- **Step 1: Create a project name.**

  First when you finished installed the **Pycharm IDE** in your computer, open it and then create a "**project name**" after creating a project name click the "**create**" button.

- **Step 2: Create a python file.**

  Second after creating a project name, "**right click**" your project name and then click

  "**new**" after that click the "**python file**".

- **Step 3: Name your python file.**

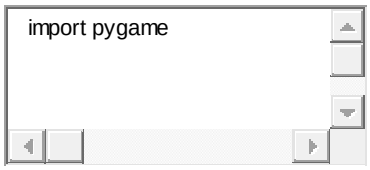
  Third after creating a python file, Name your python file after that click "**enter**".

- **Step 4: The actual code.**

  This is the actual coding on how to create **Tank Game Python**, and you are free to copy this code and download the full source code given below.

Importing Pygame Module

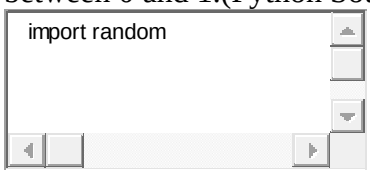
In the code below, which is pygame library is an open-source module for the Python programming language specifically intended to help you make games and other multimedia applications. Pygame can run across many platforms and operating systems.(Python Source Code for Tank Game)




```
1 import pygame
```

Importing Random Module

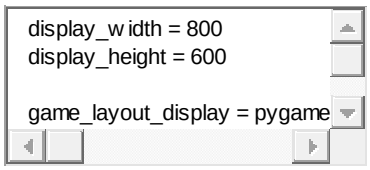
In the code below. which is for the **random**() function, which generates **random** numbers between 0 and 1.(Python Source Code for Tank Game)




```
1 import random
```

This module is for the design main window

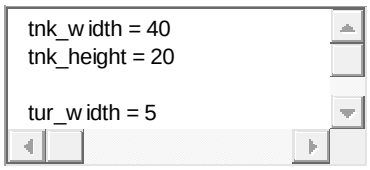
In the code given below, which is for the design of main screen window colors, title of the game and the background image.




```
1  display_width = 800
2  display_height = 600
3
4  game_layout_display = pygame.display.set_mode((display_width, display_height))
5  pygame.display.set_caption('Tanks Game - Brought To You By Itsourcecode.com')
6
7  Resources = pygame.image.load("resources/game_background.png")
8  pygame.display.set_icon(Resources)
9
10 # colors
11 wheat = (245, 222, 179)
12
13 white = (255, 255, 255)
14 black = (0, 0, 0)
15 blue = (0, 0, 255)
16
17 red = (200, 0, 0)
18 light_red = (255, 0, 0)
19
20 yellow = (200, 200, 0)
21 light_yellow = (255, 255, 0)
22
23 green = (34, 177, 76)
24 light_green = (0, 255, 0)
```

This module is for the geometry of tank

In the code given below, which is for the geometry of tank width and height.

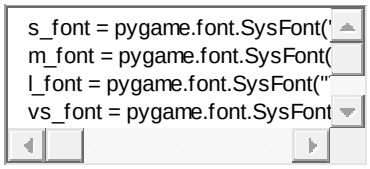



```
1 tnk_width = 40
2 tnk_height = 20
3
4 tur_width = 5
```

```
5 whl_width = 5
```

This module is for the font size

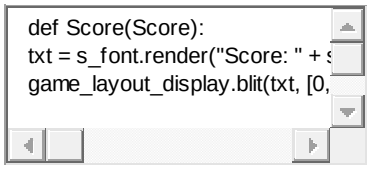
In the code given below, which is for the font size use such as small, medium, large, and very small.




```
1 s_font = pygame.font.SysFont("Times New Roman", 25)
2 m_font = pygame.font.SysFont("Times New Roman", 50)
3 l_font = pygame.font.SysFont("Times New Roman", 85)
4 vs_font = pygame.font.SysFont("Times New Roman", 25)
```

This module is for the score

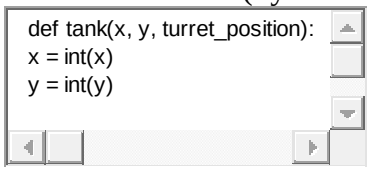
In the code given below, which is for the defining score function.




```
1 def Score(Score):
2 txt = s_font.render("Score: " + str(Score), True, white)
3 game_layout_display.blit(txt, [0, 0])
```

This module is for the player tank

In the code given below, which is for the function of player tank, defining turrets position and wheels dimension.(Python Source Code for Tank Game)

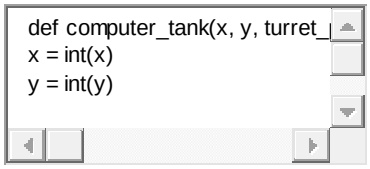



```
1   def tank(x, y, turret_position):
2   x = int(x)
3   y = int(y)
4
5   pos_Turrets = [(x - 27, y - 2),
6   (x - 26, y - 5),
7   (x - 25, y - 8),
8   (x - 23, y - 12),
```

```
9   (x - 20, y - 14),
10 (x - 18, y - 15),
11 (x - 15, y - 17),
12 (x - 13, y - 19),
13 (x - 11, y - 21)
14 ]
15
16 pygame.draw.circle(game_layout_display, blue, (x, y), int(tnk_height / 2))
17 pygame.draw.rect(game_layout_display, blue, (x - tnk_height, y, tnk_width, tnk_height))
18
19 pygame.draw.line(game_layout_display, blue, (x, y), pos_Turrets[turret_position],
20 tur_width)
21
22 pygame.draw.circle(game_layout_display, blue, (x - 15, y + 20), whl_width)
23 pygame.draw.circle(game_layout_display, blue, (x - 10, y + 20), whl_width)
24
25 pygame.draw.circle(game_layout_display, blue, (x - 15, y + 20), whl_width)
26 pygame.draw.circle(game_layout_display, blue, (x - 10, y + 20), whl_width)
27 pygame.draw.circle(game_layout_display, blue, (x - 5, y + 20), whl_width)
28 pygame.draw.circle(game_layout_display, blue, (x, y + 20), whl_width)
29 pygame.draw.circle(game_layout_display, blue, (x + 5, y + 20), whl_width)
30 pygame.draw.circle(game_layout_display, blue, (x + 10, y + 20), whl_width)
31 pygame.draw.circle(game_layout_display, blue, (x + 15, y + 20), whl_width)
32
    return pos_Turrets[turret_position]
```

