Odd Semester (2020)



**BINUS UNIVERSITY**

**BINUS INTERNATIONAL**

**Assignment Cover Letter**

**(Individual Work****)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | |  | |  | |
| **Student Information**: **Surname** | | | | | **Given Names**  **Ryan** | | **Student ID Number**  **2101704413** | |
| 1. | | **Divas Tjahya** |  | |
|  |  |
| **Course Code** | **: COMP6502** |  |  | | **Course Name** | | **: Introduction to Programming** | |
| **Class** | **: L1AC** |  |  | | **Name of Lecturer(s)** | | **:** 1. Bagus Kerthyayana | |
|  |  |  |  | |  | | 2. Tri Asih Budiono | |
| **Major** | **: CS** |  |  | |  | |  | |
| **Title of Assignment**  (if any) | : Boxy Maze | |  |  | |  | |  | |
| **Type of Assignment**    **Submission Pattern** | **: Final Project** |  |  | |  | |  | |
| **Due Date** | **: 7-11-2017** |  |  | | **Submission Date** | | **: 7-11-2017** | |

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer’s instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

# Plagiarism/Cheating

BiNus International seriously regards all forms of plagiarism, cheating and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

# Declaration of Originality

By signing this assignment, I understand, accept and consent to BiNus International terms and policy on plagiarism. Herewith I declare that the work contained in this assignment is my own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student:

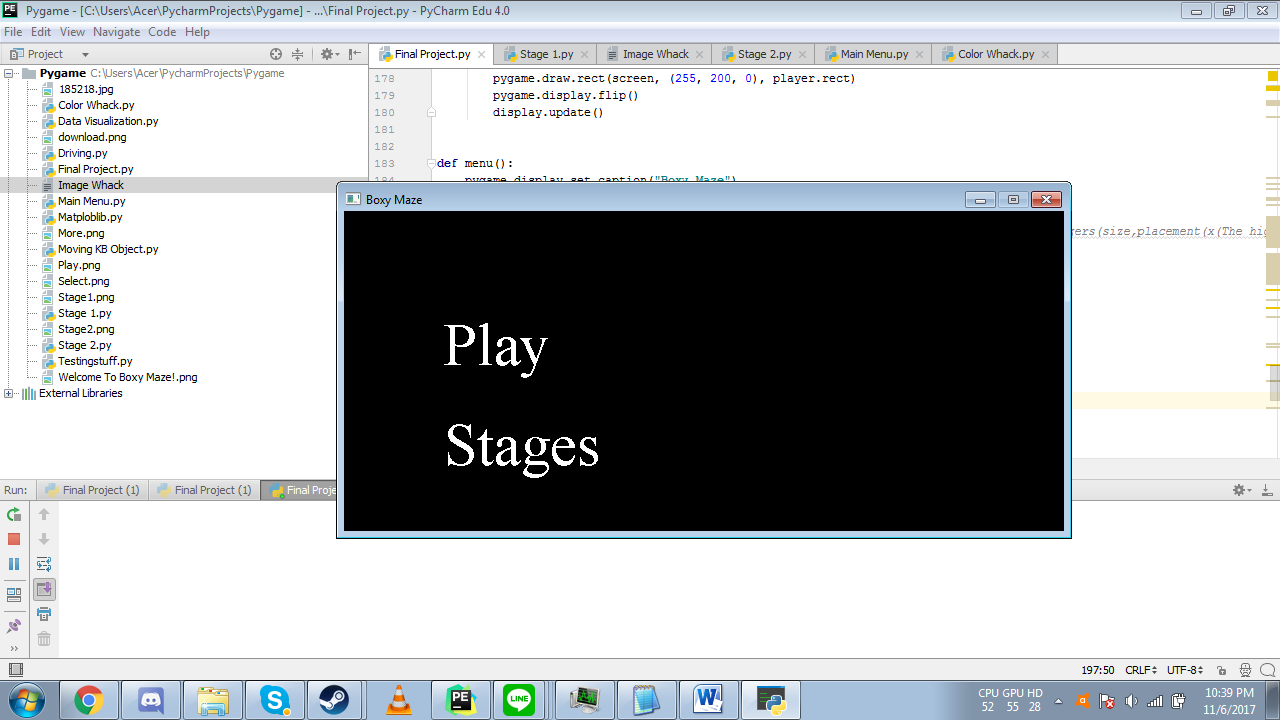
(Name of Student)

