
# GAME #2

## “COIN COLLECTOR”


VORTEX ACADEMY
THE ART OF THINKING

VORTEX ACADEMY
THE ART OF THINKING

# WELCOME BACK!

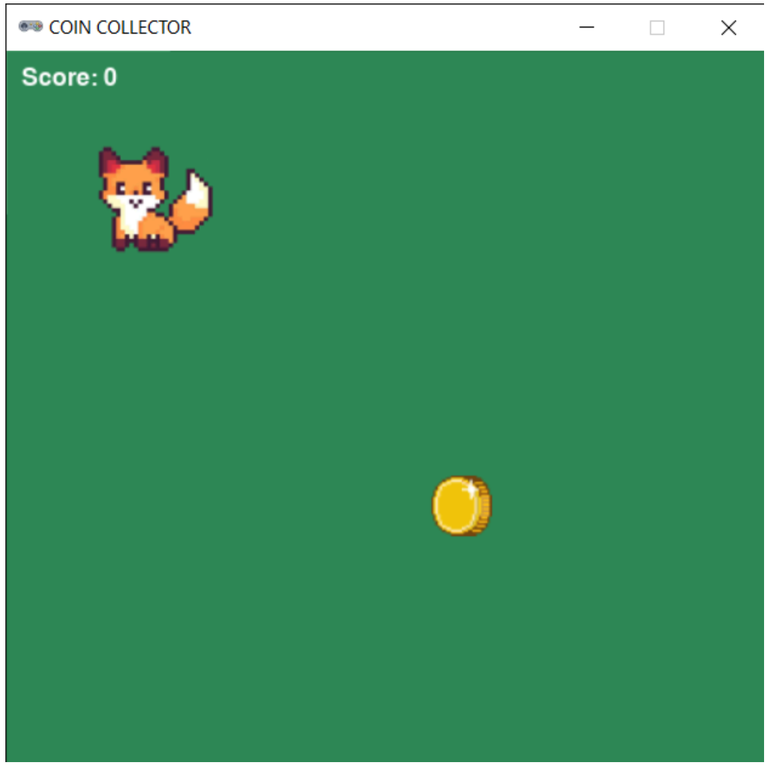
# CAN YOU GUESS WHAT WE'LL CREATE TODAY?