This module is for the computer tank

In the code given below, which is for the function for computer tank , defining turrets position and wheel dimension.

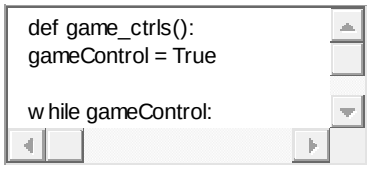



```
1   def computer_tank(x, y, turret_position):
2   x = int(x)
3   y = int(y)
4
5   pos_Turrets = [(x + 27, y - 2),
6   (x + 26, y - 5),
7   (x + 25, y - 8),
8   (x + 23, y - 12),
9   (x + 20, y - 14),
10 (x + 18, y - 15),
11 (x + 15, y - 17),
12 (x + 13, y - 19),
13 (x + 11, y - 21)
```

```
14 ]
15
16 pygame.draw.circle(game_layout_display, blue, (x, y), int(tnk_height / 2))
17 pygame.draw.rect(game_layout_display, blue, (x - tnk_height, y, tnk_width, tnk_height))
18
19 pygame.draw.line(game_layout_display, blue, (x, y), pos_Turrets[turret_position],
20 tur_width)
21
22 pygame.draw.circle(game_layout_display, blue, (x - 15, y + 20), whl_width)
23 pygame.draw.circle(game_layout_display, blue, (x - 10, y + 20), whl_width)
24
25 pygame.draw.circle(game_layout_display, blue, (x - 15, y + 20), whl_width)
26 pygame.draw.circle(game_layout_display, blue, (x - 10, y + 20), whl_width)
27 pygame.draw.circle(game_layout_display, blue, (x - 5, y + 20), whl_width)
28 pygame.draw.circle(game_layout_display, blue, (x, y + 20), whl_width)
29 pygame.draw.circle(game_layout_display, blue, (x + 5, y + 20), whl_width)
30 pygame.draw.circle(game_layout_display, blue, (x + 10, y + 20), whl_width)
31 pygame.draw.circle(game_layout_display, blue, (x + 15, y + 20), whl_width)
32
    return pos_Turrets[turret_position]
```

This module is for the game control

In the code given below, which is for the function of game control screen.

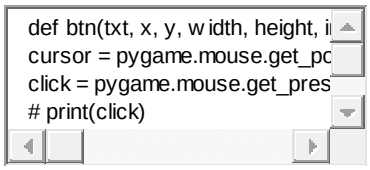



```
1   def game_ctrls():
2   gameControl = True
3
4   while gameControl:
5   for event in pygame.event.get():
6   # print(event)
7   if event.type == pygame.QUIT:
8   pygame.quit()
9   quit()
10
11 game_layout_display.fill(black)
12 msg_screen("Controls", white, -100, size="large")
13 msg_screen("Fire: Spacebar", wheat, -30)
14 msg_screen("Move Turret: Up and Down arrows", wheat, 10)
15 msg_screen("Move Tank: Left and Right arrows", wheat, 50)
16 msg_screen("Press D to raise Power % AND Press A to lower Power % ", wheat, 140)
17 msg_screen("Pause: P", wheat, 90)
18
19 btn("Play", 150, 500, 100, 50, green, light_green, action="play")
```

```
20 btn("Main", 350, 500, 100, 50, yellow, light_yellow, action="main")
21 btn("Quit", 550, 500, 100, 50, red, light_red, action="quit")
22
23 pygame.display.update()
24
25 clock.tick(15)
```

This module is for the buttons

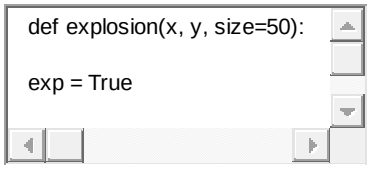
In the code given below, which is for the function of buttons having action calls and text.




```
1   def btn(txt, x, y, width, height, inactive_color, active_color, action=None,size=" "):
2   cursor = pygame.mouse.get_pos()
3   click = pygame.mouse.get_pressed()
4   # print(click)
5   if x + width &gt; cursor[0] &gt; x and y + height &gt; cursor[1] &gt; y:
6   pygame.draw.rect(game_layout_display, active_color, (x, y, width, height))
7   if click[0] == 1 and action != None:
8   if action == "quit":
9   pygame.quit()
10 quit()
11
12 if action == "controls":
13 game_ctrls()
14
15 if action == "play":
16 gameLoop()
17
18 if action == "main":
19 game_intro()
20
21 else:
22 pygame.draw.rect(game_layout_display, inactive_color, (x, y, width, height))
23
24 txt_btn(txt, black, x, y, width, height)
```