1. Ryan Divas Tjahya

* The Whole Point of the game is to reach the red box

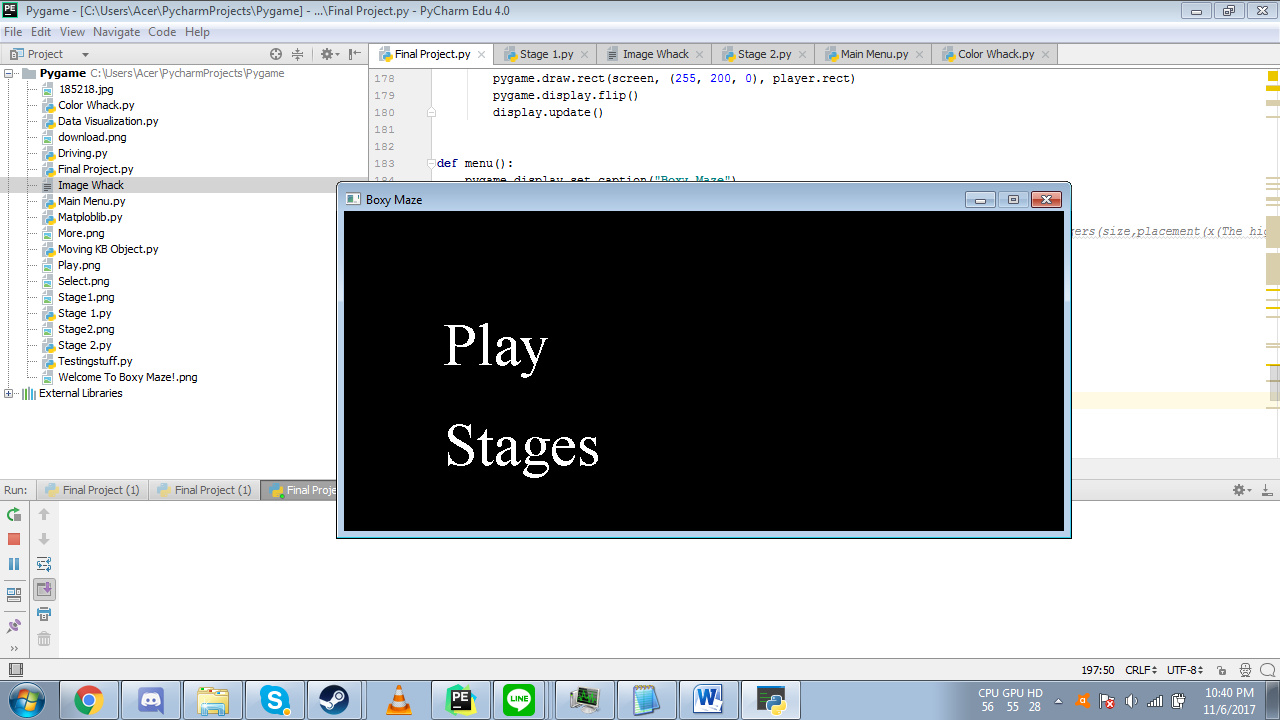
This is how the Opening screen is created

**def** menu():  
 pygame.display.set\_caption(**"Boxy Maze"**)  
 screen = pygame.display.set\_mode((720, 320))  
 screen.fill((0, 0, 0))  
 play = textsprite(**"Times New Roman"**, **"Play"**, 60, 100, 100, 255, 255, 255)*#For the integers(size,placement(x(The higher the input it basically goes right),y(The higher the input the lower the placement),Colour(R,G,B)* stages = textsprite(**"Times New Roman"**, **"Stages"**, 60, 100, 200, 255, 255, 255)