VORTEX ACADEMY
THE ART OF THINKING


# HOW DO WE MAKE COIN COLLECTOR?

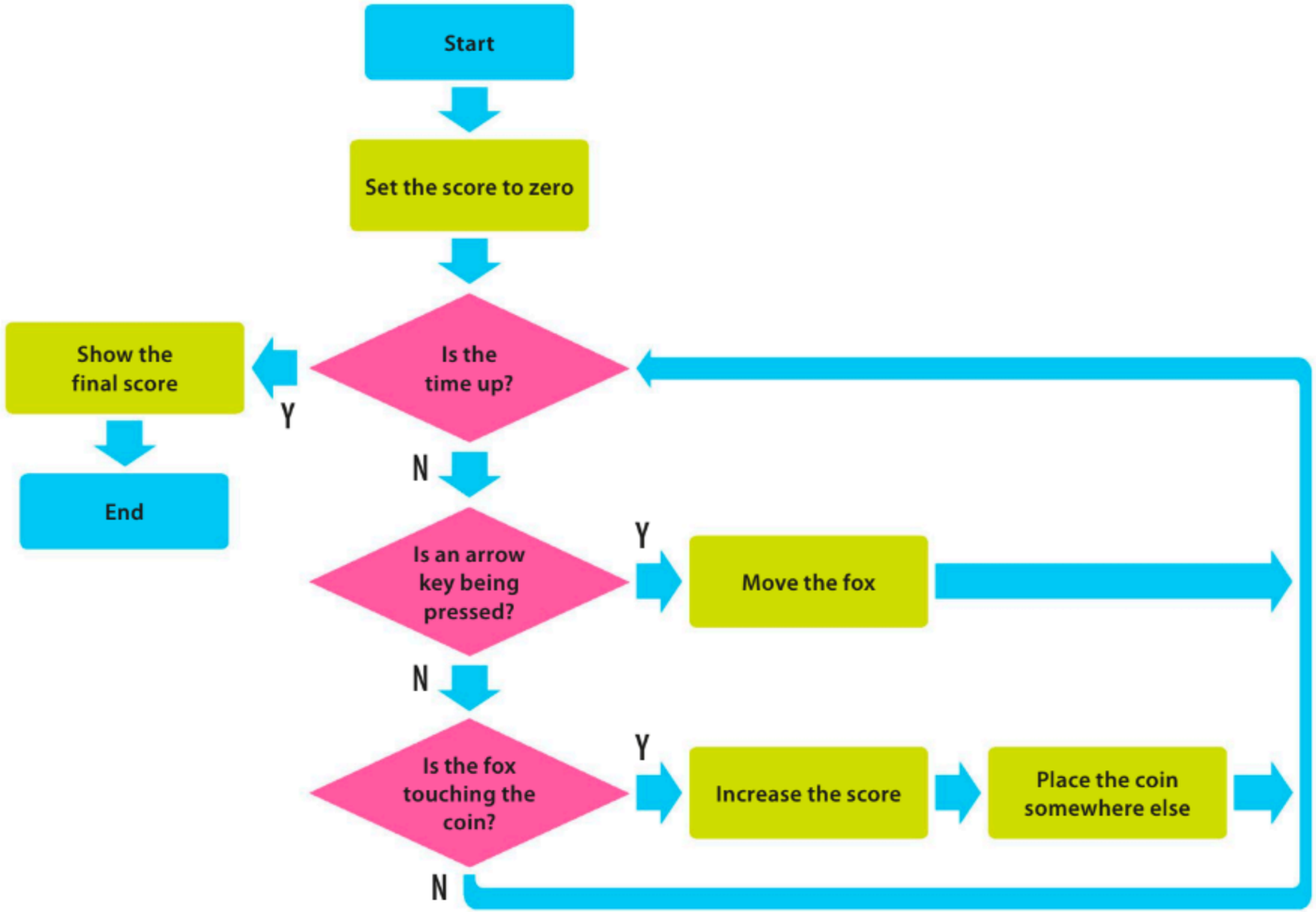
VORTEX ACADEMY
THE ART OF THINKING

Using what you've learned from the previous session, how do you think we could build COIN COLLECTOR?




# HOW DO WE MAKE COIN COLLECTOR?

VORTEX ACADEMY
THE ART OF THINKING




# TOPICS

VORTEX ACADEMY
THE ART OF THINKING

01

## Using The Keyboard

- How do we make our game use the keyboard to move characters?