This module is for the explosion of tank

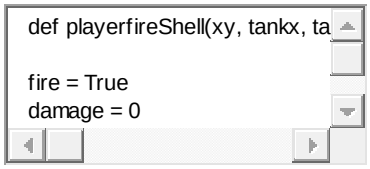
In the code given below, which is for the function for explosion of both tanks.




```
def explosion(x, y, size=50):

exp = True

while exp:
for event in pygame.event.get():
if event.type == pygame.QUIT:
pygame.quit()
quit()

startPoint = x, y

choice_colors = [red, light_red, yellow, light_yellow]

mgntde = 1

while mgntde &lt; size:
exploding_bit_x = x + random.randrange(-1 * mgntde, mgntde)
exploding_bit_y = y + random.randrange(-1 * mgntde, mgntde)

pygame.draw.circle(game_layout_display, choice_colors[random.randrange(0, 4)],
(exploding_bit_x, exploding_bit_y),
random.randrange(1, 5))
mgntde += 1

pygame.display.update()
clock.tick(100)

exp = False
```

This module is for the firing for player tank

In the code given below, which is for the function of firing for player tank.



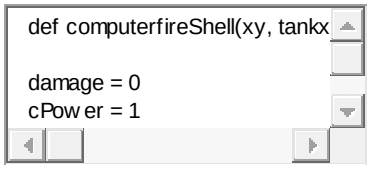

```
def playerfireShell(xy, tankx, tanky, turPost, gun_power, xloc, bar_width, ranHeight,
eTankX, eTankY):

fire = True
damage = 0
```

```
6
7   startShell = list(xy)
8
9   print("FIRE!", xy)
10
11 while fire:
12 for event in pygame.event.get():
13 if event.type == pygame.QUIT:
14 pygame.quit()
15 quit()
16
17 # print(startingShell[0],startingShell[1])
18 pygame.draw.circle(game_layout_display, red, (startShell[0], startShell[1]), 5)
19
20 startShell[0] -= (12 - turPost) * 2
21
22 # y = x**2
23 startShell[1] += int(
24 (((startShell[0] - xy[0]) * 0.015 / (gun_power / 50)) ** 2) - (turPost + turPost / (12 -
25 turPost)))
26
27 if startShell[1] &gt; display_height - grnd_height:
28 print("Last shell:", startShell[0], startShell[1])
29 hit_x = int((startShell[0] * display_height - grnd_height) / startShell[1])
30 hit_y = int(display_height - grnd_height)
31 print("Impact:", hit_x, hit_y)
32
33 if eTankX + 10 &gt; hit_x &gt; eTankX - 10:
34 print("Critical Hit!")
35 damage = 25
36 elif eTankX + 15 &gt; hit_x &gt; eTankX - 15:
37 print("Hard Hit!")
38 damage = 18
39 elif eTankX + 25 &gt; hit_x &gt; eTankX - 25:
40 print("Medium Hit")
41 damage = 10
42 elif eTankX + 35 &gt; hit_x &gt; eTankX - 35:
43 print("Light Hit")
44 damage = 5
45
46 explosion(hit_x, hit_y)
47 fire = False
48
49 check_x_1 = startShell[0] &lt;= xloc + bar_width
50 check_x_2 = startShell[0] &gt;= xloc
51
52 check_y_1 = startShell[1] &lt;= display_height
53 check_y_2 = startShell[1] &gt;= display_height - ranHeight
54
55 if check_x_1 and check_x_2 and check_y_1 and check_y_2:
```

```
56 print("Last shell:", startShell[0], startShell[1])
57 hit_x = int((startShell[0]))
58 hit_y = int(startShell[1])
59 print("Impact:", hit_x, hit_y)
60 explosion(hit_x, hit_y)
61 fire = False
62
63 pygame.display.update()
    clock.tick(60)
    return damage
```

