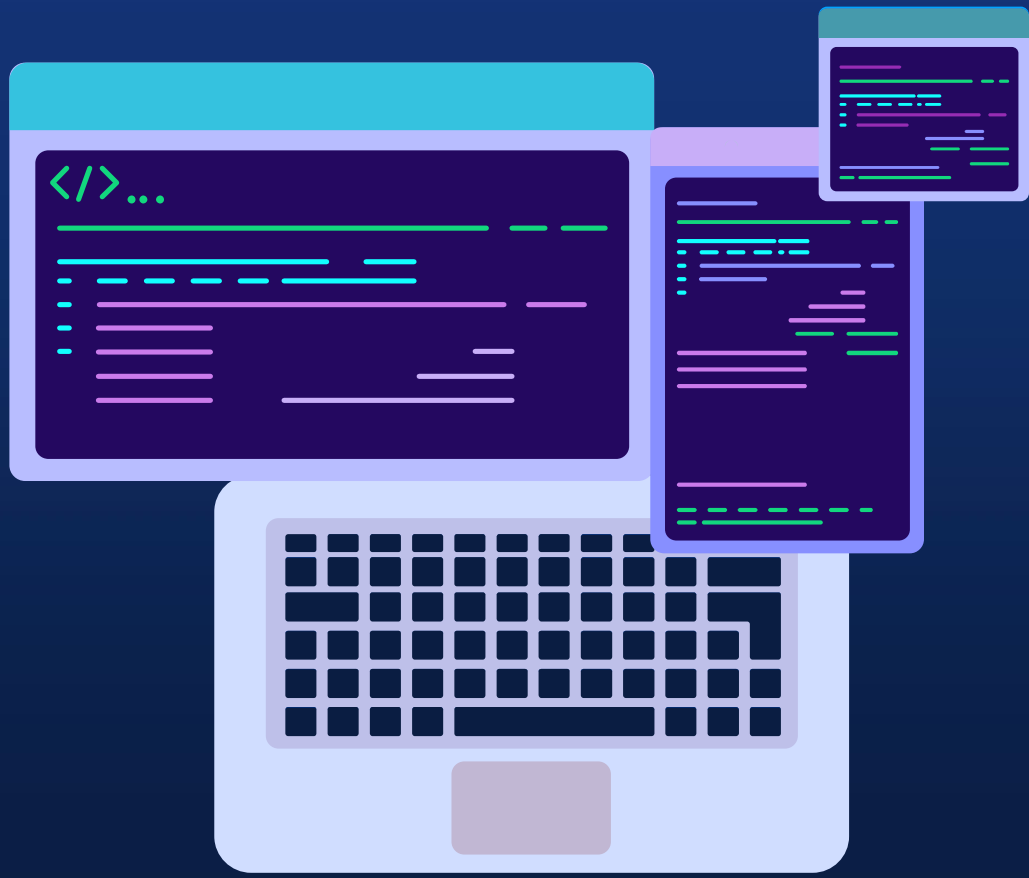
PROGETTO MARATONA

Applicazione dei thread in python



Presentazione di: Bastianelli Lorenzo, Fiore Enrico

01

Introduzione Progetto

Introduzione al progetto

Fare una demo per un gioco che simulasse la maratona annuale di New York.

Obiettivi progetto

1. Simulare una maratona di 42 km.
2. Usare la libreria threading per far gareggiare parallelamente i giocatori.
3. Gestire eventi casuali che possono avvenire durante la gara.