02

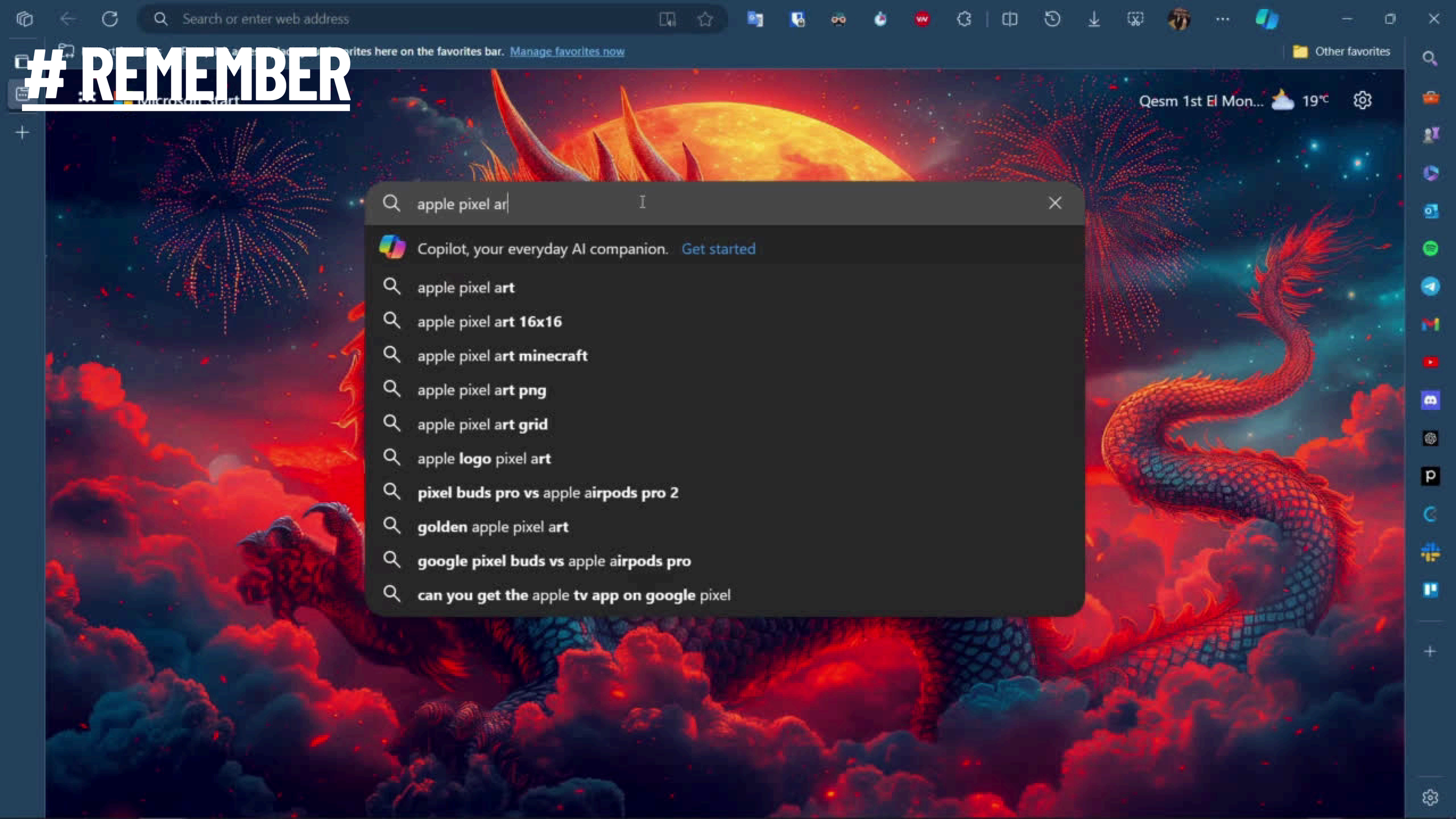
## Collision Detection

- How do we make our game do something when two actors touch?

03

## Clock & COIN COLLECTOR

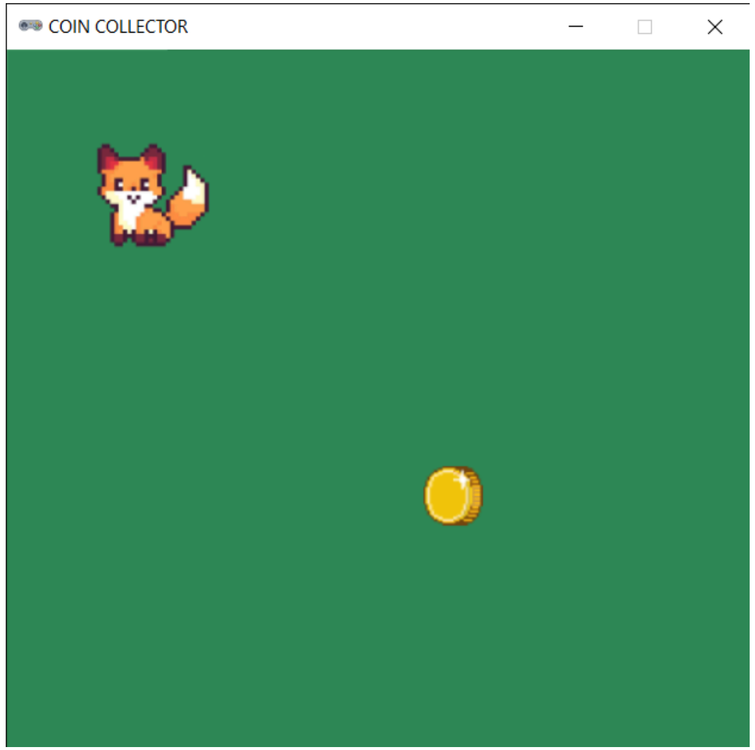
- How do we make our game do something after a certain amount of time?
- Building coin collector.


