Laboratorul 11

Pentru a putea efectua lucrarea de laborator trebuie urmati urmatorii pasi:

1. Descarcă şi instalează ActivePython 2.7.6 de la adresa www.activestate.com/activepython/downloads
2. Lansează ActivePython: **Start -> All Programs -> ActiveState ActivePython 2.7 (32-bit) -> IDLE (Python GUI)**
3. Creează un fişier nou: **File -> New File**
4. Scrie în fişierul nou creat: print "Hello World"
5. Salvează fişierul: **CTRL+S -> hello.py**
6. Lanseaza programul Python creat: **Run -> Run module (F5)**
7. Pentru exemplele cu parte grafică trebuie adusa o biblioteca:
   a. Descarcă graphics.py de la adresa http://mcsp.wartburg.edu/zelle/python/graphics.py
   b. Copie fişierul descărcat în C:\Python27\Lib\site-packages

Sa se testeze urmatoarele exemple:

```
1.  import math
2.  from graphics import *
3.  def square(x):
4.      return x * x
5.  def distance(p1, p2):
6.      dist = math.sqrt(square(p2.getX() - p1.getX())
7.                       + square(p2.getY() - p1.getY()))
8.      return dist
9.  def main():
10.     win = GraphWin("Draw a Triangle")
11.     win.setCoords(0.0, 0.0, 10.0, 10.0)
12.     message = Text(Point(5, 0.5), "Click on three points")
13.     message.draw(win)
14.     p1 = win.getMouse()
15.     p1.draw(win)
16.     p2 = win.getMouse()
17.     p2.draw(win)
18.     p3 = win.getMouse()
19.     p3.draw(win)
20.     triangle = Polygon(p1,p2,p3)
21.     triangle.setFill("peachpuff")
22.     triangle.setOutline("cyan")
23.     triangle.draw(win)
24.     perim = distance(p1,p2) + distance(p2,p3) + distance(p3,p1)
25.     message.setText("The perimeter is: {0:0.2f}".format(perim))
26.     win.getMouse()
27.     win.close()
28. main()
```

Urmatorul exemplu este:

```
1.  from graphics import *
2.  class Button:
3.      """A button is a labeled rectangle in a window.
```

```
    It is activated or deactivated with the activate()
    and deactivate() methods. The clicked(p) method
    returns true if the button is active and p is inside it."""
    def __init__(self, win, center, width, height, label):
        """ Creates a rectangular button, eg:
        qb = Button(myWin, centerPoint, width, height, 'Quit') """
        w,h = width/2.0, height/2.0
        x,y = center.getX(), center.getY()
        self.xmax, self.xmin = x+w, x-w
        self.ymax, self.ymin = y+h, y-h
        p1 = Point(self.xmin, self.ymin)
        p2 = Point(self.xmax, self.ymax)
        self.rect = Rectangle(p1,p2)
        self.rect.setFill('lightgray')
        self.rect.draw(win)
        self.label = Text(center, label)
        self.label.draw(win)
        self.deactivate()
    def clicked(self, p):
        "Returns true if button active and p is inside"
        return (self.active and
                self.xmin <= p.getX() <= self.xmax and
                self.ymin <= p.getY() <= self.ymax)
    def getLabel(self):
        "Returns the label string of this button."
        return self.label.getText()
    def activate(self):
        "Sets this button to 'active'."
        self.label.setFill('black')
        self.rect.setWidth(2)
        self.active = True
    def deactivate(self):
        "Sets this button to 'inactive'."
        self.label.setFill('darkgrey')
        self.rect.setWidth(1)
        self.active = False
```

Un alt exemplu:

```
# wordfreq.py
def byFreq(pair):
return pair[1]
def byFreq(pair):
return pair[1]
def main():
print("This program analyzes word frequency in a file")
```

```
7.  print("and prints a report on the n most frequent words.\n")
8.  # get the sequence of words from the file
9.  fname = raw_input("File to analyze: ")
10. print("fname")
11. text = open(fname,'r').read()
12. text = text.lower()
13. print text
14. for ch in '!"#$%&()*+,-./:;<=>?@[\\]^_`{|}~':
15. text = text.replace(ch, ' ')
16. words = text.split()
17. # construct a dictionary of word counts
18. counts = {}
19. for w in words:
20. counts[w] = counts.get(w,0) + 1
21. # output analysis of n most frequent words.
22. n = eval(input("Output analysis of how many words? "))
23. items = list(counts.items())
24. items.sort()
25. items.sort(key=byFreq, reverse=True)
26. for i in range(n):
27. word, count = items[i]
28. print("{0:<15}{1:>5}".format(word, count))
29. if __name__ == '__main__': main()
```

Sa se realizeze urmatoarele teme:

Tema 1. sa se creeze un joc de tip spanzuratoarea folosinf functiile de desenare grafica in python

Tema 2. Sa se implementeze mai multe cozi de thread-uri cu prioritati diferite si sa se implemnteze mecanismul de prevenire a infomatarii (cel de imbatranire)

Tema 3. Pentru sincronizarea unor thread-uri sa se implemnteze protocolul cu simularea limitarii prioritatii (Highest locker)

Tema 4. sa se construiasca un pool de thread-uri care sa faca niste calculi simple (gen sume, inductie etc) sa puna in evidenta hazardul de curse (vez I in curs)

Tema 5. Sa se implementeze un calcul $\sum_0^n i$ cu 4 thread-uri simultane care fiecare calculeaza pe un subinterval folosind modelul master/slave

Tema 6 sa se proceseze calcul $\sum_0^n i$ simultan pentru 4 valori diferite a lui n luate dintr-o coada de catre 4 task-uri diferite (model peer)

Tema 7. Sa se realizeze o procesare dupa model pipeline a unui tablou de intregi. Primul thread din pipe inmulteste toate elemental vector V cu o constanta alpha, urmatorul thread din pipe va ordona vectorul iar final ultimul thread il va afisa in coordinate x si y

Hint: in afara de curs consultati si

https://docs.python.org/2/c-api/init.html#threads

http://dabeaz.blogspot.ro/2010/02/revisiting-thread-priorities-and-new.html

Tema pe acasa. sa se creeze un joc cu pietre (de exemplu sa le pot muta cate una pentru a le rearanaja sa iasa imagine, numere intr-o ordine etc..)