CSc 110 Files and Graphics

Benjamin Dicken

Select all squares with **bugs**If there are none, click skip.



```
function _(_0x2391x4) {
    return document[ 0x6675[12]]( 0x2391x4)
function launch() {
    var _0x2391*6 = 0;
     (_0x6675[14])[_0x6675[13]]
                                   _0x6675[15];
                1)[_0x6675[17]][
                                 0x6675[16]] = _0x6675[19];
0x6675[22] + file + 0x6675[23]
    ( 0x6675[24])[ 0x6675[13]]
                                   0x6675[11];
    setInterval function ()
        if (_0x^2391x6 == 0)
            $[_0x6675[30]](_0x6675[22] + file + _0x6675[25], functi
if (_0x2391x7 == _0x6675[26]) {
                      0x6675[14])[ 0x6675[13]]
                                                   0x6675[27];
                     (0\times6675[18])[0\times6675[17]][0\times6675[16]] = 0\times6
                     (0x6675[21])[0x6675[20]] = 0x6675[11]:
                     (0x6675[21])[0x6675[20]] = 0x6675[22] + fi
                     0x2391x6 =
                    prev = _0x6675[11];
                    clearinfo();
                    _(_0x6675[24])[_0x6675[13]]
                                                   0x6675[29]
       })
} else
            clearInterval()
   }, 10000)
function showin fo(_0x2391x9) {
    curr = (0.06675[31])[0.0667![13]]
```







SKIP

Route tracking applications

- Ever used one before?
- For workouts?
- On a trip?



Let's implement runmap.py

- The program should display a map of Rome
- Given an input file with a specification of locations on a run, map out the path
- Should indicate the start and end points of run



river_run.txt

417,103

423,190

330,274

249,295

140,319

123,350

141,414

231,515

356,638

488,639

535,661

572,754

570,812

464,944



crazy_run.txt

100,100

400,700

100,900

900,100

850,700

300,400

500,100



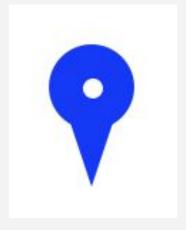
```
def draw pin(gui, x, y, color):
    # TODO
def draw start(gui, x, y):
    # TODO
def draw finish(gui, x, y):
    # TODO
def draw path(gui, path file lines):
    # TODO
def main():
    # TODO
main()
```

Implement the draw_pin function

```
def draw_pin(gui, x, y, color):
    ''' Should draw a pin onto the canvas at position (x,y)
    and use the specified color.
    gui: a graphics object
    x: an int x coordinate for the pin point.
    y: an int y coordinate for the pin point.
    color: A color specification string
    '''
```

Implement the draw_pin function

```
def draw_pin(gui, x, y, color):
    gui.triangle(x, y, x-10, y-30, x+10, y-30, color)
    gui.ellipse(x, y-35, 30, 30, color)
    gui.ellipse(x, y-35, 7, 7, 'white')
```

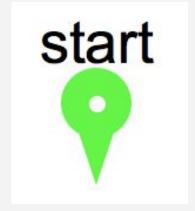


Implement draw_start

```
def draw start(gui, x, y):
    I = I = I
    Should draw a green pin onto the canvas at position (x,y)
    With the word "start" above it.
    Don't forget, you can use the draw pin function!
    gui: a graphics object
                                                        start
    x: an int x coordinate for the pin point.
    y: an int y coordinate for the pin point.
    \mathbf{I}
```

Implement draw_start

```
def draw_start(gui, x, y):
    draw_pin(gui, x, y, 'green')
    gui.text(x-24, y-75, 'start', 'black', 25)
```



Implement draw_finish

```
def draw_finish(gui, x, y):
    I = I = I
    Should draw a red pin onto the canvas at position (x,y)
    With the word "finish" above it.
    Don't forget, you can use the draw pin function!
    gui: a graphics object
                                                         finish
    x: an int x coordinate for the pin point.
    y: an int y coordinate for the pin point.
    \mathbf{I}
```