# REMEMBER
Search or enter web address
Other favorites
Qesm 1st El Mon...
19°C
apple pixel ar
Copilot, your everyday AI companion. Get started
apple pixel art
apple pixel art 16x16
apple pixel art minecraft
apple pixel art png
apple pixel art grid
apple logo pixel art
pixel buds pro vs apple airpods pro 2
golden apple pixel art
google pixel buds vs apple airpods pro
can you get the apple tv app on google pixel

# LET'S START

VORTEX ACADEMY
THE ART OF THINKING

Use what you've learned to make the background green and place our two actors.
You can look at your previous code and copy from it.




You can use this coin

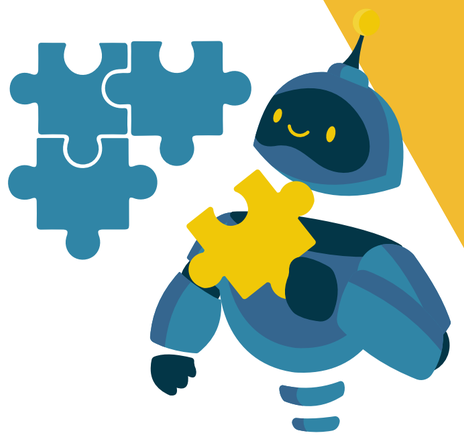
You can use this fox

VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#1

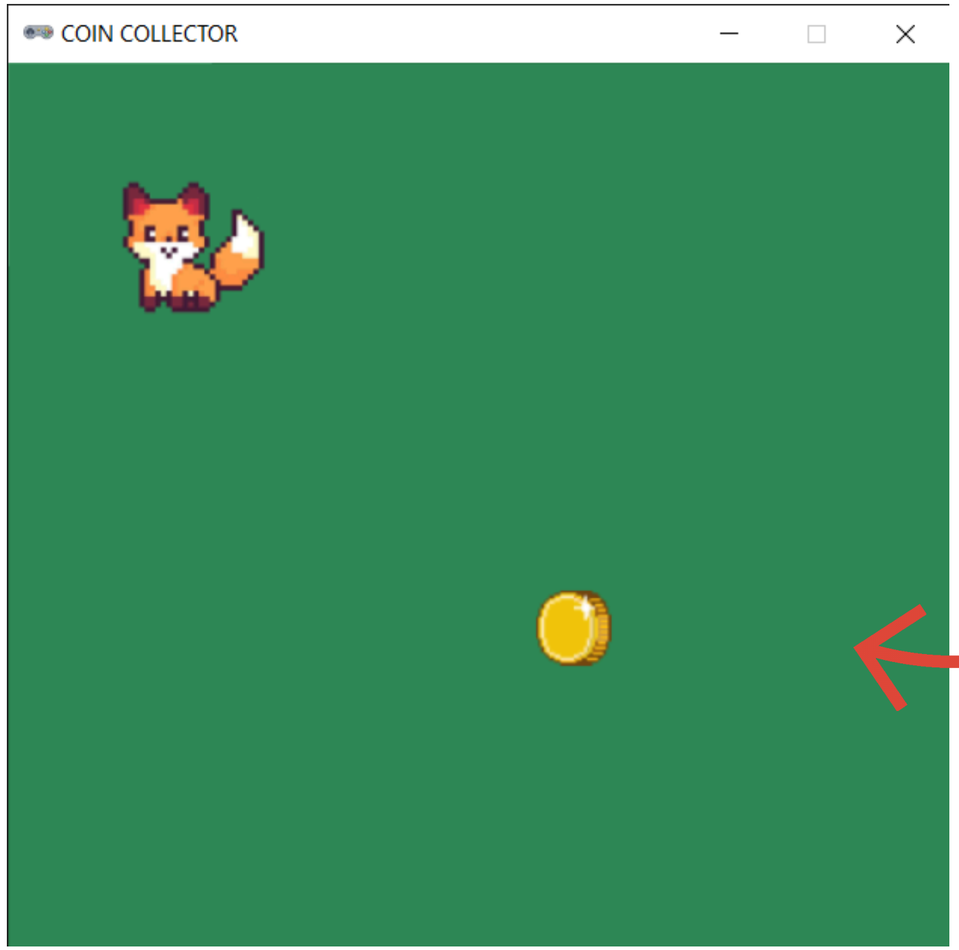
## Try the following code

```
import pgzrun

WIDTH = 500
HEIGHT = 500

TITLE = "COIN COLLECTOR"

player = Actor('fox')
player.pos = 100, 100

coin = Actor('coin')
coin.pos = 300, 300

def draw():
  screen.clear()
  screen.fill("SeaGreen")
  coin.draw()
  player.draw()

def update():
  pass

pgzrun.go()
```

