**Esercizio 1**

Si consideri il seguente programma Python:

def f(n):

if n == 0:

return 0

if n < 0:

n = - n

return n + g(n - 1)

def g(n):

return f(- n)

print(f(-4))

Si descriva sinteticamente la funzione svolta dal programma e si mostrino l'output prodotto e la traccia d'esecuzione.

**Esercizio 2**

Si scriva una funzione *estrai\_lista* che riceve in ingresso due matrici *M* ed *N* e restituisce una lista contenente gli elementi di *M* in posizione *(i,j)*  tale che *(i+j)* è pari e che occorrano almeno una volta in *N* in una qualsiasi posizione *(i’,j’)* tale che *(i’+j’)* è dispari.

|  |  |  |
| --- | --- | --- |
| 8 | 1 | 7 |
| 14 | 5 | 2 |
| 4 | 8 | 3 |

|  |  |  |
| --- | --- | --- |
| 2 | 7 | 5 |
| 5 | 10 | 1 |
| 8 | 3 | 14 |

*Esempio:* Se *M* = ed *N* = allora la funzione restituisce la lista [2, 8, 14] perché:

* 2 si trova in posizione (0,0) in *M*, 0+0 è pari, 2 occorre in posizione (1,2) in *N* e 1+2 è dispari.
* 7 si trova in posizione (0,1) in *M* e 0+1 è dispari.
* 5 si trova in posizione (0,2) in *M*, 0+2 è pari, 5 occorre solo in posizione (1,1) in *N* e 1+1 è pari.
* 5 si trova in posizione (1,0) in *M* e 1+0 è dispari.
* 10 si trova in posizione (1,1) in *M*, 1+1 è pari, ma 10 non occorre in *N*.
* 1 si trova in posizione (1,2) in *M* e 1+2 è dispari.
* 8 si trova in posizione (2,0) in *M*, 2+0 è pari, 8 occorre in posizione (2,1) in *N* e 2+1 è dispari.
* 3 si trova in posizione (2,1) in *M* e 2+1 è dispari.
* 14 si trova in posizione (2,2) in *M*, 2+2 è pari, 14 occorre in posizione (1,0) in *N* e 1+0 è dispari.

**Esercizio 3**

Si vuole realizzare un'applicazione di supporto ad un giocatore di *Othello*. Il gioco si svolge su una scacchiera 8x8 su cui vengono disposte pedine. Ogni pedina ha una faccia bianca e una nera. La scacchiera è rappresentata mediante una matrice *S* di caratteri, in cui il carattere 'B' rappresenta una pedina bianca, il carattere 'N' rappresenta una pedina nera ed il carattere 'V' indica una casella vuota. Ad ognuno dei due giocatori viene assegnato un colore (bianco o nero) e 32 pedine. La mossa di un giocatore consiste nel posizionare, in una casella vuota, una pedina del colore assegnato al giocatore stesso in modo che avvenga la *cattura* di almeno una pedina del colore assegnato al giocatore avversario. Una pedina viene catturata se si trova, in orizzontale, verticale o diagonale, tra due pedine del colore opposto alla pedina stessa (si veda la Figura 1). A seguito di una mossa, tutte le pedine catturate cambiano colore. Una mossa non è valida se nessuna pedina viene catturata. La partita termina quando nessuno dei due giocatori può posizionare ulteriori pedine sulla scacchiera, nel qual caso la vittoria va al giocatore che ha il maggior numero di pedine del proprio colore sulla scacchiera.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| V | V | V | V | V | V | V | V |
| V | V | V | V | V | V | V | V |
| V | V | V | V | V | V | V | V |
| V | V | V | V | V | V | V | V |
| V | V | V | **B** | **B** | **B** | V | V |
| V | V | **B** | **B** | **B** | **B** | **B** | V |
| V | **N** | **N** | **N** | **N** | **N** | **N** | **N** |
| V | V | V | V | V | V | V | V |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| V | V | V | V | V | V | V | V |
| V | V | V | V | V | V | V | V |
| V | V | V | V | V | V | V | V |
| V | V | V | V | **N** | V | V | V |
| V | V | V | ***N*** | ***N*** | ***N*** | V | V |
| V | V | ***N*** | **B** | ***N*** | **B** | ***N*** | V |
| V | **N** | **N** | **N** | **N** | **N** | **N** | **N** |
| V | V | V | V | V | V | V | V |

(b)

(a)

**Figura 1:** Configurazione della scacchiera prima (a) e dopo (b) il posizionamento di una pedina nera nella cella (3,4). Le pedine catturate sono riportate in corsivo su sfondo grigio.

Si scrivano in Python le seguenti funzioni (la scrittura di eventuali funzioni di supporto è ovviamente ammessa):

* *conta\_catture(S, c, i, j)* che riceve *S*, un colore *c* e due indici *i* e *j*. La funzione restituisce il numero di pedine avversarie catturate se il giocatore a cui è assegnato il colore *c* posiziona una pedina nella casella *(i,j).* Se la casella *(i,j)* è già occupata, la funzione restituisce 0.
* *vittoria(S)* che riceve la scacchiera *S* e restituisce il colore ('B' o 'N') del vincitore, se la partita è terminata; altrimenti, restituisce 'Non terminata'.
* *esiste\_mossa\_valida(S, c, k)* che riceve la scacchiera *S* e un colore *c* e restituisce *True* se e solo se sulla scacchiera esiste almeno una casella che consenta la cattura di esattamente *k* pedine avversarie da parte del giocatore a cui è assegnato il colore *c*.
* *mossa(S, c, i, j)* che riceve la scacchiera *S*, un colore *c* e due indici *i* e *j.* La funzione:
  + aggiunge alla scacchiera una pedina di colore c nella casella *(i,j)*, cambia il colore delle pedine catturate e restituisce *True*, se il posizionamento della nuova pedina rappresenta una mossa valida;
  + restituisce *False*, altrimenti.

*Esempio:* Si supponga che la scacchiera *S* sia quella riportata in **Figura 1-a**, allora:

* *conta\_catture (S,‘N’,3,4)* restituisce il valore 6 perché a seguito del posizionamento della pedina nera in (3,4) vengono catturate 6 pedine bianche, quelle nelle caselle (4,3), (4,4), (4,5), (5,2), (5,4) e (5,6).
* *vittoria(S) retituisce ‘Non terminata’* perché sono ancora presenti mosse disponibili.
* *esiste\_mossa\_valida(S,‘N’,2)* restituisce *True* perché se si posiziona una pedina nera in (4,2) è possibile catturare esattamente 2 pedine avversarie.
* *mossa(S,‘N’,3,4*) modifica la matrice *S* ottenendo come risultato la matrice mostrata in Figura 1-b e restituisce *True*.