CSE 512 – Winter 2019 – Lab 2

Instructor: Kerstin Voigt
Tuesdays 1:30-3:20pm in JB 359

Work on the following two exercises for this lab.

Exercise 1:

Copy file 'graphics.py' into your directory for CSE 515. From within this directory, open Python's Idle (or equivalent), and copy program 'catch_me.py into your own version of this file. Correct all errors/typos until your programs loads and runs as intended.

You will find the code for catch_me.pl at the end of this document.

Exercise 2:

Answer the questions in document "A First Round of Python Questions", based on the example programs who have seen up to know. For some questions, you will want to consult any of the Python online tutorials, or other sources.

Completion of Exercise 2 is also your first homework assignment (see "Homework Assignments" on Blackboard).

Credit for this lab: (1) Work diligently on the exercises above. (2) Sign your name of the signup-sheet which will be circulated toward the end of this lab session.

Python code for catch_me.py on the next page.

```
# catch_me.py

# by Kerstin Voigt, Jan 2019; a first program for CSE 512 students
# objective is to catch the moving black with the grey cup;
# best player's scores is kept and updated;

from graphics import *
import random
import time
import pickle

WORLD_MAX_X = 500
WORLD_MAX_Y = 500
STEP = 30
CUP_MAX = 20

BEST_SCORE = 1000

# the "dotbug" robot;

class DotBug:
    def __init__(self,loc = Point(100,100), col="black"):
        self.location = loc
        self.color = col
        self.the_dotbug = Oval(self.location,\
                               Point(self.location.x + 20,\
                                     self.location.y + 20))

    def __str__(self):
        return "dotbug at (%d,%d)" % (self.location.x,\
                                      self.location.y)

    def update_dotbug(self):
        self.the_dotbug.move(self.location.x - self.the_dotbug.p1.x,\
                             self.location.y - self.the_dotbug.p1.y)

    def draw(self):
        self.update_dotbug()
        self.the_dotbug.setFill(self.color)
        self.the_dotbug.draw(win)

    def undraw(self):
        self.the_dotbug.undraw()

    def jump(self):
        if random.randint(0,1) == 0:
            for i in range(random.randint(2,10)):
                self.move_up()
        else:
            for i in range(random.randint(2,10)):
                self.move_down()
        if random.randint(0,1) == 0:
```

```
            for i in range(random.randint(2,10)):
                self.move_left()
        else:
            for i in range(random.randint(2,10)):
                self.move_right()
        self.update_dotbug()

    def move_up(self):
        newloc = Point(self.location.x, self.location.y - STEP)
        if self.location.y >= STEP:
            self.location = newloc

    def move_down(self):
        newloc = Point(self.location.x, self.location.y + STEP)
        if self.location.y <= WORLD_MAX_Y - STEP:
            self.location = newloc

    def move_left(self):
        newloc = Point(self.location.x - STEP, self.location.y)
        if self.location.x >= STEP:
            self.location = newloc

    def move_right(self):
        newloc = Point(self.location.x + STEP, self.location.y)
        if self.location.x <= WORLD_MAX_X - STEP:
            self.location = newloc

    def is_caught(self, cup):
        if cup.cup_covers(self.location):
            cup.undraw()
            cup.color = "green"
            cup.draw()
            txt = Text(Point(self.location.x + 20,\
                       self.location.y + 20),\
                       "YOU GOT IT!!")
            txt.draw(win)
            return True
        else:
            return False

    def taunt(self):
        tnt = Text(Point(random.randint(100,400),\
                         random.randint(100,400)), "Catch me ;-)")
        tnt.draw(win)
        self.draw()
        time.sleep(0.8)
        tnt.undraw()
        self.undraw()


class Cup:
    def __init__(self, loc = Point(WORLD_MAX_X/2,WORLD_MAX_Y/2),\
                 rad = 60,col="grey"):
```

```
        self.radius = rad
        self.color = col
        self.location = loc
        self.tries = CUP_MAX
        self.the_cup = Oval(loc,Point(loc.x + rad, loc.y + rad))

    def __str__(self):
        return "cup at (%d,%d)" % (self.location.x,\
                                   self.location.y)

    def update_cup(self):
        self.the_cup.move(self.location.x - self.the_cup.p1.x,\
                          self.location.y - self.the_cup.p1.y)

    def draw(self):
        self.update_cup()
        self.the_cup.setFill(self.color)
        self.the_cup.draw(win)

    def undraw(self):
                    self.the_cup.undraw()

    def cup_down(self):
        if self.tries > 0:
            click = win.getMouse()
            self.location = click
            self.tries-=1
            self.undraw()
            self.draw()
            return True
        else:
            self.cup_lost()
            return False

    def cup_covers(self, bugpt):
        if pow(bugpt.x - self.location.x,2) + \
           pow(bugpt.y - self.location.y,2) <= pow(self.radius,2):
            return True
        else:
            return False

    def cup_lost(self):
        self.undraw()
        self.color = "purple"
        self.draw()
        txt = Text(Point(self.location.x + 20, self.location.y + 20),\
                   "No Luck ... hehehe")
        txt.draw(win)


if __name__ == '__main__':

    mp = open("catch_me.pickle", "wb")
```

```
    pickle.dump(BEST_SCORE,mp)
    mp.close()

    win = GraphWin("Catch Me!!", 500, 500)
    bug = DotBug(Point(random.randint(100,400),\
                       random.randint(100,400)))
    mycup = Cup()
    mycup.draw()

    again = True
    while not bug.is_caught(mycup) and again:
        bug.taunt()
        bug.jump()
        again = mycup.cup_down()
        print "bug at (%d,%d)" % (bug.location.x, bug.location.y)
        print "cup at (%d,%d)" % (mycup.location.x, mycup.location.y)

    if bug.is_caught(mycup):
        mp = open("catch_me.pickle", "rb")
        best = pickle.load(mp)
        mp.close()
        if CUP_MAX - mycup.tries < best:
            print "\n\nBEST SCORE at %d tries!!!\n" % (CUP_MAX -
mycup.tries)
            mp = open("catch_me.pickle", "wb")
            pickle.dump(mycup.tries,mp)
            mp.close()
        else:
            print "\n\nOthers have been better ...\n"

    # one click to exit
    click = win.getMouse()
    win.close()
```