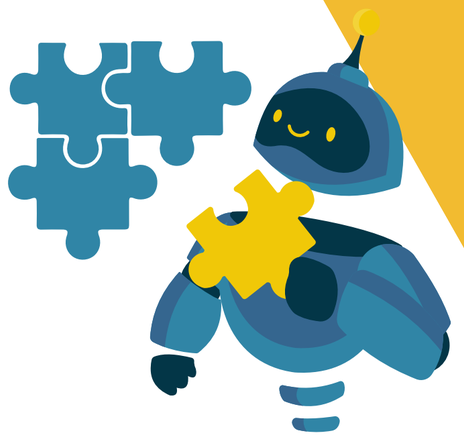


# ACTIVITY#1

## OUTPUT:




**Now we have setup the base for our game, but how do we add the background picture?**



VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#2

## Add To The Images Folder:

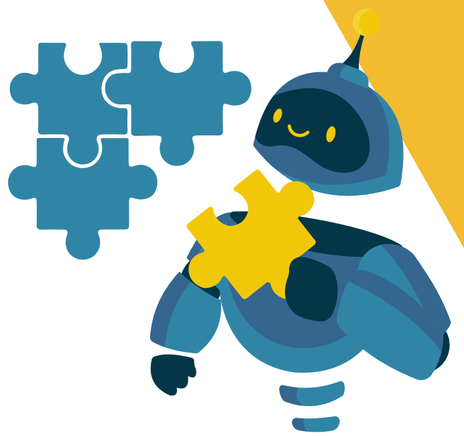
-A 500X500 pixel image to be used as the game background.