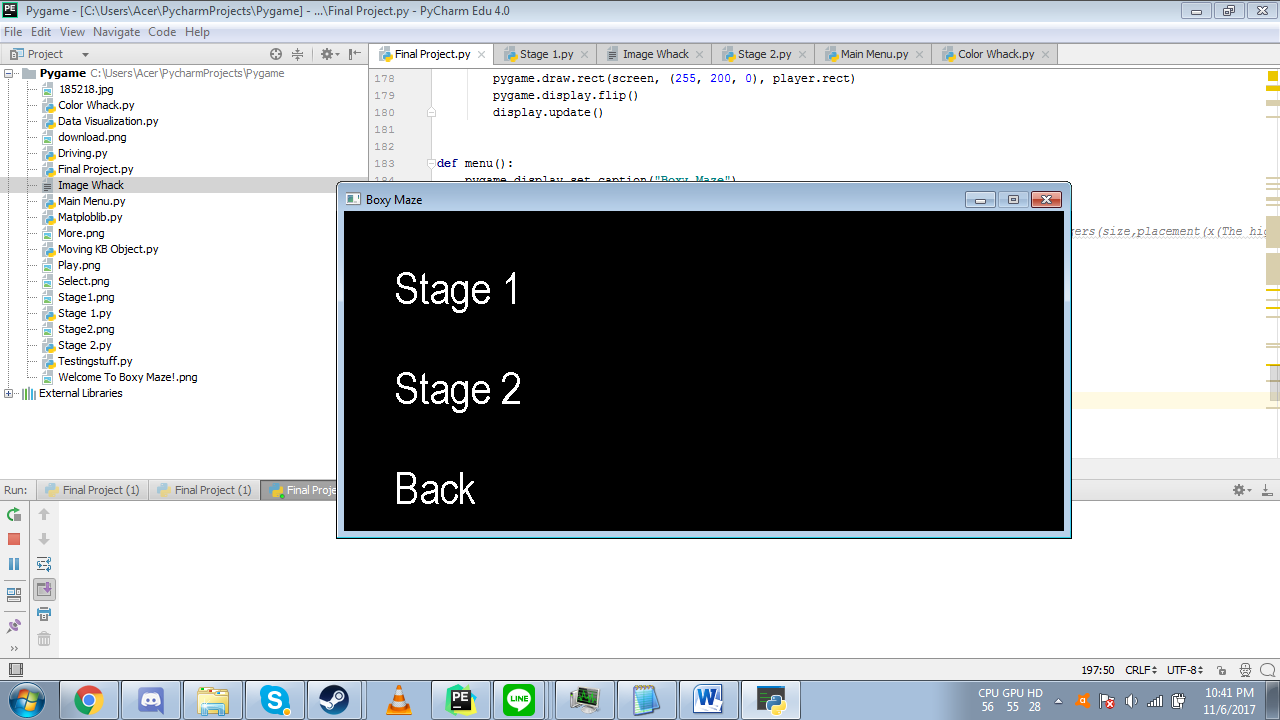


This is how you select the game

**while True**:  
 all\_text = Group(play, stages)  
 all\_text.draw(screen)  
 menu\_wait = event.wait()  
 **if** play.rect.collidepoint(mouse.get\_pos()):  
 **if** menu\_wait.type == MOUSEBUTTONDOWN:  
 stage(1)



It will Transfer you here



Then you can pick whichever stage you want to play

These are the contents of the window which opens when you click “Select”

**def** stage\_selection():  
 pygame.display.set\_caption(**"Boxy Maze"**)  
 screen = pygame.display.set\_mode((720, 320))  
 screen.fill((0, 0, 0))  
 stage1 = textsprite(**"Arial"**, **"Stage 1"**, 45, 50, 50, 255, 255, 255)  
 stage2 = textsprite(**"Arial"**, **"Stage 2"**, 45, 50, 150, 255, 255, 255)  
 back = textsprite(**"Arial"**, **"Back"**, 45, 50, 250, 255, 255, 255)  
 **while True**:  
 all\_text = Group(stage1, stage2, back)  
 all\_text.draw(screen)  
 menu\_wait = event.wait()  
 **if** stage1.rect.collidepoint(mouse.get\_pos()):  
 **if** menu\_wait.type == MOUSEBUTTONDOWN:  
 stage(1)  
 **if** stage2.rect.collidepoint(mouse.get\_pos()):  
 **if** menu\_wait.type == MOUSEBUTTONDOWN:  
 stage(2)  
 **if** back.rect.collidepoint(mouse.get\_pos()):  
 **if** menu\_wait.type == MOUSEBUTTONDOWN:  
 menu()  
 **if** menu\_wait.type == QUIT:  
 pygame.quit()  
 **break** display.update()  
menu()

