Ojas Chaturvedi

AP CSP – Bradley

Protect the President Create task writeup

3a.

i) “Protect the President” was created in PyCharm. The overall purpose of the program is for the user to play a fun game called “Protect the President,” in which the user has to kill the aliens and protect the president. This allows for the user to think up different strategies and implement them and see which strategy would work the best, while also letting the user play an entertaining game for their enjoyment.

ii) The video shows the levels of the game “protect the president.” It also shows the different power ups used to help the player achieve the win.

iii) The inputs of the program include keys on the keyboard. The “W,A,S,D” keys are used to move the shooter across the board. The spacebar is used to shoot bullets at the aliens. The “Q,E,X” keys are used for the different powerups.

3b.

i) Code:

numberAliens = 14

aliens = []

for i in range(numberAliens):

aliens.append(turtle.Turtle())

ii) Code:

while True:

shooter.sety(shooter.ycor() - gravity)

if shooter.ycor() <= -100:

shooter.sety(-100)

pass

for alien in aliens:

alien.setx(alien.xcor() - alienspeed)

if alien.xcor() == -325:

winsound.PlaySound("explosion.wav",winsound.SND\_ASYNC)

shooter.hideturtle()

shelter.hideturtle()

scorePen.penup()

scorePen.goto(0, 0)

scorePen.color("red")

scorePen.write("You failed your job", False, align="center", font=('Arial', 24, 'normal'))

failScreen()

if bulletstate == "fire":

if Collision(bullet, alien):

bullet.hideturtle()

winsound.PlaySound("explosion.wav",winsound.SND\_ASYNC)

alien.goto(1000, 1000)

bulletstate = "ready"

score += 10

scoreString = "Score: %s" % score

scorePen.clear()

scorePen.write(scoreString, False, align="left", font=('Arial', 14, 'normal'))

if score == 140:

shooter.hideturtle()

shelter.hideturtle()

scorePen.clear()

screen.setup(1200, 800)

screen.bgpic("flag.gif")

president.showturtle()

scorePen.penup()

scorePen.goto(0, 0)

scorePen.color("red")

scorePen.write("You saved the president and Completed the game!", False, align="center", font=('Arial', 24, 'normal'))

time.sleep(3)

screen.bye()

print(f"Ended: {time.strftime('%X')}")

if showshelter==True:

if Collision(shelter, alien):

alienspeed = 0

smashwall += 1

if smashwall == 50:

shelter.hideturtle()

alienspeed = 2.5

showshelter=False

if deployarmies == True:

for army in armies:

army.setx(army.xcor() + armyspeed)

if army.xcor() >= alien.xcor():

for alien in aliens:

alien.setx(alien.xcor() + 50)

if alien.xcor() >= 450:

alien.setx(450)

for army in armies:

army.hideturtle()

army.goto(-450, -450)

deployarmies = False

bullet.setx(bullet.xcor() + bulletspeed)

if bullet.xcor() >= 450:

bullet.hideturtle()

bulletstate = "ready"

iii) The name of the list being used in this program code segment is called “aliens”

iv) The list “aliens” is being used to keep track of all the aliens and to set them up on the board. After all the aliens are set up, there is a for loop used to keep the aliens moving across the board towards the shooter to reach the president.

v) The list “aliens” decreases the complexity of the program by organizing the code efficiently and decreasing the length of the code. If the list was not here, there would be a lot of extra code used to move the aliens one by one, and it could mess up the code and cause a bunch of mistakes from having disorganized code.

3c.

i) Code:

**def** Collision**(**t1, t2**)**:

distance = math.sqrt**(**math.pow**(**t1.xcor**()** - t2.xcor**()**, 2**)** + math.pow**(**t1.ycor**()** - t2.ycor**()**, 2**))**

**if** distance <= 15:

**return True**

**else**:

**return False**

ii) Code:

**for** alien **in** aliens:

alien.setx**(**alien.xcor**()** - alienspeed**)**

**if** alien.xcor**()** == -325:

winsound.PlaySound**("explosion.wav"**, winsound.SND\_ASYNC**)**

shooter.hideturtle**()**

shelter.hideturtle**()**

scorePen.penup**()**

scorePen.goto**(**0, 0**)**

scorePen.color**("red")**

scorePen.write**("You failed your job"**, **False**, align=**"center"**, font=**('Arial'**, 24, **'normal'))**

failScreen**()**

**if** bulletstate == **"fire"**:

**if** Collision**(**bullet, alien**)**:

bullet.hideturtle**()**

winsound.PlaySound**("explosion.wav"**, winsound.SND\_ASYNC**)**

alien.goto**(**1000, 1000**)**

bulletstate = **"ready"**

score += 10

scoreString = **"Score: %s"** % score

scorePen.clear**()**

scorePen.write**(**scoreString, **False**, align=**"left"**, font=**('Arial'**, 14, **'normal'))**

**if** score == 140:

shooter.hideturtle**()**

shelter.hideturtle**()**

scorePen.clear**()**

screen.setup**(**1200, 800**)**

screen.bgpic**("flag.gif")**

president.showturtle**()**

scorePen.penup**()**

scorePen.goto**(**0, 0**)**

scorePen.color**("red")**

scorePen.write**("You saved the president and Completed the game!"**, **False**, align=**"center"**, font=**('Arial'**, 24, **'normal'))**

time.sleep**(**3**)**

screen.bye**()**

print**(f"Ended: {**time.strftime**('%X')}")**

**if** showshelter==**True**:

**if** Collision**(**shelter, alien**)**:

alienspeed = 0

smashwall += 1

**if** smashwall == 50:

shelter.hideturtle**()**

alienspeed = 2.5

showshelter=**False**

iii) The procedure “Collision” is used to check if the bullet hits one of the aliens, therefore killing the alien. It is inside the for loop to check if the bullet hits any one of the multiple aliens on the screen.

iv) The algorithm using in the procedure “Collision” is the basic distance formula, which is the square root of x2 - x1 squared, plus y2 - y1 squared. To create this, you will need to import math, and then use the “math.pow()” function to square the values. To get the variables x1,y1,x2,y2, you will need to ask for them as the parameter.

3d.

i) Both of the calls are located inside the for loop that moves the aliens one by one. The first call is used to check the collision of the bullet and the alien, while the second call is used to check the collision of the wall and the alien.

ii) For the first call, the condition that should be tested is if the alien is actually disappearing off the screen and the score updating if the bullet hits the alien. The condition that should be tested for the second call is if the alien actually stops moving when hitting the wall.

iii) The result for the first call is the alien is disappearing and the score is increasing by 10. The result of the second call is that the aliens are stopping in front of the wall until the wall breaks down.