EVENTI CASUALI

Scatto

Il tempo viene ridotto del 30% ogni km per 2 secondi poi ritorna normale

Contrattura

il tempo a Km viene raddoppiato fino a prossimo evento

Andatura normale

Il tempo a km rimane invariato

Stiramento

Il tempo viene quadrupicato fino alla fine

Ritmo in aumento

Tempo ridotto del 10%

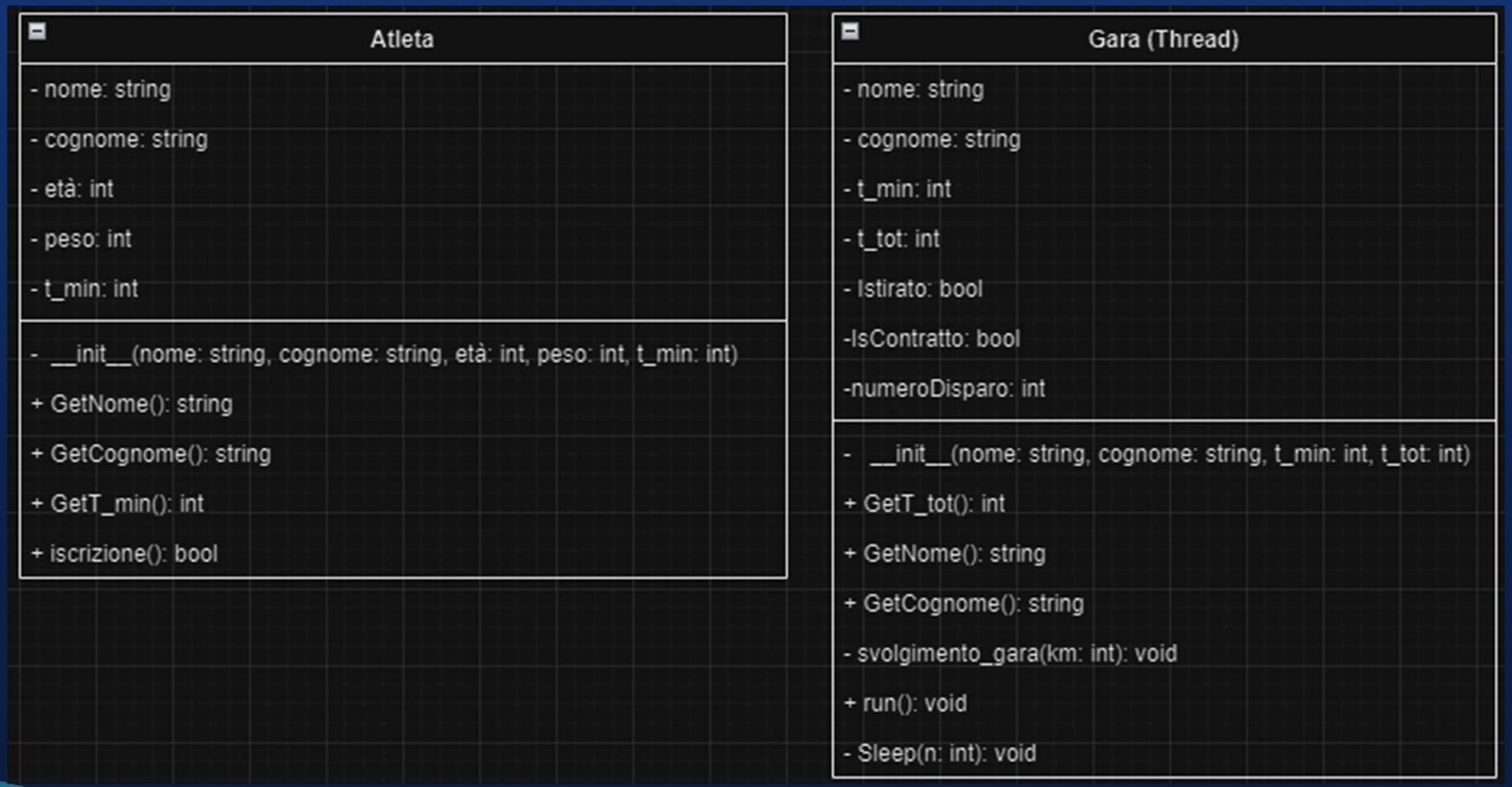
Stanchezza

Tempo aumentato del 10%

02

Schema UML

Schema UML


Atleta
- nome: string
- cognome: string
- età: int
- peso: int
- t_min: int
- __init__(nome: string, cognome: string, età: int, peso: int, t_min: int)
+ GetNome(): string
+ GetCognome(): string
+ GetT_min(): int
+ iscrizione(): bool
Gara (Thread)
- nome: string
- cognome: string
- t_min: int
- t_tot: int
- Istirato: bool
-IsContratto: bool
-numeroDisparo: int
- __init__(nome: string, cognome: string, t_min: int, t_tot: int)
+ GetT_tot(): int
+ GetNome(): string
+ GetCognome(): string
- svolgimento_gara(km: int): void
+ run(): void
- Sleep(n: int): void

03

Spiegazione Codice

LIBRERIE UTILIZZATE

```
1    from threading import Thread
2    import time
3    import random
```