This module is for the firing for computer tank

In the code given below, which is for the function of firing for computer tank.




```
1   def computerfireShell(xy, tankx, tanky, turPost, gun_power, xloc, bar_width, ranHeight,
2   ptankx, ptanky):
3
4   damage = 0
5   cPower = 1
6   pow_found = False
7
8   while not pow_found:
9   cPower += 1
10  if cPower &gt; 100:
11  pow_found = True
12  # print(currentPower)
13
14  fire = True
15  startShell = list(xy)
16
17  while fire:
18  for event in pygame.event.get():
19  if event.type == pygame.QUIT:
20  pygame.quit()
21  quit()
22
23  # pygame.draw.circle(gameDisplay, red, (startingShell[0],startingShell[1]),5)
24
25  startShell[0] += (12 - turPost) * 2
26  startShell[1] += int(
27  (((startShell[0] - xy[0]) * 0.015 / (cPower / 50)) ** 2) - (turPost + turPost / (12 -
28  turPost)))
29
```

```
30  if startShell[1] &gt; display_height - grnd_height:
31  hit_x = int((startShell[0] * display_height - grnd_height) / startShell[1])
32  hit_y = int(display_height - grnd_height)
33  # explosion(hit_x,hit_y)
34  if ptankx + 15 &gt; hit_x &gt; ptankx - 15:
35  print("target acquired!")
36  power_found = True
37  fire = False
38
39  check_x_1 = startShell[0] &lt;= xloc + bar_width
40  check_x_2 = startShell[0] &gt;= xloc
41
42  check_y_1 = startShell[1] &lt;= display_height
43  check_y_2 = startShell[1] &gt;= display_height - ranHeight
44
45  if check_x_1 and check_x_2 and check_y_1 and check_y_2:
46  hit_x = int((startShell[0]))
47  hit_y = int(startShell[1])
48  # explosion(hit_x,hit_y)
49  fire = False
50
51  fire = True
52  startShell = list(xy)
53  print("FIRE!", xy)
54
55  while fire:
56  for event in pygame.event.get():
57  if event.type == pygame.QUIT:
58  pygame.quit()
59  quit()
60
61  pygame.draw.circle(game_layout_display, red, (startShell[0], startShell[1]), 5)
62
63  startShell[0] += (12 - turPost) * 2
64
65
66
67  gun_power = random.randrange(int(cPower * 0.90), int(cPower * 1.10))
68
69  startShell[1] += int(
70  (((startShell[0] - xy[0]) * 0.015 / (gun_power / 50)) ** 2) - (turPost + turPost / (12 -
71  turPost)))
72
73  if startShell[1] &gt; display_height - grnd_height:
74  print("last shell:", startShell[0], startShell[1])
75  hit_x = int((startShell[0] * display_height - grnd_height) / startShell[1])
76  hit_y = int(display_height - grnd_height)
77  print("Impact:", hit_x, hit_y)
78
79  if ptankx + 10 &gt; hit_x &gt; ptankx - 10:
```

```
80  print("Critical Hit!")
81  damage = 25
82  elif ptankx + 15 &gt; hit_x &gt; ptankx - 15:
83  print("Hard Hit!")
84  damage = 18
85  elif ptankx + 25 &gt; hit_x &gt; ptankx - 25:
86  print("Medium Hit")
87  damage = 10
88  elif ptankx + 35 &gt; hit_x &gt; ptankx - 35:
89  print("Light Hit")
90  damage = 5
91
92  explosion(hit_x, hit_y)
93  fire = False
94
95  check_x_1 = startShell[0] &lt;= xloc+ bar_width
96  check_x_2 = startShell[0] &gt;= xloc
97
98  check_y_1 = startShell[1] &lt;= display_height
99  check_y_2 = startShell[1] &gt;= display_height - ranHeight
100
101 if check_x_1 and check_x_2 and check_y_1 and check_y_2:
102 print("Last shell:", startShell[0], startShell[1])
103 hit_x = int((startShell[0]))
104 hit_y = int(startShell[1])
105 print("Impact:", hit_x, hit_y)
106 explosion(hit_x, hit_y)
107 fire = False
108
    pygame.display.update()
    clock.tick(60)
    return damage
```

This module is for the game over screen

In the code given below, which is for the function of game over screen.

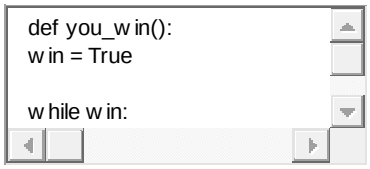
```
def game_over():
game_over = True

w hile game_over:
```

```
1  def game_over():
2  game_over = True
3
4  while game_over:
5  for event in pygame.event.get():
6  # print(event)
7  if event.type == pygame.QUIT:
```

```
8   pygame.quit()
9   quit()
10
11 game_layout_display.fill(black)
12 msg_screen("Game Over", white, -100, size="large")
13 msg_screen("You died.", wheat, -30)
14
15 btn("Play Again", 150, 500, 150, 50, wheat, light_green, action="play")
16 btn("Controls", 350, 500, 100, 50, wheat, light_yellow, action="controls")
17 btn("Quit", 550, 500, 100, 50, wheat, light_red, action="quit")
18
19 pygame.display.update()
20
21 clock.tick(15)
```

This module is for the player winner screen

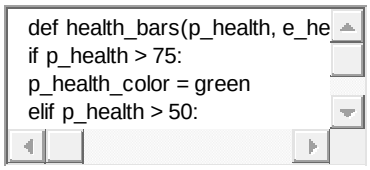
In the code given below, which is for the function of player winner screen




```
1   def you_win():
2   win = True
3
4   while win:
5   for event in pygame.event.get():
6   # print(event)
7   if event.type == pygame.QUIT:
8   pygame.quit()
9   quit()
10
11 game_layout_display.fill(black)
12 msg_screen("You won!", white, -100, size="large")
13 msg_screen("Congratulations!", wheat, -30)
14
15 btn("play Again", 150, 500, 150, 50, wheat, light_green, action="play")
16 btn("controls", 350, 500, 100, 50, wheat, light_yellow, action="controls")
17 btn("quit", 550, 500, 100, 50, wheat, light_red, action="quit")
18
19 pygame.display.update()
20
21 clock.tick(15)
```

This module is for the health bars

In the code given below ,which is for the function for health bars of player tank and computer tank.




```
1  def health_bars(p_health, e_health):
2  if p_health &gt; 75:
3  p_health_color = green
4  elif p_health &gt; 50:
5  p_health_color = yellow
6  else:
7  p_health_color = red
8
9  if e_health &gt; 75:
10 e_health_color = green
11 elif e_health &gt; 50:
12 e_health_color = yellow
13 else:
14 e_health_color = red
15
16 pygame.draw.rect(game_layout_display, p_health_color, (680, 25, p_health, 25))
17 pygame.draw.rect(game_layout_display, e_health_color, (20, 25, e_health, 25))
```