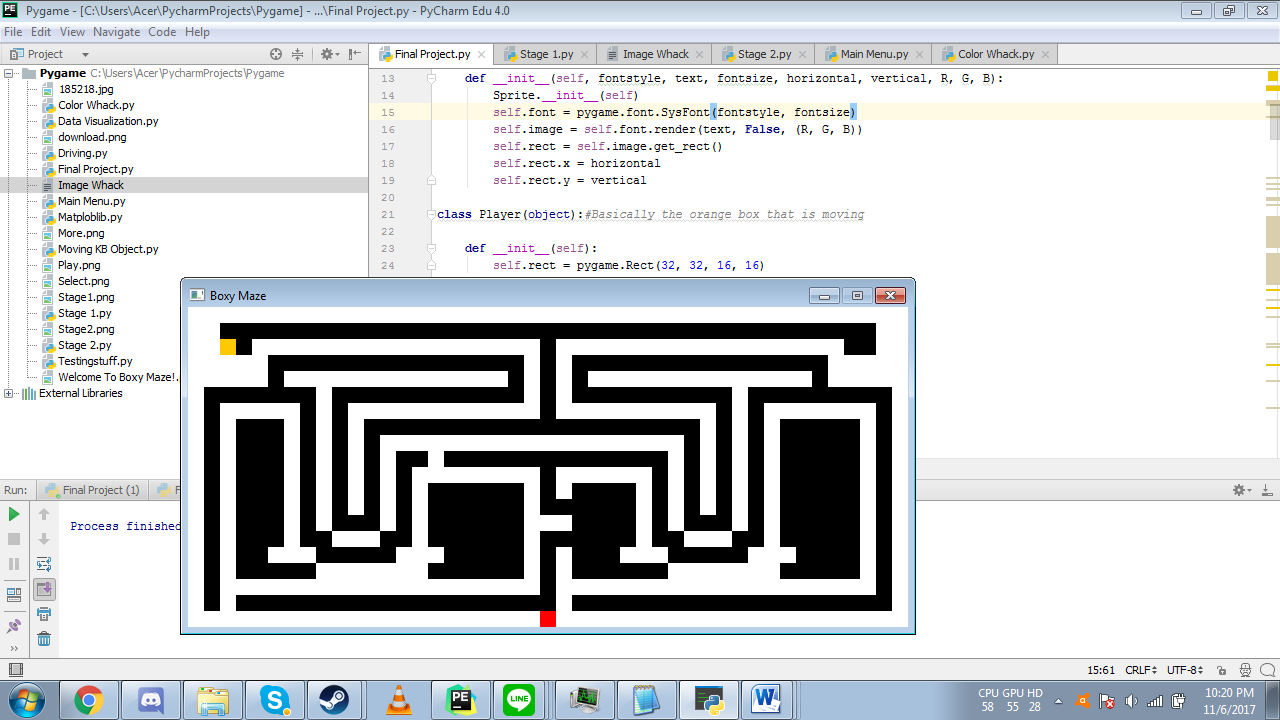
The variables/objects for the game

hitboxes = []  
blocks =[]

player = Player[]

The main character that you will be controlling

**class** Player(object):*#Basically the orange box that is moving* **def** \_\_init\_\_(self):  
 self.rect = pygame.Rect(32, 32, 16, 16)



The code that is going to detect the collision of the sprites

