CSc 110 Files and Graphics

Benjamin Dicken

Route tracking applications

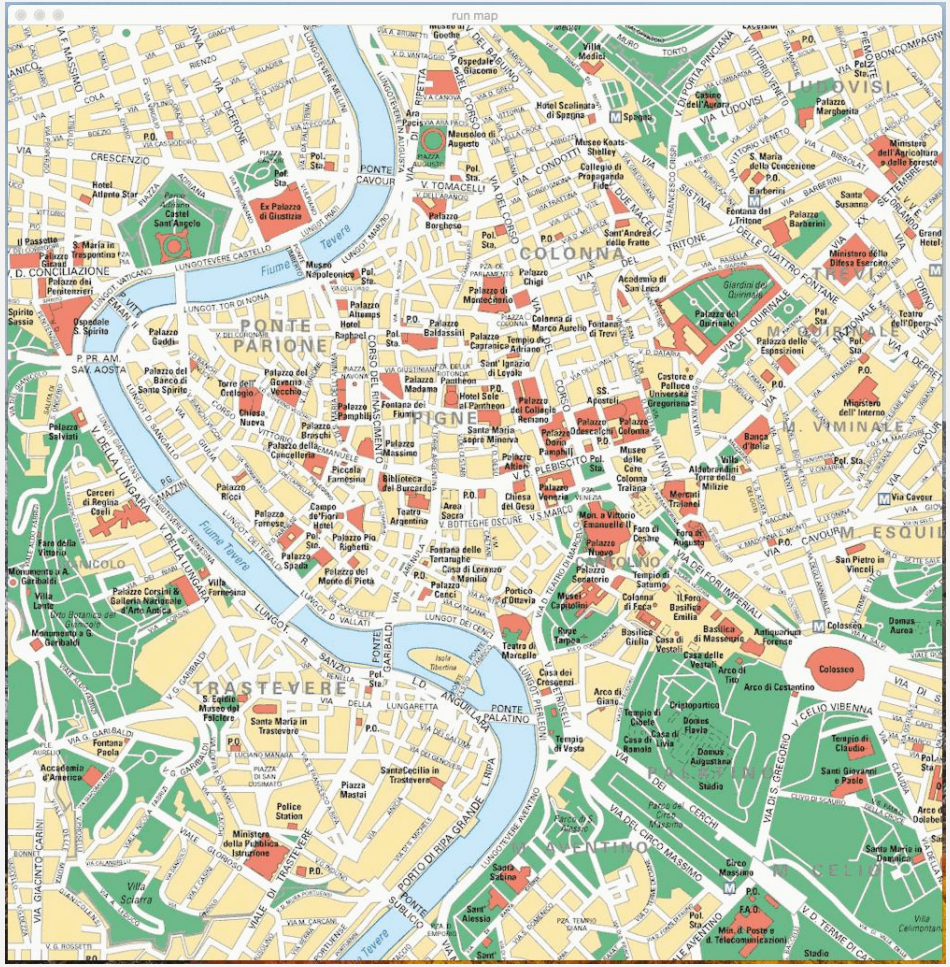
- Ever used one before?
- For workouts?
- On a trip?

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

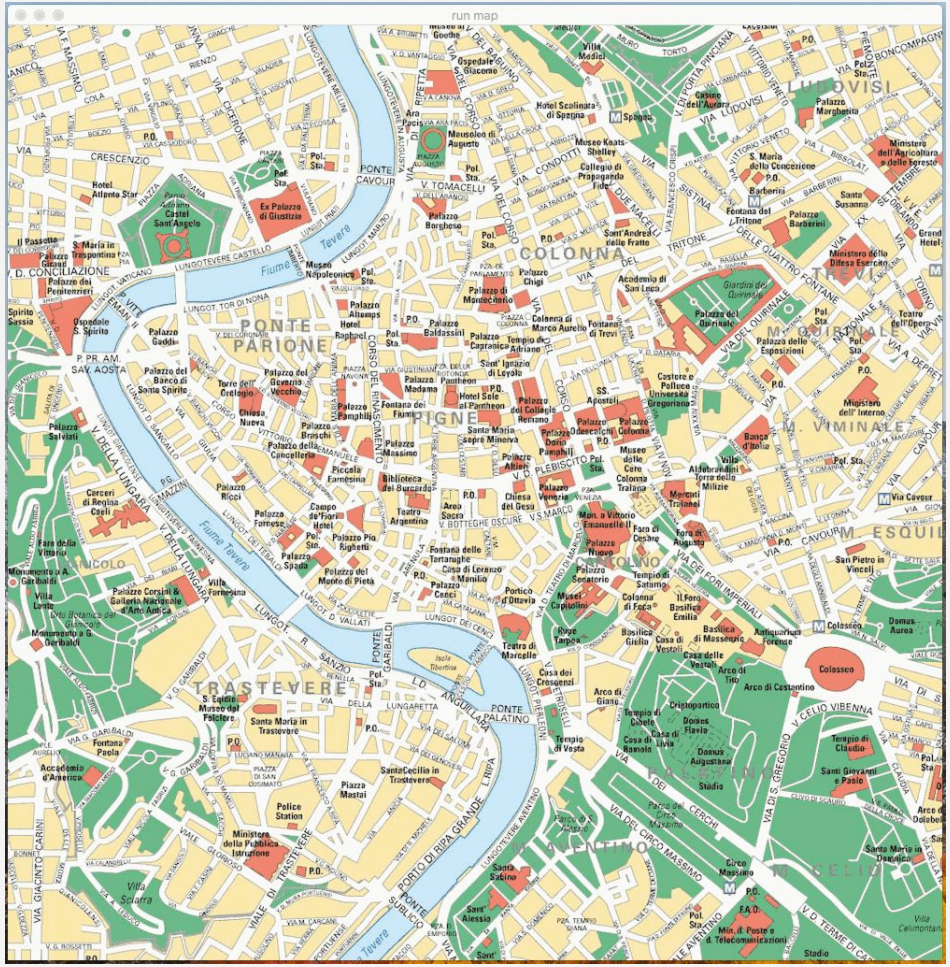
river_run.txt

```
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):
    '''
    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
    x: an int x coordinate for the pin point.
    y: an int y coordinate for the pin point.
    '''
```

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):
    '''
    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
    x: an int x coordinate for the pin point.
    y: an int y coordinate for the pin point.
    '''
```

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_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')

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

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

```
def draw_path(gui, path_file_lines):
    # TODO

def main():
    # TODO

main()
```

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_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')

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

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

```
def draw_path(gui, path_file_lines):
    # TODO

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)

main()
```

Implement **draw_path**

```
def draw_path(gui, path_file_lines):
    ''' This function should:
        * For each line of the input file
            * Get the x and y coordinate
            * Determine which type of pin to draw
    gui: a graphics object
    path_file_lines: The lines form the path
                     file in a list
    '''
```

```
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
```

Activity

```
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
```

How do we add lines between the points?

```
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 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)

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')

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

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

```
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

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 black_pin(gui, mouse_x, mouse_y):
    draw_pin(gui, mouse_x, mouse_y, 'black')

def main():
    gui = graphics(1000, 1000, 'run map')
    gui.set_right_click_action(black_pin)
    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)

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')

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

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

```
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

main()
```