Complete Source Code of Tank Game Python


```
import pygame<br>import random<br><br>pygame.init()<br><br>display_width = 800<br>display_height = 600<br><br>game_layout_display = pygame.display.set_mode((display_width, display_height))<br>pygame.display.set_caption('Tanks Game - Brought To You By Itsourcecode.com')<br><br>Resources = pygame.image.load("resources/game_background.png")<br>pygame.display.set_icon(Resources)<br><br># colors<br>wheat = (245, 222, 179)<br><br>white = (255, 255, 255)<br>black = (0, 0, 0)<br>blue = (0, 0, 255)<br><br>red = (200, 0, 0)<br>light_red = (255, 0, 0)<br><br>yellow = (200, 200, 0)<br>light_yellow = (255, 255, 0)<br><br>green = (34, 177, 76)<br>light_green = (0, 255, 0)<br><br># for picking current time for the frames per second<br>clock = pygame.time.Clock()<br># geometry of tank and its turret<br>tnk_width = 40<br>tnk_height = 20<br><br>tur_width = 5<br>whl_width = 5<br><br>grnd_height = 35<br># fonts with size, for text_object function<br>s_font = pygame.font.SysFont("Times New Roman", 25)<br>m_font = pygame.font.SysFont("Times New Roman", 50)<br>l_font = pygame.font.SysFont("Times New Roman", 85)<br>vs_font = pygame.font.SysFont("Times New Roman",
```

25)

# defining score function
def Score(Score):
  txt = s_font.render("Score: " + str(Score), True, white)
  game_layout_display.blit(txt, [0, 0])

# defining function to get the fonts and sizes assigned with them by size names by default size="small"
def txt_object(txt, color, size="small"):
  if size == "small":
  txtSrfc = s_font.render(txt, True, color)
  if size == "medium":
  txtSrfc = m_font.render(txt, True, color)
  if size == "large":
  txtSrfc = l_font.render(txt, True, color)
  if size == "vsmall":
  txtSrfc = vs_font.render(txt, True, color)

  return txtSrfc, txtSrfc.get_rect()

# function for texts that has to appear over button
def txt_btn(message, color, btnx, btny, btnwidth, btnheight, size="vsmall"):
  txtSrf, textRect = txt_object(message, color, size)
  textRect.center = ((btnx + (btnwidth / 2)), btny + (btnheight / 2))
  game_layout_display.blit(txtSrf, textRect)

# function for texts that has to appear over screen
def msg_screen(message, color, y_displace=0, size="small"):
  txtSrf, textRect = txt_object(message, color, size)
  textRect.center = (int(display_width / 2), int(display_height / 2) + y_displace)
  game_layout_display.blit(txtSrf, textRect)

# function for players tank , defining turrets positions and wheels dimensions
def tank(x, y, turret_position):
  x = int(x)
  y = int(y)

  pos_Turrets = [(x - 27, y - 2),
  (x - 26, y - 5),
  (x - 25, y - 8),
  (x - 23, y - 12),
  (x - 20, y - 14),
  (x - 18, y - 15),
  (x - 15, y - 17),
  (x - 13, y - 19),
  (x - 11, y - 21)
  ]

  pygame.draw.circle(game_layout_display, blue, (x, y), int(tnk_height / 2))
  pygame.draw.rect(game_layout_display, blue, (x - tnk_height, y, tnk_width, tnk_height))

  pygame.draw.line(game_layout_display, blue, (x, y), pos_Turrets[turret_position], tur_width)

  pygame.draw.circle(game_layout_display, blue, (x - 15, y + 20), whl_width)
  pygame.draw.circle(game_layout_display, blue, (x - 10, y + 20), whl_width)

  pygame.draw.circle(game_layout_display, blue, (x - 15, y + 20), whl_width)
  pygame.draw.circle(game_layout_display, blue, (x - 10, y + 20), whl_width)
  pygame.draw.circle(game_layout_display, blue, (x - 5, y + 20), whl_width)
  pygame.draw.circle(game_layout_display, blue, (x, y + 20), whl_width)
  pygame.draw.circle(game_layout_display, blue, (x + 5, y + 20), whl_width)
  pygame.draw.circle(game_layout_display, blue, (x + 10, y + 20), whl_width)
  pygame.draw.circle(game_layout_display, blue, (x + 15, y + 20), whl_width)

  return pos_Turrets[turret_position]

# function for computers tank , defining turrets positions and wheels dimensions
def computer_tank(x, y, turret_position):
  x = int(x)
  y = int(y)

  pos_Turrets = [(x + 27, y - 2),
  (x + 26, y - 5),
  (x + 25, y - 8),
  (x + 23, y - 12),
  (x + 20, y - 14),
  (x + 18, y - 15),
  (x + 15, y - 17),
  (x + 13, y - 19),
  (x + 11, y - 21)
  ]

  pygame.draw.circle(game_layout_display, blue, (x, y), int(tnk_height / 2))
  pygame.draw.rect(game_layout_display, blue, (x - tnk_height, y, tnk_width, tnk_height))

  pygame.draw.line(game_layout_display, blue, (x, y), pos_Turrets[turret_position], tur_width)

  pygame.draw.circle(game_layout_display, blue, (x - 15, y + 20), whl_width)
  pygame.draw.circle(game_layout_display, blue, (x - 10, y + 20), whl_width)

  pygame.draw.circle(game_layout_display, blue, (x - 15, y + 20),

whl_width)
    pygame.draw.circle(game_layout_display, blue, (x - 10, y + 20),
whl_width)
    pygame.draw.circle(game_layout_display, blue, (x - 5, y + 20),
whl_width)
    pygame.draw.circle(game_layout_display, blue, (x, y + 20),
whl_width)
    pygame.draw.circle(game_layout_display, blue, (x + 5, y + 20),
whl_width)
    pygame.draw.circle(game_layout_display, blue, (x + 10, y + 20),
whl_width)
    pygame.draw.circle(game_layout_display, blue, (x + 15, y + 20),
whl_width)

    return pos_Turrets[turret_position]

