Putting it all together

Have a look at final.py

To finish off the game, we need to add some bullets for the rocket to fire at the UFO. We can use a bullet png image.

```
bulletImage = pygame.image.load('bullet.png')
```

We also need some variables for the position of the bullet on screen

```
bullet_X = 0
bullet_Y = y #y is the y position of the rocket
```

How many pixels to move the bullet at a time

```
bullet_Xchange = 0
bullet_Ychange = 3
```

and a condition as to whether a bullet has been fired or not

```
bullet_state = "nofire"
```

Next, we need to update the event handler, so we can fire the bullet with the space bar. We move the position of the bullet image to the same position as the rocket plus about 30 pixels so it looks like its coming out of the front of the rocket. Draw the bullet on the screen and set the bullet state to fire. This is how the program knows when a bullet has been fired or not.

```
for event in pygame.event.get():
   if event.type == pygame.QUIT:
      running = 0 #close
   elif event.type == pygame.KEYDOWN:
      if event.key == pygame.K_LEFT:
         x = x - 10 #shift image left 10 pixels
      elif event.key == pygame.K_RIGHT:
         x = x + 10 #shift image right 10 pixels
      elif event.key == pygame.K_SPACE:  # file bullet
         if bullet_state == "nofire":
            bullet_X = x + 30 #move bullet to rocket position
            gamewindow.blit(bulletImage, (bullet_X, bullet_Y))
            bullet_state = "fire"
```

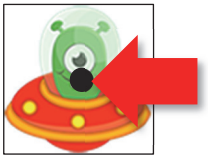
Next we need to move the bullet until it goes off the top of the screen which is 0 on the y axis

```
if bullet_Y <= 0:
   bullet_Y = y
   bullet_state = "nofire"
```

```
if bullet_state == "fire":
   gamewindow.blit(bulletImage, (bullet_X, bullet_Y))
   bullet_state = "fire"
   bullet_Y -= bullet_Ychange
```

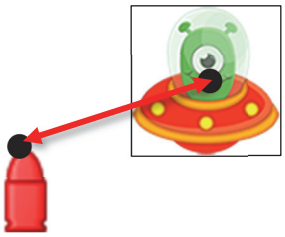
Now, what happens when a bullet hits the ufo? This is called a collision. First we need to add some variables to contain the actual position of the ufo. Here, we've taken the center of the rectangle containing the ufo image.

```
ufo_X_pos = ufo_rect.centerx
ufo_Y_pos = ufo_rect.centery
```



We need to define a function. We take the x pos and y pos of the ufo, along with the x and y position of the bullet and run it through a formula to calculate the distance between them. If the distance is less than 35 pixels we consider that a hit and return true.

```
def isCollision(ufo_X_pos, ufo_Y_pos, bullet_X, bullet_Y):
    distance = math.sqrt(math.pow(ufo_X_pos - bullet_X, 2) +
    (math.pow(ufo_Y_pos - bullet_Y, 2)))
    if distance < 35:
      return True
    else:
      return False
```



After we've defined the collision detection function, we can call the function to check the distance between the bullet and the ufo

```
collision = isCollision(ufo_X_pos, ufo_Y_pos, bullet_X, bullet_Y)
```

Once the UFO is hit, we can reset the bullet position, and the bullet state then show an explosion image to show the player the ufo has been hit

```
if collision:
   bullet_Y = 380
   bullet_state = "nofire"
   gamewindow.blit(exp, (ufo_X_pos, ufo_Y_pos))
```

Once the ufo has been hit, we can redraw the ufo on a random position on the screen.

```
   #place ufo back on screen in random position
   ufo_rect.centerx = random.randint(0, 736)
   ufo_rect.centery = random.randint(50, 300)
```

There are a few bugs in the code, try it out and see if you can find them and improve the program. Have fun.