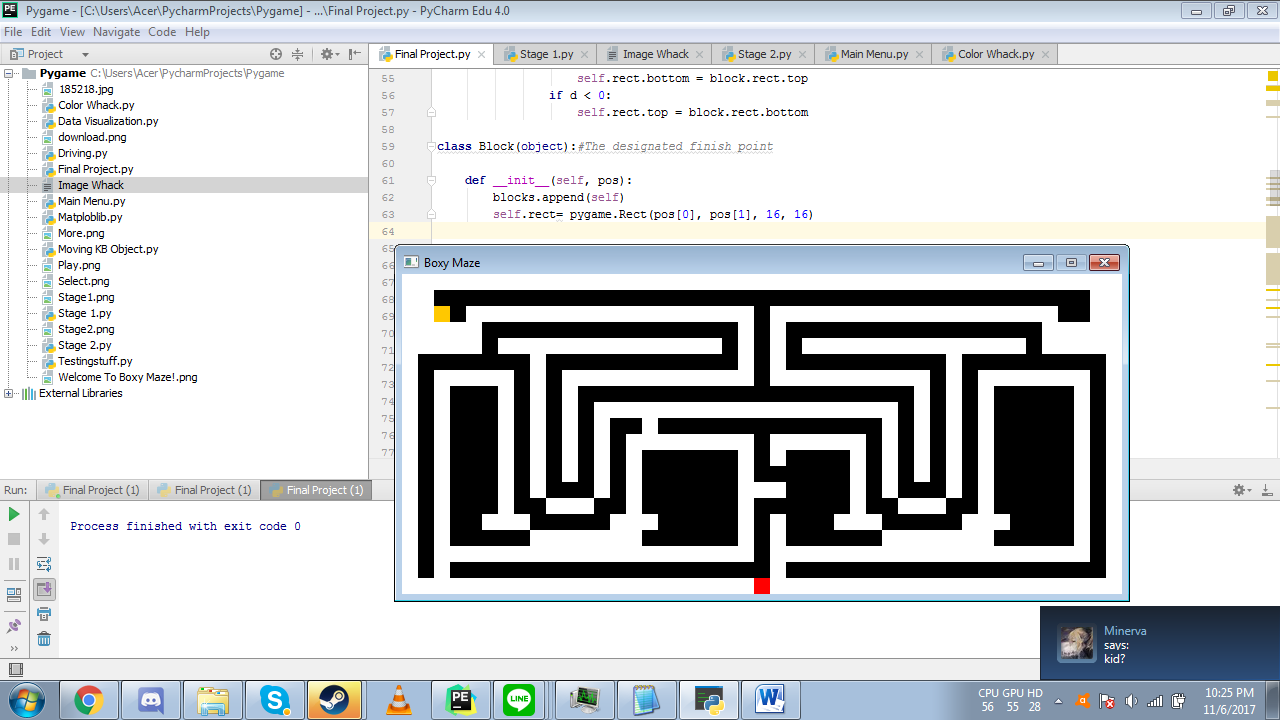
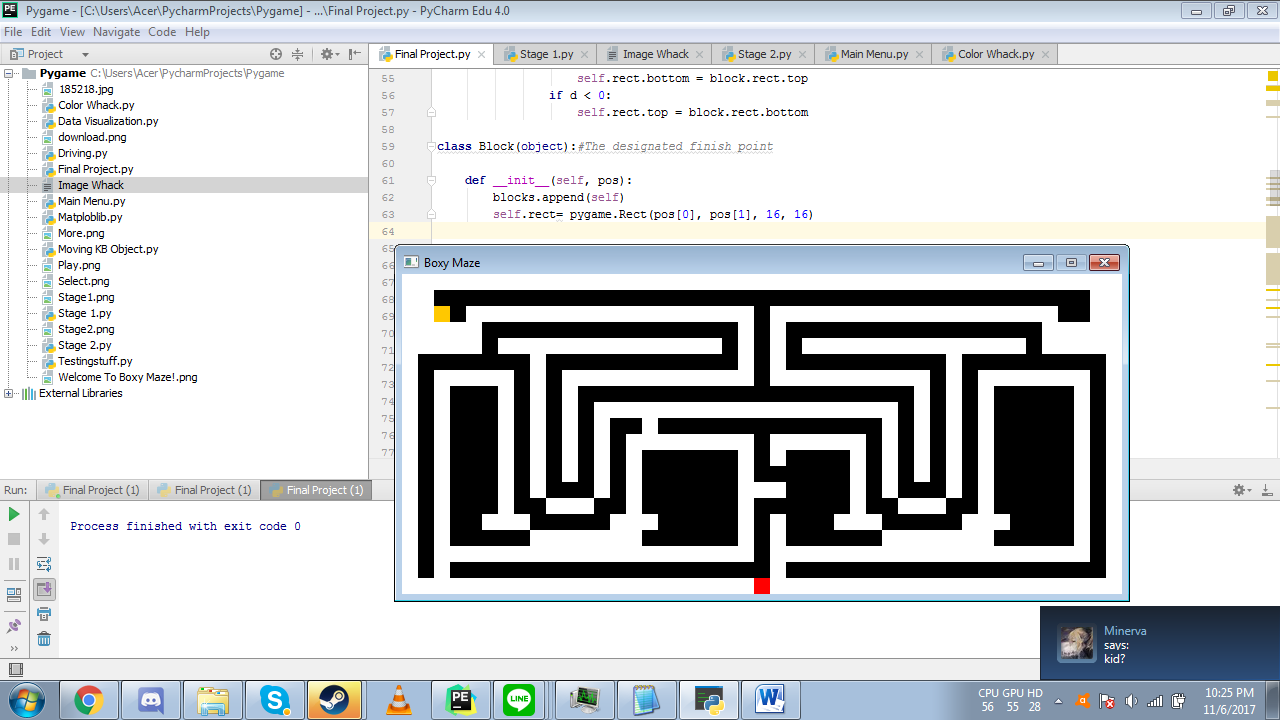
**for** hitbox **in** hitboxes:  
 **if** self.rect.colliderect(hitbox.rect):  
 **if** c > 0:  
 self.rect.right = hitbox.rect.left  
 **if** c < 0:  
 self.rect.left = hitbox.rect.right  
 **if** d > 0:  
 self.rect.bottom = hitbox.rect.top  
 **if** d < 0:  
 self.rect.top = hitbox.rect.bottom  
  