Game control
Screen
def game_ctrls():
    gameControl = True

    while
gameControl:
        for event in pygame.event.get():
            #
print(event)
            if event.type ==
pygame.QUIT:
                pygame.quit()
                quit()

        game_layou
t_display.fill(black)
        msg_screen("Controls", white, -100,
size="large")
        msg_screen("Fire: Spacebar", wheat, -
30)
        msg_screen("Move Turret: Up and Down arrows", wheat,
10)
        msg_screen("Move Tank: Left and Right arrows", wheat,
50)
        msg_screen("Press D to raise Power % AND Press A to lower Power % ",
wheat, 140)
        msg_screen("Pause: P", wheat, 90)

        btn("Play", 150,
500, 100, 50, green, light_green, action="play")
        btn("Main", 350, 500, 100, 50,
yellow, light_yellow, action="main")
        btn("Quit", 550, 500, 100, 50, red, light_red,
action="quit")

        pygame.display.update()

        clock.tick(15)

function for buttons having action calls and text on it callings
def btn(txt, x, y,
width, height, inactive_color, active_color, action=None,size=" "):
    cursor =
pygame.mouse.get_pos()
    click = pygame.mouse.get_pressed()
    #
print(click)
    if x + width > cursor[0] > x and y + height > cursor[1] >
y:
        pygame.draw.rect(game_layout_display, active_color, (x, y, width,
height))
        if click[0] == 1 and action != None:
            if action ==
"quit":
                pygame.quit()
                quit()

            if action ==
"controls":
                game_ctrls()

            if action ==
"play":
                gameLoop()

            if action ==
"main":
                game_intro()

    else:
        pygame.draw.rect(game_lay
out_display, inactive_color, (x, y, width, height))

    txt_btn(txt, black, x, y, width,
height)

function for pause having transparent background, uncommenting fill
statement will make it black
def pause():
    paused =
True
    msg_screen("Paused", white, -100, size="large")
    msg_screen("Press C
to continue playing or Q to quit", wheat, 25)
    pygame.display.update()
    while
paused:
        # gameDisplay.fill(black)
        for event in
pygame.event.get():

            if event.type ==
pygame.QUIT:
                pygame.quit()
                quit()
            if event.type
== pygame.KEYDOWN:
                if event.key ==
pygame.K_c:
                    paused = False
                elif event.key ==
pygame.K_q:
                    pygame.quit()
                    quit()

        clock.tic
k(5)

function for barrier
def barrier(x_loc, ran_height,
bar_width):
    pygame.draw.rect(game_layout_display, green, [x_loc, display_height -
ran_height, bar_width, ran_height])

function for explosion for both
tanks
def explosion(x, y, size=50):

    exp = True

    while
exp:
        for event in pygame.event.get():
            if event.type ==
pygame.QUIT:
                pygame.quit()
                quit()

        startPoint =
x, y

        choice_colors = [red, light_red, yellow,
light_yellow]

        mgntde = 1

        while mgntde <
size:
            exploding_bit_x = x + random.randrange(-1 * mgntde,

mgntde)
 exploding_bit_y = y + random.randrange(-1 * mgntde, mgntde)

 pygame.draw.circle(game_layout_display, choice_colors[random.randrange(0, 4)], (exploding_bit_x, exploding_bit_y),
 random.randrange(1, 5))
 mgntde += 1

 pygame.display.update()
 clock.tick(100)

 exp = False

# firing function for players tank
def playerfireShell(xy, tankx, tanky, turPost, gun_power, xloc, bar_width, ranHeight, eTankX, eTankY):

 fire = True
 damage = 0

 startShell = list(xy)

 print("FIRE!", xy)

 while fire:
 for event in pygame.event.get():
 if event.type == pygame.QUIT:
 pygame.quit()
 quit()

 # print(startingShell[0],startingShell[1])
 pygame.draw.circle(game_layout_display, red, (startShell[0], startShell[1]), 5)

 startShell[0] -= (12 - turPost) * 2

 # y = x**2
 startShell[1] += int(
 (((startShell[0] - xy[0]) * 0.015 / (gun_power / 50)) ** 2) - (turPost + turPost / (12 - turPost)))

 if startShell[1] > display_height - grnd_height:
 print("Last shell:", startShell[0], startShell[1])
 hit_x = int((startShell[0] * display_height - grnd_height) / startShell[1])
 hit_y = int(display_height - grnd_height)
 print("Impact:", hit_x, hit_y)

 if eTankX + 10 > hit_x > eTankX - 10:
 print("Critical Hit!")
 damage = 25
 elif eTankX + 15 > hit_x > eTankX - 15:
 print("Hard Hit!")
 damage = 18
 elif eTankX + 25 > hit_x > eTankX - 25:
 print("Medium Hit")
 damage = 10
 elif eTankX + 35 > hit_x > eTankX - 35:
 print("Light Hit")
 damage = 5

 explosion(hit_x, hit_y)
 fire = False

 check_x_1 = startShell[0] <= xloc + bar_width
 check_x_2 = startShell[0] >= xloc

 check_y_1 = startShell[1] <= display_height
 check_y_2 = startShell[1] >= display_height - ranHeight

 if check_x_1 and check_x_2 and check_y_1 and check_y_2:
 print("Last shell:", startShell[0], startShell[1])
 hit_x = int((startShell[0]))
 hit_y = int(startShell[1])
 print("Impact:", hit_x, hit_y)
 explosion(hit_x, hit_y)
 fire = False

 pygame.display.update()
 clock.tick(60)
 return damage

# firing function for computer's tank
def computerfireShell(xy, tankx, tanky, turPost, gun_power, xloc, bar_width, ranHeight, ptankx, ptanky):

 damage = 0
 cPower = 1
 pow_found = False

 while not pow_found:
 cPower += 1
 if cPower > 100:
 pow_found = True
 # print(currentPower)

 fire = True
 startShell = list(xy)

 while fire:
 for event in pygame.event.get():
 if event.type == pygame.QUIT:
 pygame.quit()
 quit()

