invece ad ogni richiesta pervenuta dall'incrocio adiacente nella colonna a destra, o da quello nella righa in alto, risponde prendendo una biglia dal cestino più vicino e consegnandola all'incrocio richiedente;

4. inizialmente nessuno dei 25 rappresentanti degli incroci possiede alcuna bilgia;

5. il gioco inizia da Paolo che, in conformità con la regola 3, richiede le biglie a Simone e Bruno. Questi ultimi, per soddisfare la richiesta, dovranno instradare ulteriori richieste, e così via fino a giungere gli incroci base che, finalmente, risponderanno ad ogni richiesta consegnando una biglia;

6. ogni rappresentante di incrocio annoterà nel proprio block notes il numero di richieste che ha dovuto soddisfare per ciascun rappresentante richiedente, e il numero totale di biglie che ha consegnato ad ogni richiesta;

7. finché un incrocio non ha soddisfatto una richiesta, cioè finché non ha consegnato le biglie al richiedente, non potrà occuparsi di altre nuove richieste, che però accoderà in una lista di attesa nel blocknotes;

8. il gioco termina quando le due richieste del rappresentante dell'incrocio di partenza vengono soddisfatte.

3 Scheda n. 3s: attività in laboratorio d'informatica

Gruppo: Data:

Nel seguente listato sono definite nel linguaggio Pascal due funzioni, che, analogamente all'algoritmo del problema "Manhattan", hanno una struttura ricorsiva, cioè richiamano loro stesse nel corpo della definizione.

La prima funzione, **fattoriale**(n), restituisce il prodotto dei primi n interi se $n \geq 1$ mentre, per definizione, **fattoriale**$(0) = 1$ [2]

```
function fattoriale(n: integer): integer;
begin
   if (n = 0) or (n = 1) then fattoriale:=1
   else fattoriale:= n*fattoriale(n − 1)
end;
```

La seconda funzione, **pot**(a, n), è definita per ogni a reale e per ogni intero $n \geq 0$ da

```
function pot(a: real; n: integer): real;
begin
   if n = 0 then pot:=1
   else pot:= a*pot(a, n − 1)
end;
```

Consegne

1. Dopo aver specificato per entrambe le funzioni le linee di codice in cui vengono trattati casi base e casi risorsivi , si descriva il significato matematico della funzione **pot**. Successivamente si scriva un programma che faccia uso della funzione **pot**, per verificarne la buona definizione.

[2] Il fattoriale di n, si indica in genere con la scrittura $n!$.

2. La funzione Manhattan $\mathbf{M}$ è definita per ricorsione da

$$\begin{cases} \forall m, n \in \mathbb{N} \ : \ \mathbf{M}(m, 0) = \mathbf{M}(0, n) = 1 \\ \forall m, n \in \mathbb{N}_0 : \ \mathbf{M}(m, n) = \mathbf{M}(m-1, n) + \mathbf{M}(m, n-1) \end{cases}$$

Si completi il seguente programma di modo che $\mathbf{M}$ implementi la funzione Manhattan.

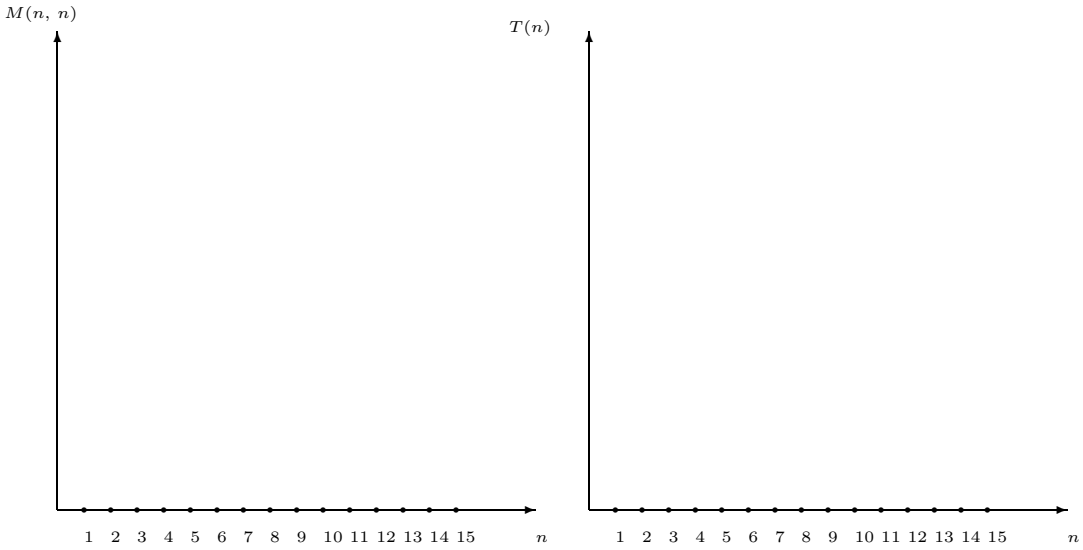
```
program manhattan;
var m,n: integer;
function M(                          ): longint;
begin




end;
begin
   writeln('Inserisci le coordiante ( m, n) del punto P ');
   write('m= (≥ 0)');
   readln(m);
   write('n= (≥ 0)');
   readln(n);
   writeln('Vi sono ', M( m, n), ' percorsi minimi da P ad O');
   readln
end.
```

3. Si verifichi speriementalmente il corretto funzionamento del programma per confronto con i risultati ottenuti negli esempi proposti in classe e nel gioco-simulazione (ad es. $\mathbf{M}(4, 3) = 35$).

4. Si raccolgano i valori $\mathbf{M}(n, n)$ e i relativi tempi di esecuzione $\mathbf{T}(n)$ del programma, misurati con un cronometro da polso, per i valori $n = 1, \ldots, 15$. Si traccino due grafici che riportino l'andamento di $\mathbf{M}(n, n)$ e $\mathbf{T}(n)$ al variare di n, provando a fornire una spiegazione della crescita vertiginosa dei tempi di computazione dell'algoritmo.



Esercizi per casa [lavoro individuale]

- estendere la definizione della funzione **pot** al caso di esponenti negativi;
- realizzare una funzione ricorsiva che realizzi $n \to (n!)!$;
- realizzare una funzione che realizzi $n \to 1 + \frac{1}{2!} + \cdots + \frac{1}{n!}$