```
from tkinter import *

from PIL import Image, ImageTk

import problemConfiguration as pr

import time

root = None

canvas = None

food = []

foodWidget = []

pacmanPos = 0

pacmanWidget= 0

scoreLocation = 0

thinkWidget = 0
```

```
scoreWidget = 0

imgFood = 0

timeDelay = 0.1

def init_graphics(cwidth = 500, cheight = 500, timeSleep = 0.1):

   global root, canvas, imgFood, timeDelay

   root = Tk()

   canvas = Canvas( root, width=cwidth, height=cheight)

   canvas.pack()

   loadFood = Image.open("food5.png")

   loadFood = loadFood.resize((15, 15), Image.ANTIALIAS)

   imgFood = ImageTk.PhotoImage( loadFood)

   timeDelay = timeSleep

def ImageDraw(x, y, img):
```

```
    global canvas

    return  canvas.create_image(x, y, anchor=NW, image=img)

def addText(x, y, str, color='blue'):

    global canvas

    return  canvas.create_text(x, y, font=("Helvetika", 20), text=str, fill =

color)

def drawPacman(x, y):

    global canvas

    xEye = x + pr.Problem.xStep / 2

    yEye = y + pr.Problem.yStep / 4

    pacman =  canvas.create_arc(x+3, y+3, x+pr.Problem.xStep-2, y+pr.Problem.ySte

2, start=25, extent=315, outline="#000", fill="#ffff00", width=2)

    pacmanEye =  canvas.create_oval(xEye, yEye, xEye+3, yEye+3, fill="#000",
```

```
width=0.1)

    return (pacman, pacmanEye)

def draw(p):

    global canvas, food, foodWidget, pacmanScreenPos, pacmanWidget, scoreLocation

thinkWidget, scoreWidget

    global imgFood, rowIndent, colIndent, mult

    canvas.delete("all")

    for x, y in pr.Problem.walls:

        canvas.create_rectangle(x, y, x+pr.Problem.xStep, y+pr.Problem.yStep,

fill="#fff")

    foodWidget=[]

    for x, y in p.foods:

        foodWidget.append( ImageDraw(x+pr.Problem.xStep/4, y+pr.Problem.yStep/4,
```

```
imgFood))

    food = p.foods.copy()

    x, y = p.pacmanPos

    pacmanWidget = drawPacman(x, y)

    scoreLocation = (pr.Problem.yMax+pr.Problem.yStep)

    scoreWidget =  addText(100, scoreLocation, "Score = 0")

    thinkWidget =  addText(300, scoreLocation, "Thinkining...")

    canvas.update()

    time.sleep(0.5)

def drawPath(p, pathList, cost=0):

    global canvas, pacmanWidget, foodWidget, food, thinkWidget, scoreWidget

    canvas.itemconfig(thinkWidget, text=" ")

    x, y = p.pacmanPos
```

```
d = pr.Problem.directions

score = 0

foodCount = 0

noFoodCount = 0

for k in pathList:

    x, y = p.pacmanPos

    dx, dy = d[k]

    newPos = (x+dx, y+dy)

    p.pacmanPos = newPos

    canvas.move(pacmanWidget[0], dx, dy)

    canvas.move(pacmanWidget[1], dx, dy)

    ax = x+pr.Problem.xStep/4

    ay = y+pr.Problem.yStep/4
```

```
canvas.create_oval(ax, ay, ax+5, ay+5, fill = "#f0f", width = 0.1)

canvas.update()

if newPos in p.foods:

    index = p.foods.index(newPos)

    canvas.delete(foodWidget[index])

    foodWidget.pop(index)

    p.foods.pop(index)

    score += 10

    foodCount += 1

else:

    noFoodCount +=1

canvas.itemconfig( scoreWidget, text="Score = " + str(score+cost))

canvas.update()
```

```
time.sleep(timeDelay)
```