 # pygame.draw.circle(gameDisplay, red, (startingShell[0],startingShell[1]),5)

 startShell[0] += (12 - turPost) * 2
 startShell[1] += int(
 (((startShell[0] - xy[0]) * 0.015 / (cPower / 50)) ** 2) - (turPost + turPost / (12 - turPost)))

 if startShell[1] > display_height - grnd_height:
 hit_x = int((startShell[0] * display_height - grnd_height) / startShell[1])
 hit_y = int(display_height - grnd_height)
 # explosion(hit_x,hit_y)
 if ptankx + 15 > hit_x > ptankx - 15:
 print("target acquired!")
 power_found =

True
 fire = False

 check_x_1 = startShell[0] <= xloc + bar_width
 check_x_2 = startShell[0] >= xloc

 check_y_1 = startShell[1] <= display_height
 check_y_2 = startShell[1] >= display_height - ranHeight

 if check_x_1 and check_x_2 and check_y_1 and check_y_2:
 hit_x = int((startShell[0]))
 hit_y = int(startShell[1])
 # explosion(hit_x,hit_y)
 fire = False

 fire = True
 startShell = list(xy)
 print("FIRE!", xy)

 while fire:
 for event in pygame.event.get():
 if event.type == pygame.QUIT:
 pygame.quit()
 quit()

 pygame.draw.circle(game_layout_display, red, (startShell[0], startShell[1]), 5)

 startShell[0] += (12 - turPost) * 2

 gun_power = random.randrange(int(cPower * 0.90), int(cPower * 1.10))

 startShell[1] += int(
 (((startShell[0] - xy[0]) * 0.015 / (gun_power / 50)) ** 2) - (turPost + turPost / (12 - turPost)))

 if startShell[1] > display_height - grnd_height:
 print("last shell:", startShell[0], startShell[1])
 hit_x = int((startShell[0] * display_height - grnd_height) / startShell[1])
 hit_y = int(display_height - grnd_height)
 print("Impact:", hit_x, hit_y)

 if ptankx + 10 > hit_x > ptankx - 10:
 print("Critical Hit!")
 damage = 25
 elif ptankx + 15 > hit_x > ptankx - 15:
 print("Hard Hit!")
 damage = 18
 elif ptankx + 25 > hit_x > ptankx - 25:
 print("Medium Hit")
 damage = 10
 elif ptankx + 35 > hit_x > ptankx - 35:
 print("Light Hit")
 damage = 5

 explosion(hit_x, hit_y)
 fire = False

 check_x_1 = startShell[0] <= xloc+ bar_width
 check_x_2 = startShell[0] >= xloc

 check_y_1 = startShell[1] <= display_height
 check_y_2 = startShell[1] >= display_height - ranHeight

 if check_x_1 and check_x_2 and check_y_1 and check_y_2:
 print("Last shell:", startShell[0], startShell[1])
 hit_x = int((startShell[0]))
 hit_y = int(startShell[1])
 print("Impact:", hit_x, hit_y)
 explosion(hit_x, hit_y)
 fire = False

 pygame.display.update()
 clock.tick(60)
 return damage

# function for power level of players tank
def power(level):
 text = s_font.render("Power: " + str(level) + "%", True, wheat)
 game_layout_display.blit(text, [display_width / 2, 0])

# function for intro screen
def game_intro():
 intro = True

 while intro:
 for event in pygame.event.get():
 # print(event)
 if event.type == pygame.QUIT:
 pygame.quit()
 quit()

 if event.type == pygame.KEYDOWN:
 if event.key == pygame.K_c:
 intro = False
 elif event.key == pygame.K_q:

 pygame.quit()
 quit()

 game_layout_display.fill(black)
 msg_screen("Welcome to Tanks War!", white, -100, size="Medium")
 msg_screen("The goal is to shoot and destroy", wheat, 15)
 msg_screen("the enemy tank before they destroy you.", wheat, 60)
 msg_screen("The more enemies you destroy, the highest score you get.", wheat, 110)
 msg_screen("Brought To You by :itsourcecode.com", wheat, 280)
 # message_to_screen("Press C to play, P to pause or Q to quit",black,180)

 btn("Play", 150, 500, 100, 50, wheat, light_green, action="play",size="vsmall")
 btn("Controls", 350, 500, 100, 50, wheat,

light_yellow, action="controls",size="vsmall")
 btn("Quit", 550, 500, 100, 50, wheat, light_red,
action="quit",size="vsmall")

 pygame.display.update()

 clock.tick(15)

# function for game Over screen
def game_over():
 game_over = True

 while game_over:
 for event in pygame.event.get():
 # print(event)
 if event.type == pygame.QUIT:
 pygame.quit()
 quit()

 game_layout_display.fill(black)
 msg_screen("Game Over", white, -100, size="large")
 msg_screen("You died.", wheat, -30)

 btn("Play Again", 150, 500, 150, 50, wheat, light_green, action="play")
 btn("Controls", 350, 500, 100, 50, wheat, light_yellow, action="controls")
 btn("Quit", 550, 500, 100, 50, wheat, light_red,
action="quit")

 pygame.display.update()

 clock.tick(15)