**for** block **in** blocks:  
 **if** self.rect.colliderect(block.rect):  
 **if** c > 0:  
 self.rect.right = block.rect.left  
 **if** c < 0:  
 self.rect.left = block.rect.right  
 **if** d > 0:  
 self.rect.bottom = block.rect.top  
 **if** d < 0:  
 self.rect.top = block.rect.bottom

These codes will present the texts that are going to appear in the main menu

**class** textsprite (Sprite): *#This code is reusable, just fill in the parameters to reuse* **def** \_\_init\_\_(self, fontstyle, text, fontsize, horizontal, vertical, R, G, B):  
 Sprite.\_\_init\_\_(self)  
 self.font = pygame.font.SysFont(fontstyle, fontsize)  
 self.image = self.font.render(text, **False**, (R, G, B))  
 self.rect = self.image.get\_rect()  
 self.rect.x = horizontal  
 self.rect.y = vertical

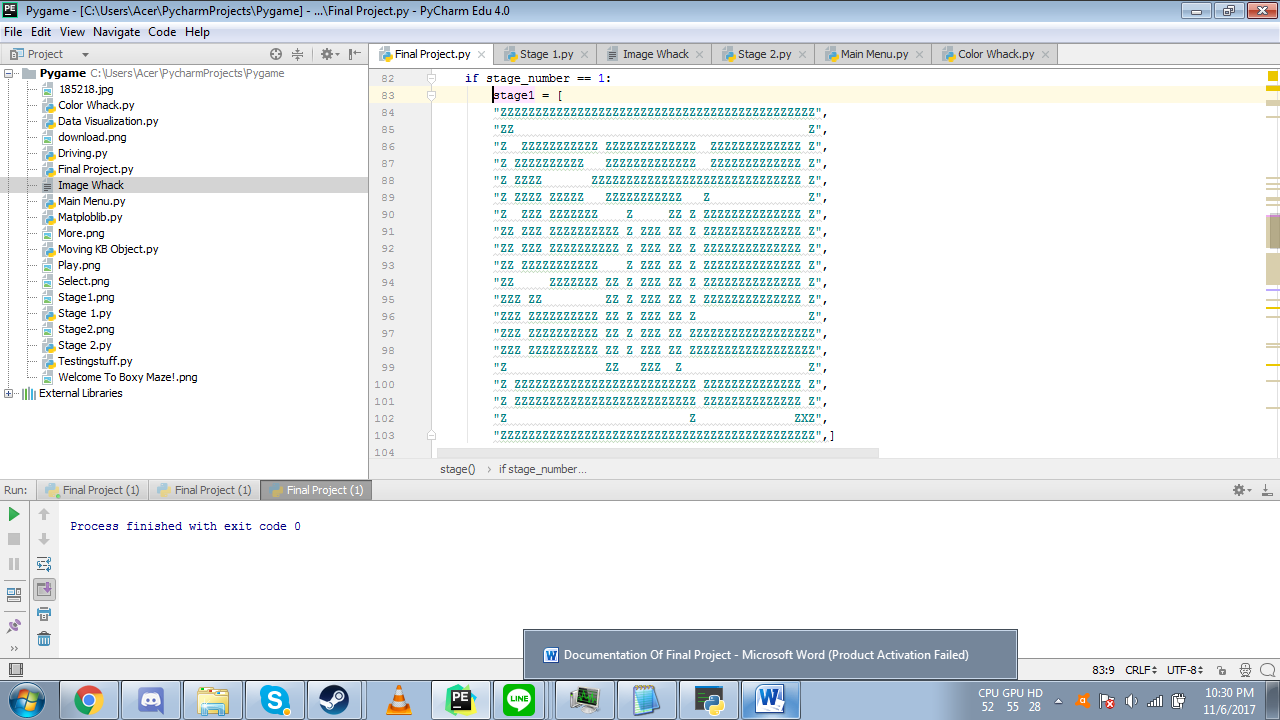
The Objects

**class** Block(object):*#The designated finish point* **def** \_\_init\_\_(self, pos):  
 blocks.append(self)  
 self.rect= pygame.Rect(pos[0], pos[1], 16, 16)

  
  
**class** Wall(object):*#The walls* **def** \_\_init\_\_(self, pos):  
 hitboxes.append(self)  
 self.rect = pygame.Rect(pos[0], pos[1], 16, 16) The code that sets the title and size of the window

pygame.display.set\_caption(**"Boxy Maze"**)  
screen = pygame.display.set\_mode((720, 320))

The Pattern for the actual stage



This code is how the maze will actually work

a = b = 0  
**if** stage\_number == 1:  
 **for** row **in** stage1:  
 **for** col **in** row:  
 **if** col == **"Z"**:  
 Wall((a, b))  
 **if** col == **"X"**:  
 end\_rect = pygame.Rect(a, b, 16, 16)  
 Block((a,b))  
 a += 16  
 b += 16  
 a = 0

**The code to exit the game**

**while** running:

**for** e **in** pygame.event.get():  
 **if** e.type == pygame.QUIT:  
 pygame.quit()  
 **break  
 if** e.type == pygame.KEYDOWN **and** e.key == pygame.K\_ESCAPE:  
 pygame.quit()  
 **break**

This is the code that lets you control the player

key = pygame.key.get\_pressed()  
**if** key[pygame.K\_LEFT]:  
 player.move(-2, 0)*#These are the speed,i put the speed on 2 because of the tight angles in stage 2***if** key[pygame.K\_RIGHT]:  
 player.move(2, 0)  
**if** key[pygame.K\_UP]:  
 player.move(0, -2)  
**if** key[pygame.K\_DOWN]:  
 player.move(0, 2)

This is how the game will be executed

screen.fill((0, 0, 0))  
**for** hitbox **in** hitboxes:  
 pygame.draw.rect(screen, (255, 255, 255), hitbox.rect)  
pygame.draw.rect(screen, (255, 0, 0), end\_rect)  
pygame.draw.rect(screen, (255, 200, 0), player.rect)  
pygame.display.flip()  
display.update()

The Actual Code:

**import** os  
**import** random  
**import** pygame  
**from** pygame.sprite **import** \*  
**from** pygame **import** \*  
  
pygame.init()  
  
hitboxes = []  
blocks =[]  
**class** textsprite (Sprite): *#This code is reusable, just fill in the parameters to reuse* **def** \_\_init\_\_(self, fontstyle, text, fontsize, horizontal, vertical, R, G, B):  
 Sprite.\_\_init\_\_(self)  
 self.font = pygame.font.SysFont(fontstyle, fontsize)  
 self.image = self.font.render(text, **False**, (R, G, B))  
 self.rect = self.image.get\_rect()  
 self.rect.x = horizontal  
 self.rect.y = vertical  
  
**class** Player(object):*#Basically the orange box that is moving* **def** \_\_init\_\_(self):  
 self.rect = pygame.Rect(32, 32, 16, 16)  
  
 **def** move(self, c, d):  
 **if** c != 0:  
 self.move\_single\_axis(c, 0)  
 **if** d != 0:  
 self.move\_single\_axis(0, d)  
  