CLASSE ATLETA

```
5   #Classe statistiche atleta
6   class Atleta:
7       def __init__(self, nome, cognome, età, peso, t_min):
8           self.nome = nome
9           self.cognome = cognome
10          self.età = età
11          self.peso = peso
12          self.t_min = t_min
13
14      def GetNome(self):
15          return self.nome
16      def GetCognome(self):
17          return self.cognome
18      def GetT_min(self):
19          return self.t_min
20
21      def iscrizione(self) :
22          if(self.t_min <= 5) :
23              return True
24          else :
25              print(self.nome + " " + self.cognome + " non puoi parteciapare perchè il tempo è maggiore di 5, tempo minimo atleta: " + str(self.t_min) + "\n")
26              return False
```

CLASSE GARA (parte 1)

```
30  #Classe con Thread
31  class Gara(Thread):
32
33      def __init__(self,nome,cognome,t_min,t_tot):
34          Thread.__init__(self)#creo Thread
35          self.nome = nome
36          self.t_min = t_min
37          self.t_tot = t_tot
38          self.cognome = cognome
39
40      # variabili generali per salvare il dato del singolo maratoneta
41      numeroDisparo = 0
42      Istirato = False
43      IsContratto = False
44
45      def Sleep(n):
46          time.sleep(n)
47
48      def GetT_tot(self):
49          return self.t_tot
50      def GetNome(self):
51          return self.nome
52      def GetCognome(self):
53          return self.cognome
```

```
55      def svolgimento_gara(self,km):
56          # il segno // divide per interi invece / è la divisione normale con la virgola
57          if(self.Istirato==False):
58              if(self.IsContratto == False):
59                  if(km <= 1):# se è il primo km o meno lo indirizzo direttamente ad una andatura normale
60                      n_random = 3
61                  elif(km % 2 == 0):# ogni 2 km estraggo un numero casuale per l'evento
62                      n_random = random.randint(1,10)#numero casuale da 1 a 10
63                      self.numeroDisparo  = n_random
64                  elif(km % 2 != 0 and km >2):# l'evento successo nel km precedente (pari) continua anche per il km successivo (disparo)
65                      n_random = self.numeroDisparo
66                  if(n_random == 1 and self.Istirato == False): #scatto
67                      print(self.nome + " " + self.cognome + " ha fatto uno scatto\n")
68                      self.t_tot += self.t_min // 0.7
69                      Gara.Sleep(2)
70                      self.t_tot += self.t_min
71                  if(n_random == 2 and self.Istirato == False): #contrattura
72                      print(self.nome + " " + self.cognome + " ha ricevuto una cotrattura\n")
73                      self.t_tot += self.t_min * 2
74                      self.IsContratto = True
75                  if(n_random>=3 and n_random<=7 and self.Istirato == False):#andatura normale
76                      print(self.nome + " " + self.cognome + " corre spensierato\n")
77                      self.t_tot += self.t_min
78                  if(n_random == 8): # stiramento
79                      print(self.nome + " " + self.cognome + " ha ricevuto stiramento\n")
80                      self.t_tot += self.t_min * 4
81                      self.Istirato = True
82                  if(n_random == 9 and self.Istirato == False):#ritmo in aumento
83                      print(self.nome + " " + self.cognome + " ha aumentato il ritmo\n")
84                      self.t_tot += self.t_min // 0.9
85                  if(n_random == 10 and self.Istirato == False):#stanchezza
86                      print(self.nome + " " + self.cognome + " inizia a sentire la stanchezza!\n")
87                      self.t_tot += self.t_min * 1.1
88              else:
89                  if(random.randint(1,2)==2): #non ha più la contrattura
90                          self.t_tot -= self.t_min
91                          print(self.nome + " " + self.cognome + " non ha più la contrattura\n")
92                          self.IsContratto = False
93          else:
94               print(self.nome + " " + self.cognome + " è ancora stirato\n")
95               self.t_tot += self.t_min * 4
96          Gara.Sleep(1)#da il tempo di un 1 secondo a km
```

CLASSE GARA (parte 3)

```
100     def run(self):
101         print("è partito: " + self.nome + " " +self.cognome )
102         for km in range(42):#simulazione di una maratona, ogni km succede qualcosa
103             Gara.svolgimento_gara(self,km+1)
104             print(self.nome + " " + self.cognome +" al chilometro: " + str(km+1) + " di corsa\n")
105             #print(self.nome + " " + self.cognome +" tempo: " + str(self.t_tot))
```

«Funzioni Esterne»(parte 1)

```
123   def mostraAtleti():
124       j=1
125       for i  in Atleti:
126           print("Partecipante "+ str(j) + ": " + i.GetNome() + " " + i.GetCognome()+ "\n")
127           j+=1
128
```

