Tartalomjegyzék

Linkek

- Google dokumentum https://docs.google.com/document/d/1k8hpm1o9429d-Gs_OFAJMtkUWACGdYxvr0baZCnq6b8/edit

Feladatok

El?z? órai feladat folytatása

- Az eddigi kód:

```
class Player:
  def __init__(self,name,height,power,speed,acceleration):
    self.name = name
    self.height = height
    self.power = power
    self.speed = speed
    self.acceleration = acceleration
  def display(self):
    print self.name, self.height, self.power, self.speed, self.acceleration


player1 = Player("AB", 10, 14, 100, 10)
print "Player1:"
player1.display()

class Team:
  def __init__(self,listOfPlayers):
    self.listOfPlayers = listOfPlayers
  def addPlayer(self,player):
    self.listOfPlayers.append(player)
  def displayPlayers(self):
    for player in self.listOfPlayers:
      player.display()

player2 = Player("BC", 10, 15, 10, 3)
player3 = Player("CD", 9, 14, 100, 2)
player4 = Player("EF", 8, 20, 800, 1)
team1 = Team([player1,player2,player3,player4])
print "Team1:"
team1.displayPlayers()

class PlayerRank:
  def __init__(self,team):
    self.setTeam(team)
  def setTeam(self,team):
```

```
    self.team = team
  def getBestPlayer(self):
    bestPlayer = self.team.listOfPlayers[0]
    for player in self.team.listOfPlayers:
      betterPlayer = self.getBetterPlayer(player,bestPlayer)
      if(betterPlayer!=bestPlayer):
        bestPlayer = betterPlayer
    return bestPlayer
  def getBetterPlayer(self,player1,player2):
    pass

class PlayerRankSpeed(PlayerRank):
  def getBetterPlayer(self,player1,player2):
    if player1.speed > player2.speed:
      return player1
    else:
      return player2

class PlayerRankAcceleration(PlayerRank):
  def getBetterPlayer(self,player1,player2):
    if player1.acceleration > player2.acceleration:
      return player1
    else:
      return player2
# ez itt nem a legszebb
class TeamCompare:
  def __init__(self,playerRank,team1,team2):
    self.playerRank = playerRank
    self.team1 = team1
    self.team2 = team2
  def compareTeams(self):
    playerRank.setTeam(team1)
    bestPlayer1 = playerRank.getBestPlayer()
    playerRank.setTeam(team2)
    bestPlayer2 = playerRank.getBestPlayer()
    bestPlayers = Team([bestPlayer1, bestPlayer2])
    playerRank.setTeam(bestPlayers)
    finalBestPlayer = playerRank.getBestPlayer()
    if finalBestPlayer == bestPlayer1:
      print "team1 is better"
    else:
      print "team2 is better"

playerRank = PlayerRankSpeed(team1)
player = playerRank.getBestPlayer()
player.display()

player5 = Player("GH", 10, 15, 10, 3)
player6 = Player("IJ", 9, 14, 100, 2)
player7 = Player("KL", 8, 20, 700, 1)
team2 = Team([player5,player6,player7])

compare = TeamCompare(playerRank,team1,team2)
compare.compareTeams()
```

- Módosítsuk a kódot úgy, hogy egy külön osztályt írunk, amely képes 2 játékost összehasonlítani (PlayerCompare).
- A különböz? összehasonlító stratégiákat ebb?l az osztályból származtassuk le.
- A PlayerRank osztály tartalmazza a PlayerCompare egy példányát.

Modulok

- Próbáljuk ki az el?adás anyagának "http://wiki.math.bme.humodulok"http://wiki.math.bme.hu részét
- Szervezzük külön file-okba az 1. feladat (Player, Team ...) osztályait

Vektortér

- Írjunk osztályt, melyben k dimenziós vektorok vektorterét reprezentálja
- Valósítsuk meg a vektorok összeadását
- Skalárral való szorzását
- Definiáljunk két vektor skaláris szorzatát.

Eloszlásfüggvény

- Valósítsuk meg a 3. óra anyagában leírt eloszlásfüggvényt.
- A program olvason be egy 1 oszlopot tartalmazó, valós értékeket tartalmazó file-t
- Emellett kapja meg paraméterként az eloszlásfüggvény pontosságát (pl 0.01)
- Számoljuk ki a minta eloszlását (gyakoriság függvényét).