 **def** move\_single\_axis(self, c, d):  
  
 self.rect.x += c  
 self.rect.y += d  
  
 **for** hitbox **in** hitboxes:  
 **if** self.rect.colliderect(hitbox.rect):  
 **if** c > 0:  
 self.rect.right = hitbox.rect.left  
 **if** c < 0:  
 self.rect.left = hitbox.rect.right  
 **if** d > 0:  
 self.rect.bottom = hitbox.rect.top  
 **if** d < 0:  
 self.rect.top = hitbox.rect.bottom  
  
 **for** block **in** blocks:  
 **if** self.rect.colliderect(block.rect):  
 **if** c > 0:  
 self.rect.right = block.rect.left  
 **if** c < 0:  
 self.rect.left = block.rect.right  
 **if** d > 0:  
 self.rect.bottom = block.rect.top  
 **if** d < 0:  
 self.rect.top = block.rect.bottom  
  
**class** Block(object):*#The designated finish point* **def** \_\_init\_\_(self, pos):  
 blocks.append(self)  
 self.rect= pygame.Rect(pos[0], pos[1], 16, 16)  
  
  
**class** Wall(object):*#The walls* **def** \_\_init\_\_(self, pos):  
 hitboxes.append(self)  
 self.rect = pygame.Rect(pos[0], pos[1], 16, 16)  
  
**def** stage(stage\_number):  
 os.environ[**"SDL\_VIDEO\_CENTERED"**] = **"1"***#I have no idea since i got this from the video* pygame.display.set\_caption(**"Boxy Maze"**)  
 screen = pygame.display.set\_mode((720, 320))  
  
 clock = pygame.time.Clock()  
  
 player = Player()  
 **if** stage\_number == 1:  
 stage1 = [  
 **"ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ"**,  
 **"ZZ Z"**,  
 **"Z ZZZZZZZZZZZ ZZZZZZZZZZZZZ ZZZZZZZZZZZZZ Z"**,  
 **"Z ZZZZZZZZZZ ZZZZZZZZZZZZZ ZZZZZZZZZZZZZ Z"**,  
 **"Z ZZZZ ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ Z"**,  
 **"Z ZZZZ ZZZZZ ZZZZZZZZZZZ Z Z"**,  
 **"Z ZZZ ZZZZZZZ Z ZZ Z ZZZZZZZZZZZZZZ Z"**,  
 **"ZZ ZZZ ZZZZZZZZZZ Z ZZZ ZZ Z ZZZZZZZZZZZZZZ Z"**,  
 **"ZZ ZZZ ZZZZZZZZZZ Z ZZZ ZZ Z ZZZZZZZZZZZZZZ Z"**,  
 **"ZZ ZZZZZZZZZZZ Z ZZZ ZZ Z ZZZZZZZZZZZZZZ Z"**,  
 **"ZZ ZZZZZZZ ZZ Z ZZZ ZZ Z ZZZZZZZZZZZZZZ Z"**,  
 **"ZZZ ZZ ZZ Z ZZZ ZZ Z ZZZZZZZZZZZZZZ Z"**,  
 **"ZZZ ZZZZZZZZZZ ZZ Z ZZZ ZZ Z Z"**,  
 **"ZZZ ZZZZZZZZZZ ZZ Z ZZZ ZZ ZZZZZZZZZZZZZZZZZZ"**,  
 **"ZZZ ZZZZZZZZZZ ZZ Z ZZZ ZZ ZZZZZZZZZZZZZZZZZZ"**,  
 **"Z ZZ ZZZ Z Z"**,  
 **"Z ZZZZZZZZZZZZZZZZZZZZZZZZZZ ZZZZZZZZZZZZZZ Z"**,  
 **"Z ZZZZZZZZZZZZZZZZZZZZZZZZZZ ZZZZZZZZZZZZZZ Z"**,  
 **"Z Z ZXZ"**,  
 **"ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ"**,]  
  
 **if** stage\_number == 2:  
 stage2 = [  
 **"ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ"**,  
 **"ZZ ZZ"**,  
 **"ZZ ZZZZZZZZZZZZZZZZZZ ZZZZZZZZZZZZZZZZZZ ZZ"**,  
 **"ZZZZZ Z Z ZZZZZ"**,  
 **"ZZZZZ ZZZZZZZZZZZZZZ Z Z ZZZZZZZZZZZZZZ ZZZZZ"**,  
 **"Z Z Z Z Z Z"**,  
 **"Z ZZZZZ Z ZZZZZZZZZZZZ ZZZZZZZZZZ Z ZZZZZZZ Z"**,  
 **"Z Z Z Z Z Z Z Z Z Z"**,  
 **"Z Z Z Z Z ZZZZZZZZZZZZZZZZZZZ Z Z Z Z Z"**,  
 **"Z Z Z Z Z Z Z Z Z Z Z Z Z"**,  
 **"Z Z Z Z Z Z ZZZZZZZZ ZZZZZZ Z Z Z Z Z Z"**,  
 **"Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z"**,  
 **"Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z"**,  
 **"Z Z Z Z Z Z ZZZ Z Z Z Z Z Z"**,  
 **"Z Z Z ZZZ Z Z Z ZZZ Z Z Z"**,  
 **"Z Z ZZZ ZZZ Z Z ZZZ ZZZ Z Z"**,  
 **"Z Z ZZZZZZZ Z Z ZZZZZZZ Z Z"**,  
 **"Z ZZZZZZZZZZZZZZZZZZZZ ZZZZZZZZZZZZZZZZZZZZ Z"**,  
 **"Z Z Z Z"**,  
 **"ZZZZZZZZZZZZZZZZZZZZZZXZZZZZZZZZZZZZZZZZZZZZZ"**,]  
  