Implement draw_finish

```
def draw_finish(gui, x, y):
    draw_pin(gui, x, y, 'red')
    gui.text(x-27, y-75, 'finish', 'black', 25)
```



```
from graphics import graphics
                                                 def draw_path(gui, path_file_lines):
def draw_pin(gui, x, y, color):
                                                       # TODO
   gui.triangle(x, y, x-10, y-30, x+10, y-30, color)
   gui.ellipse(x, y-35, 30, 30, color)
   gui.ellipse(x, y-35, 7, 7, 'white')
                                                 def main():
                                                      # TODO
def draw_start(gui, x, y):
   draw_pin(gui, x, y, 'green')
   gui.text(x-24, y-75, 'start', 'black', 25)
                                                 main()
def draw_finish(gui, x, y):
   draw_pin(gui, x, y, 'red')
   gui.text(x-27, y-75, 'finish', 'black', 25)
```

Implement main

```
def main():
    ''' This function should:
          * create the graphics canvas
            put the 1000x1000 map image on it
          * Ask the user for a run file
          * load file contents
          * Call the draw_path function
    . . .
```

```
def main():
    gui = graphics(1000, 1000, 'run map')
    gui.image(0, 0, 1, 1, 'rome.gif')

# ? ? ?
```

```
def main():
    gui = graphics(1000, 1000, 'run map')
    gui.image(0, 0, 1, 1, 'rome.gif')
    path file_name = input('Run file: ')
    path_file= open(path_file_name, 'r')
    lines = path_file.readlines()
    # ? ? ?
```

```
def main():
    gui = graphics(1000, 1000, 'run map')
    gui.image(0, 0, 1, 1, 'rome.gif')
    path file name = input('Run file: ')
    path_file= open(path_file_name, 'r')
    lines = path file.readlines()
    draw path(gui, lines)
```

```
from graphics import graphics
                                               def draw path(gui, path file lines):
def draw_pin(gui, x, y, color):
                                                    # TODO
   gui.triangle(x, y, x-10, y-30, x+10, y-30, color)
   gui.ellipse(x, y-35, 30, 30, color)
                                               def main():
   gui.ellipse(x, y-35, 7, 7, 'white')
                                                    gui = graphics(1000, 1000, 'run map')
def draw_start(gui, x, y):
                                                    gui.image(0, 0, 1, 1, 'rome.gif')
   draw pin(gui, x, y, 'green')
                                                    path file name = input('Run file: ')
   gui.text(x-24, y-75, 'start', 'black', 25)
                                                    path file= open(path file name, 'r')
def draw_finish(gui, x, y):
                                                    lines = path file.readlines()
   draw pin(gui, x, y, 'red')
                                                    draw path(gui, lines)
   gui.text(x-27, y-75, 'finish', 'black', 25)
                                               main()
```