# function for players win screen
def you_win():
 win = True

 while win:
 for event in pygame.event.get():
 # print(event)
 if event.type == pygame.QUIT:
 pygame.quit()
 quit()

 game_layout_display.fill(black)
 msg_screen("You won!", white, -100, size="large")
 msg_screen("Congratulations!", wheat, -30)

 btn("play Again", 150, 500, 150, 50, wheat, light_green, action="play")
 btn("controls", 350, 500, 100, 50, wheat, light_yellow, action="controls")
 btn("quit", 550, 500, 100, 50, wheat, light_red,
action="quit")

 pygame.display.update()

 clock.tick(15)

# function for health bars of both tanks
def health_bars(p_health, e_health):
 if p_health > 75:
 p_health_color = green
 elif p_health > 50:
 p_health_color = yellow
 else:
 p_health_color = red

 if e_health > 75:
 e_health_color = green
 elif e_health > 50:
 e_health_color = yellow
 else:
 e_health_color = red

 pygame.draw.rect(game_layout_display, p_health_color, (680, 25, p_health, 25))
 pygame.draw.rect(game_layout_display, e_health_color, (20, 25, e_health, 25))

# function for main gameloop
def gameLoop():
 gExit = False
 gOver = False
 FPS = 15

 p_health = 100
 e_health = 100

 bar_width = 50

 mTankX = display_width * 0.9
 mTankY = display_height * 0.9
 tnkMove = 0
 curTurPost = 0
 changeTurs = 0

 eTankX = display_width * 0.1
 eTankY = display_height * 0.9

 f_power = 50
 p_change = 0

 xloc = (display_width / 2) + random.randint(-0.1 * display_width, 0.1 * display_width)
 ranHeight = random.randrange(display_height * 0.1, display_height * 0.6)

 while not gExit:

 if gOver == True:
 # gameDisplay.fill(white)
 msg_screen("Game Over", red, -50, size="large")
 msg_screen("Press C to play again or Q to exit", black, 50)
 pygame.display.update()
 while gOver == True:
 for event in pygame.event.get():
 if event.type == pygame.QUIT:
 gExit = True
 gOver = False

 if event.type == pygame.KEYDOWN:
 if event.key == pygame.K_c:
 gameLoop()
 elif event.key == pygame.K_q:

 gExit = True
 gOver = False

 for event in pygame.event.get():

 if event.type == pygame.QUIT:
 gExit = True

 if event.type == pygame.KEYDOWN:
 if event.key ==

pygame.K_LEFT:
 tnkMove = -5

 elif event.key == pygame.K_RIGHT:
 tnkMove = 5

 elif event.key == pygame.K_UP:
 changeTurs = 1

 elif event.key == pygame.K_DOWN:
 changeTurs= -1

 elif event.key == pygame.K_p:
 pause()

 elif event.key == pygame.K_SPACE:

 damage = playerfireShell(gun, mTankX, mTankY, curTurPost, f_power, xloc, bar_width,
 ranHeight, eTankX, eTankY)
 e_health -= damage

 posMovement = ['f', 'r']
 moveInd = random.randrange(0, 2)

 for x in range(random.randrange(0, 10)):

 if display_width * 0.3 > eTankX > display_width * 0.03:
 if posMovement[moveInd] == "f":
 eTankX += 5
 elif posMovement[moveInd] == "r":
 eTankX -= 5

 game_layout_display.fill(black)
 health_bars(p_health, e_health)
 gun = tank(mTankX, mTankY, curTurPost)
 e_gun = computer_tank(eTankX, eTankY, 8)
 f_power += p_change

 power(f_power)

 barrier(xloc, ranHeight, bar_width)
 game_layout_display.fill(green,
 rect=[0, display_height - grnd_height, display_width, grnd_height])
 pygame.display.update()

 clock.tick(FPS)

 damage = computerfireShell(e_gun, eTankX, eTankY, 8, 50, xloc, bar_width,
 ranHeight, mTankX, mTankY)
 p_health -= damage

 elif event.key == pygame.K_a:
 p_change = -1
 elif event.key == pygame.K_d:
 p_change = 1

 elif event.type == pygame.KEYUP:
 if event.key == pygame.K_LEFT or event.key == pygame.K_RIGHT:
 tnkMove = 0

 if event.key == pygame.K_UP or event.key == pygame.K_DOWN:
 changeTurs = 0

 if event.key == pygame.K_a or event.key == pygame.K_d:
 p_change = 0

 mTankX += tnkMove

 curTurPost += changeTurs

 if curTurPost > 8:
 curTurPost = 8
 elif curTurPost < 0:
 curTurPost = 0

 if mTankX - (tnk_width / 2) < xloc + bar_width:
 mTankX += 5

 game_layout_display.fill(black)
 health_bars(p_health, e_health)
 gun = tank(mTankX, mTankY, curTurPost)
 e_gun = computer_tank(eTankX, eTankY, 8)

 f_power += p_change

 if f_power > 100:
 f_power = 100
 elif f_power < 1:
 f_power = 1

 power(f_power)

 barrier(xloc, ranHeight, bar_width)
 game_layout_display.fill(green, rect=[0, display_height - grnd_height, display_width, grnd_height])
 pygame.display.update()

 if p_health < 1:
 game_over()
 elif e_health < 1:
 you_win()
 clock.tick(FPS)

 pygame.quit()
 quit()

game_intro()
gameLoop()

How To Run the Tank Game Python with Source Code?

To run this project, you must have installed a **Pycharm** on your PC (for Windows). **Tank Game Python with Source Code** is for educational purposes only!

After downloading the project you must follow the steps below:

Step 1: Unzip the file or Extract the file

Step 2: Double click the **TankGame**

Step 3: Project is ready to run