Use this Background

**#Tip:**you can use Freepik to find images for your games

## Modify Your Previous Code & Try:

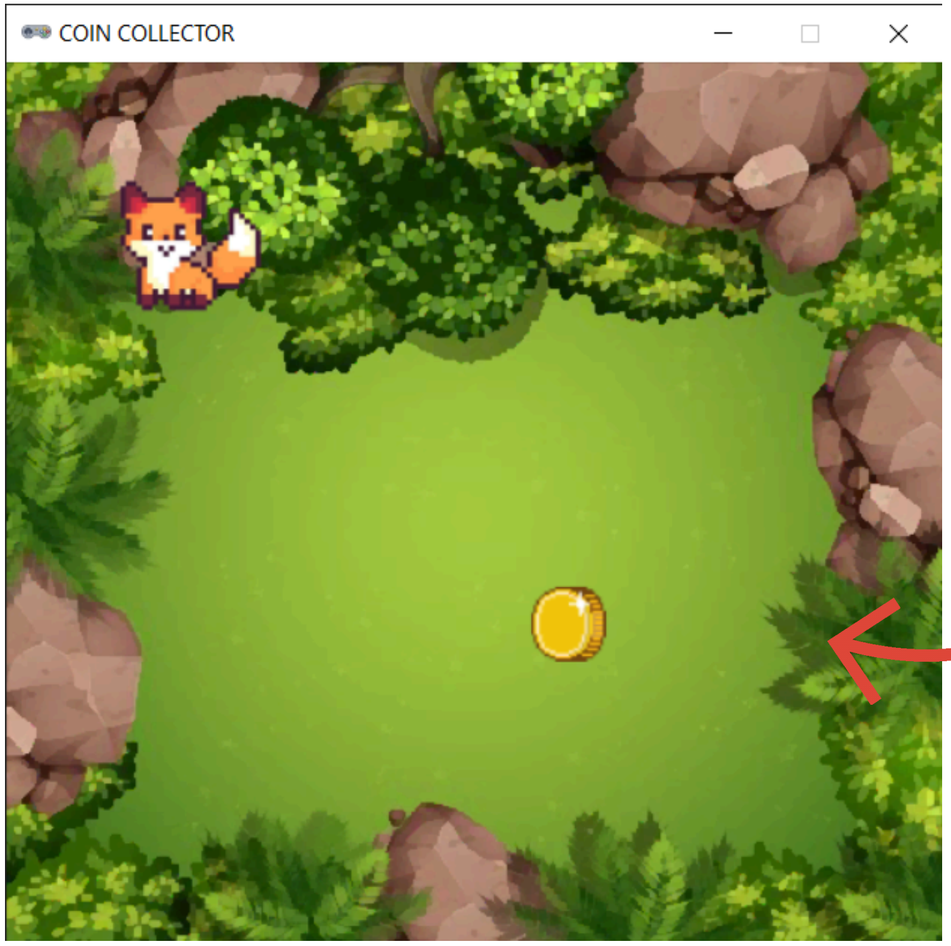
```
# Add the line in yellow to the draw() function
def draw():
  screen.clear()
  screen.blit("background", (0, 0))
  coin.draw()
  player.draw()
```



VORTEX ACADEMY
THE ART OF THINKING

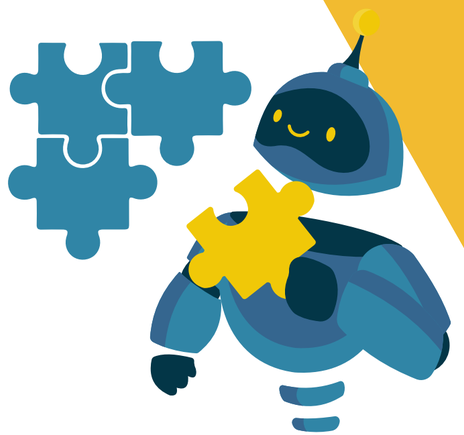
# ACTIVITY#2

## OUTPUT:




**Now we have added a image to our game as the background**

#HINT: Remember to watch out for drawing order when drawing elements to the screen.

