Esercizi di Fondamenti di Informatica

Andrea Gussoni
andrea1.gussoni at polimi.it

Politecnico di Milano

December 5, 2019

Table of Contents

Section 1

Ripasso Esercitazione Precedente

Ripasso

Problemi con esercizi della precedente esercitazione?

- Esercizio di ordinamento
- Esercizi insiemi/dizionari

Game Of Life

Ideare un programma che implementi il Conway's Game Of Life:

- Il gioco è un gioco a zero giocatori, che si svolge su una griglia bidimensionale (una matrice nel nostro caso), composta da celle.
- Ogni cella della griglia, può essere in due stati, viva o morta.
- Ogni cella ha 8 celle vicine (anche in diagonale).
- Ad ogni iterazione del gioco, lo stato successivo di ogni punto della griglia è calcolato in base allo stato delle celle vicine all'iterazione precedente:
  1. Una cella con meno di due vicini vivi muore per isolamento.
  2. Una cella con due o tre celle vicine sopravvive.
  3. Una cella con più di 3 celle vicine vive muore per sovrappopolazione.
  4. Una cella morta con 3 celle vive vicine diventa viva.

Game Of Life I

Una possibile soluzione

```
from random import randint

def matrice_iniziale_random(n):
    matrice = matrice_vuota(n)
    for i in range(n):
        for j in range(n):
            matrice[i][j] = randint(0,1)
    return matrice

def matrice_iniziale(n):
    matrice = matrice_vuota(n)

    matrice[1][3] = 1
    matrice[1][4] = 1
```

Game Of Life II

```
    matrice[2][4] = 1

    return matrice

def matrice_vuota(n):
    matrice = []
    for i in range(n):
        riga = []
        for j in range(n):
            riga.append(0)
        matrice.append(riga)
    return matrice

def stampa_matrice(matrice):
    dim = len(matrice)
    for i in range(dim):
```

Game Of Life III

```
            for j in range(dim):
                if matrice[i][j] == 1:
                    print("*", end="")
                else:
                    print(".", end="")
            print("")

    def calcola_nuova_generazione(matrice):
        dim = len(matrice)
        nuova = matrice_vuota(dim)

        for i in range(0, dim):
            for j in range(0, dim):
                vicini_vivi = 0

                for x in [-1, 0, 1]:
```

Game Of Life IV

```
            for y in [-1, 0, 1]:
                if i + x >= 0 and j + y >= 0
                    and i + x < dim and j + y < dim:
                     vicini_vivi += matrice[i + x][j + y]

        vicini_vivi -= matrice[i][j]
        # Cellula sola e muore
        if matrice[i][j] == 1 and vicini_vivi < 2:
            nuova[i][j] = 0

        # Cellula muore per sovvrapopolamento
        elif matrice[i][j] == 1 and vicini_vivi > 3:
            nuova[i][j] = 0

        # Nasce una nuova cellula
        elif matrice[i][j] == 0 and vicini_vivi == 3:
```

Game Of Life V

```
                nuova[i][j] = 1

            # Rimane uguale
            else:
                nuova[i][j] = matrice[i][j]

    return nuova

def main():
    matrice = matrice_iniziale(10)
    stampa_matrice(matrice)
    for i in range(0,3):
        print("Generazione:", i)
        matrice = calcola_nuova_generazione(matrice)
        stampa_matrice(matrice)
```

Game Of Life VI

```
main()
```