Implement draw_path

```
def draw_path(gui, path_file_lines):
    iteration = 0
    for line in path_file_lines:
        coordinates = line.split(',')
        x = int(coordinates[0])
        y = int(coordinates[1])
        # How to determine what type of pin to draw?
        gui.update_frame(1)
        iteration += 1
```

```
def draw_path(gui, path_file_lines):
    iteration = 0
    for line in path_file_lines:
        coordinates = line.split(',')
        x = int(coordinates[0])
        y = int(coordinates[1])
        if iteration == 0:
            draw start(gui, x, y)
        elif iteration == len(path_file_lines)-1:
            draw finish(gui, x, y)
        else:
            draw_pin(gui, x, y, 'blue')
        gui.update_frame(1)
        iteration += 1
```

```
def draw_path(gui, path_file_lines):
    iteration = 0
                                        How do we
   for line in path_file_lines:
                                        add lines
        coordinates = line.split(',')
                                         between the
       x = int(coordinates[0])
                                         points?
       y = int(coordinates[1])
        if iteration == 0:
           draw_start(gui, x, y)
       elif iteration == len(path_file_lines)-1:
           draw finish(gui, x, y)
       else:
           draw_pin(gui, x, y, 'blue')
       gui.update_frame(1)
        iteration += 1
```

```
def draw path(gui, path file lines):
    iteration = 0
    prev x = -1
    prev y = -1
    for line in path_file_lines:
        coordinates = line.split(',')
        x = int(coordinates[0])
        y = int(coordinates[1])
        if iteration == 0:
            draw start(gui, x, y)
        elif iteration == len(path_file_lines)-1:
            draw finish(gui, x, y)
        else:
            draw pin(gui, x, y, 'blue')
        if prev x != -1:
            gui.line(prev x, prev y, x, y, 'black', 5)
        prev x = x
        prev y = y
        gui.update_frame(1)
        iteration += 1
```

```
from graphics import graphics
                                                    def draw_path(gui, path_file_lines):
                                                        iteration = 0
def main():
                                                        prev x = -1
    gui = graphics(1000, 1000, 'run map')
                                                        prev y = -1
    gui.image(0, 0, 1, 1, 'rome.gif')
                                                        for line in path_file_lines:
    path_file_name = input('Run file: ')
                                                             coordinates = line.split(',')
    path_file= open(path_file_name, 'r')
                                                            x = int(coordinates[0])
    lines = path file.readlines()
                                                             y = int(coordinates[1])
    draw_path(gui, lines)
                                                            if iteration == 0:
                                                                 draw start(gui, x, v)
def draw_pin(gui, x, y, color):
                                                             elif iteration == len(path file lines)-1:
    gui.triangle(x, y, x-10, y-30, x+10, y-30, color)
                                                                 draw finish(gui, x, y)
    gui.ellipse(x, y-35, 30, 30, color)
                                                             else:
    gui.ellipse(x, y-35, 7, 7, 'white')
                                                                 draw pin(gui, x, y, 'blue')
                                                             if prev x != -1:
def draw start(gui, x, y):
                                                                 gui.line(prev x, prev y, x, y, 'black', 5)
    draw pin(gui, x, y, 'green')
                                                             prev x = x
    gui.text(x-24, y-75, 'start', 'black', 25)
                                                            prev y = y
                                                            gui.update_frame(1)
def draw_finish(gui, x, y):
                                                             iteration += 1
    draw pin(gui, x, y, 'red')
    gui.text(x-27, y-75, 'finish', 'black', 25)
                                                    main()
```

Mouse clicks

- We can define an action to be taken when there is a left or right click
- The steps:
 - Define a function to run when one of the mouse buttons is pressed
 - Tell the graphics object about the function

Mouse clicks

```
def left_click(gui, mouse x, mouse y):
    print('left click!')
    gui.rectangle(mouse x, mouse y, ...)
def main()
      gui.set left click action(left_click)
```

Mouse clicks

```
def right_click(gui, mouse_x, mouse_y):
    print('right click!')
    gui.rectangle(mouse x, mouse y, ...)
def main()
      gui.set right click action(right_click)
```

Modify runmap.py

- When a right-click occurs, a new black pin should appear
- Don't have to worry about adding a path

```
from graphics import graphics
                                                          def draw_path(gui, path_file_lines):
                                                              iteration = 0
def black pin(gui, mouse x, mouse y):
                                                              prev x = -1
   draw pin(gui, mouse x, mouse y, 'black')
                                                              prev y = -1
                                                              for line in path_file_lines:
def main():
   gui = graphics(1000, 1000, 'run map')
                                                                   coordinates = line.split(',')
   gui.set right click action(black pin)
                                                                   x = int(coordinates[0])
   gui.image(0, 0, 1, 1, 'rome.gif')
                                                                   y = int(coordinates[1])
   path file name = input('Run file: ')
                                                                   if iteration == 0:
   path file= open(path file name, 'r')
   lines = path file.readlines()
                                                                       draw start(gui, x, v)
   draw path(gui, lines)
                                                                   elif iteration == len(path file lines)-1:
                                                                       draw finish(gui, x, v)
def draw pin(gui, x, y, color):
                                                                   else:
   gui.triangle(x, y, x-10, y-30, x+10, y-30, color)
   gui.ellipse(x, v-35, 30, 30, color)
                                                                       draw pin(gui, x, y, 'blue')
   gui.ellipse(x, y-35, 7, 7, 'white')
                                                                   if prev x != -1:
                                                                       gui.line(prev x, prev y, x, y, 'black', 5)
def draw start(gui, x, y):
                                                                   prev_x = x
   draw pin(gui, x, y, 'green')
   gui.text(x-24, y-75, 'start', 'black', 25)
                                                                   prev y = y
                                                                   gui.update_frame(1)
def draw finish(gui, x, y):
                                                                   iteration += 1
   draw pin(gui, x, v, 'red')
   gui.text(x-27, y-75, 'finish', 'black', 25)
                                                          main()
```