 a = b = 0  
 **if** stage\_number == 1:  
 **for** row **in** stage1:  
 **for** col **in** row:  
 **if** col == **"Z"**:  
 Wall((a, b))  
 **if** col == **"X"**:  
 end\_rect = pygame.Rect(a, b, 16, 16)  
 Block((a,b))  
 a += 16  
 b += 16  
 a = 0  
 **if** stage\_number == 2:  
 **for** row **in** stage2:  
 **for** col **in** row:  
 **if** col == **"Z"**:  
 Wall((a, b))  
 **if** col == **"X"**:  
 end\_rect = pygame.Rect(a, b, 16, 16)  
 Block((a,b))  
 a += 16  
 b += 16  
 a = 0  
 running = **True  
 while** running:  
  
 clock.tick(60)  
  
 **for** e **in** pygame.event.get():  
 **if** e.type == pygame.QUIT:  
 pygame.quit()  
 **break  
 if** e.type == pygame.KEYDOWN **and** e.key == pygame.K\_ESCAPE:  
 pygame.quit()  
 **break** key = pygame.key.get\_pressed()  
 **if** key[pygame.K\_LEFT]:  
 player.move(-2, 0)*#These are the speed,i put the speed on 2 because of the tight angles in stage 2* **if** key[pygame.K\_RIGHT]:  
 player.move(2, 0)  
 **if** key[pygame.K\_UP]:  
 player.move(0, -2)  
 **if** key[pygame.K\_DOWN]:  
 player.move(0, 2)  
  
 screen.fill((0, 0, 0))  
 **for** hitbox **in** hitboxes:  
 pygame.draw.rect(screen, (255, 255, 255), hitbox.rect)  
 pygame.draw.rect(screen, (255, 0, 0), end\_rect)  
 pygame.draw.rect(screen, (255, 200, 0), player.rect)  
 pygame.display.flip()  
 display.update()  
  
  
**def** menu():  
 pygame.display.set\_caption(**"Boxy Maze"**)  
 screen = pygame.display.set\_mode((720, 320))  
 screen.fill((0, 0, 0))  
 play = textsprite(**"Times New Roman"**, **"Play"**, 60, 100, 100, 255, 255, 255)*#For the integers(size,placement(x(The higher the input it basically goes right),y(The higher the input the lower the placement),Colour(R,G,B)* stages = textsprite(**"Times New Roman"**, **"Stages"**, 60, 100, 200, 255, 255, 255)  
 **while True**:  
 all\_text = Group(play, stages)  
 all\_text.draw(screen)  
 menu\_wait = event.wait()  
 **if** play.rect.collidepoint(mouse.get\_pos()):  
 **if** menu\_wait.type == MOUSEBUTTONDOWN:  
 stage(1)  
 **if** stages.rect.collidepoint(mouse.get\_pos()):  
 **if** menu\_wait.type == MOUSEBUTTONDOWN:  
 stage\_selection()  
 **if** menu\_wait.type == QUIT:  
 pygame.quit()  
 **break** display.update()  
  
**def** stage\_selection():  
 pygame.display.set\_caption(**"Boxy Maze"**)  
 screen = pygame.display.set\_mode((720, 320))  
 screen.fill((0, 0, 0))  
 stage1 = textsprite(**"Arial"**, **"Stage 1"**, 45, 50, 50, 255, 255, 255)  
 stage2 = textsprite(**"Arial"**, **"Stage 2"**, 45, 50, 150, 255, 255, 255)  
 back = textsprite(**"Arial"**, **"Back"**, 45, 50, 250, 255, 255, 255)  
 **while True**:  
 all\_text = Group(stage1, stage2, back)  
 all\_text.draw(screen)  
 menu\_wait = event.wait()  
 **if** stage1.rect.collidepoint(mouse.get\_pos()):  
 **if** menu\_wait.type == MOUSEBUTTONDOWN:  
 stage(1)  
 **if** stage2.rect.collidepoint(mouse.get\_pos()):  
 **if** menu\_wait.type == MOUSEBUTTONDOWN:  
 stage(2)  
 **if** back.rect.collidepoint(mouse.get\_pos()):  
 **if** menu\_wait.type == MOUSEBUTTONDOWN:  
 menu()  
 **if** menu\_wait.type == QUIT:  
 pygame.quit()  
 **break** display.update()  
  
menu()

People Who Helped Me: Pier and Adrian

I got most of the code from here: https://www.youtube.com/channel/UC2vm-0XX5RkWCXWwtBZGOXg