«Funzioni Esterne» (parte 2)

```python
#MENU
def Menu():
    nelMenu = True #per entrare e rimanere nel menu
    while(nelMenu == True):
        print("Scegliere una tra le seguenti opzioni: \n1) Inserire atleta\n2) Eliminare atleta\n3) Iniziare gara\n4) Mostra partecipanti\n")
        while(True):
            try:
                scelta = int(input("Inserire il NUMERO dell'opzione: "))
                if scelta in (1, 2, 3,4):# se la scelta è 1 o 2 o 3 o 4 esci dal try catch
                    break
                else:
                    print("ATTENZIONE: bisogna inserire il numero corrispondente alla scelta")
            except ValueError:
                print("ATTENZIONE: bisogna inserire il numero corrispondente alla scelta")

        if( scelta == 1): #inserire atleta
            print("Per ISCRIVERE un atleta ci serve: NOME, COGNOME, ETA', PESO, TEMPO MINIMO A KM\n")
            print("Inserire NOME\n")
            nome = str(input("nome: "))
            print("\nInserire COGNOME\n")
            cognome = str(input("cognome: "))
            print("\nInserire ETA'\n")
            età = int(input("età: "))
            print("\nInserire PESO\n")
            peso = int(input("peso: "))
            print("\nInserire TEMPO MINIMO A KM\n")
            t_min = int(input("tempo minimo: "))
            nuovo_atleta = Atleta(nome,cognome,età,peso,t_min)#creazione nella classe Atleta
            Atleti.append(nuovo_atleta)# aggiunto nell'array
        elif(scelta == 2):# elimina atleta
            mostraAtleti()
            print("Per ELIMINARE un atleta ci serve: NOME, COGNOME\n")
            print("Inserire NOME\n")
            nome = str(input("nome: "))
            print("\nInserire COGNOME\n")
            cognome = str(input("cognome: "))
            nonTrovato =0
            for atleta in Atleti:
                if(atleta.GetNome() == nome and atleta.GetCognome() == cognome):
                    Atleti.remove(atleta)# rimosso atleta dall'array
                    print("\nL'Atleta"+ nome +" "+ cognome+" è stato rimosso con successo\n")
                    break
                else:
                    nonTrovato+=1 # per tenere traccia di quante volte il nome inserito non è stato trovato
            if(nonTrovato == len(Atleti)):# in caso il nome è stato inserito sbagliato
                print("\nAtleta non trovo, provare a reinserire atleta\n")
        elif(scelta == 3):# inizia gara
            #controllo di chi può partecipare
            for i in Atleti :
                if i.iscrizione():
                    partecipanti.append(i)
            if(len(partecipanti)<2): #controllo che ci siano abbastanza atleti per fare la gara (minimo 2)
                print("Ci sono ancora pochi alteti per iniziare la gara\n")
            else:
                nelMenu = False
        elif(scelta == 4):# mostra partecipanti
            mostraAtleti()
```

«MAIN»

```
188  #MAIN
189  #t_min = tempo di corsa per gareggiare 5 per Kilometro
190
191  Atleti = [] #array atleti
192  partecipanti = [] # partecipanti gara
193  Menu()
194  #array in cui salvare risultati
195  risultati = []
196  #Partenza gara
197  for i in partecipanti:
198      corridore = Gara(i.GetNome(),i.GetCognome(),i.t_min,0)
199      corridore.start()
200      risultati.append(corridore)
201  #Aspettando che tutti finiscano la gara
202  for thread  in risultati:
203      thread.join()
204  print("-----------------------------GARA FINITA-----------------------------\n")
205  #stampa risultati
206  tempoVincente = 999#numero default
207  nomeVincitore= ""
208  cognomeVincitore =""
209  for corridori in risultati:
210      print(corridori.GetNome() + " " + str(corridori.GetT_tot()))#controllo tempo minore
211      if(corridori.GetT_tot()< tempoVincente):
212       tempoVincente = corridori.GetT_tot()
213       nomeVincitore = corridori.GetNome()
214       cognomeVincitore = corridori.GetCognome()
215
216
217  #stampa vincitore
218  print("Vincitore maratona: " + nomeVincitore + " "+ cognomeVincitore +" con un tempo di: " + str(tempoVincente))
219
220  #ATTENZIONE!!!
221  #prendere il risultato in minuti [non in secondi]
```

04

Difficoltà Riscontrate

DIFFICOLTA' RISCONTRATE

PROBLEMI

- Mostrare i risultati quando tutti i parteciapanti hanno finito la gara
- Gestire le classi
- Implementare i Thread