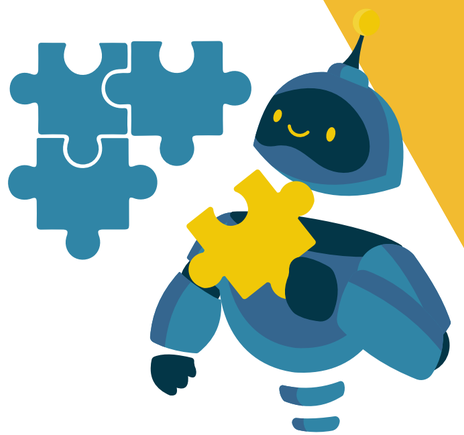


VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#3

## Modify Your Previous Code & Try:

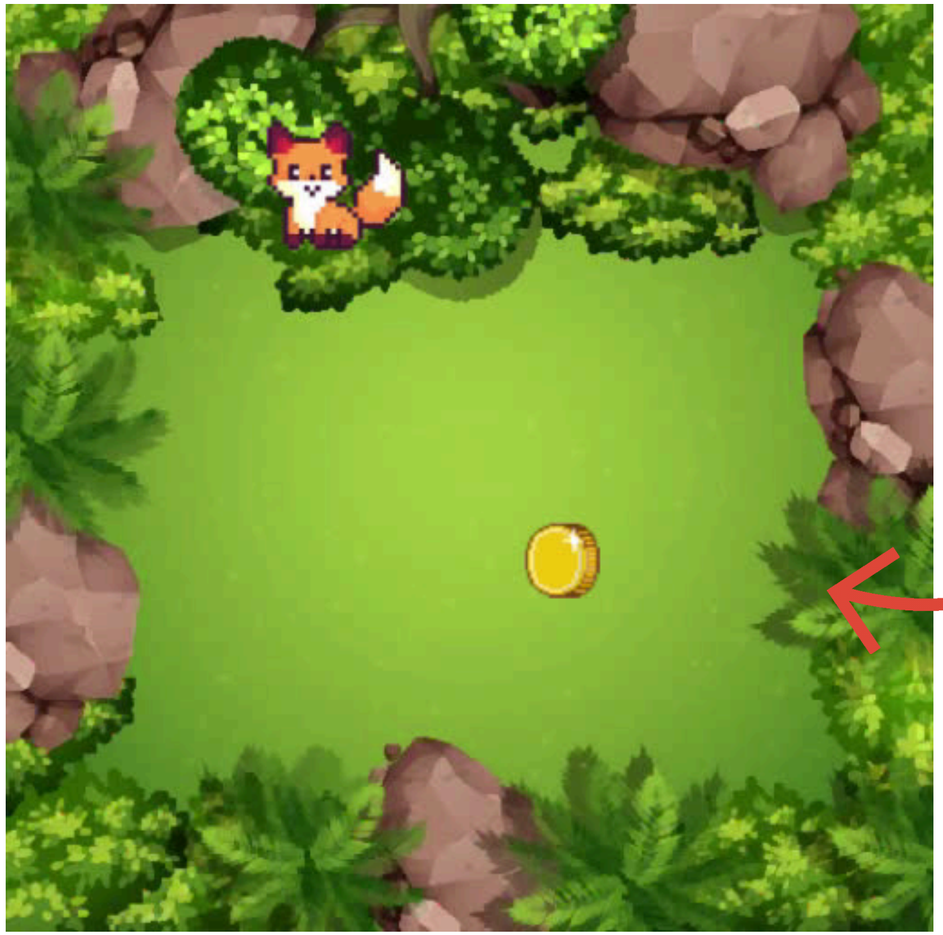
```
# Add this to the update function.
def update():
  if keyboard.Right:
    if player.x < WIDTH:
      player.x += 2
    else:
      player.x = 0
```



VORTEX ACADEMY
THE ART OF THINKING

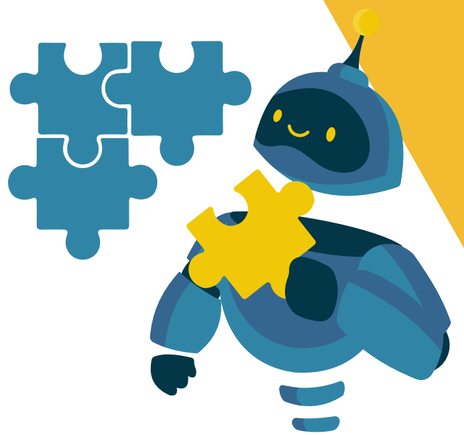
# ACTIVITY#3

## OUTPUT:



**Now the fox moves to the right when we press the right key on the keyboard.**

#HINT: try replacing keyboard.**Right** with keyboard.**D,** What happens when you press the D key on the keyboard?

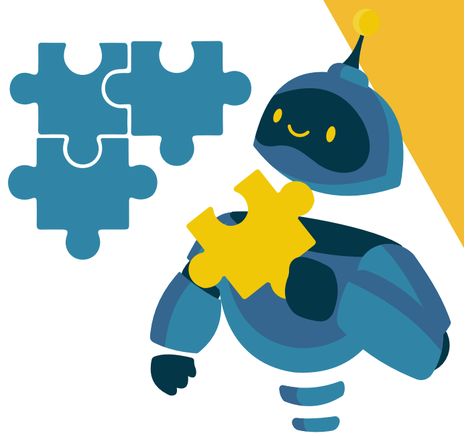


VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#4

## Modify Your Previous Code & Try:

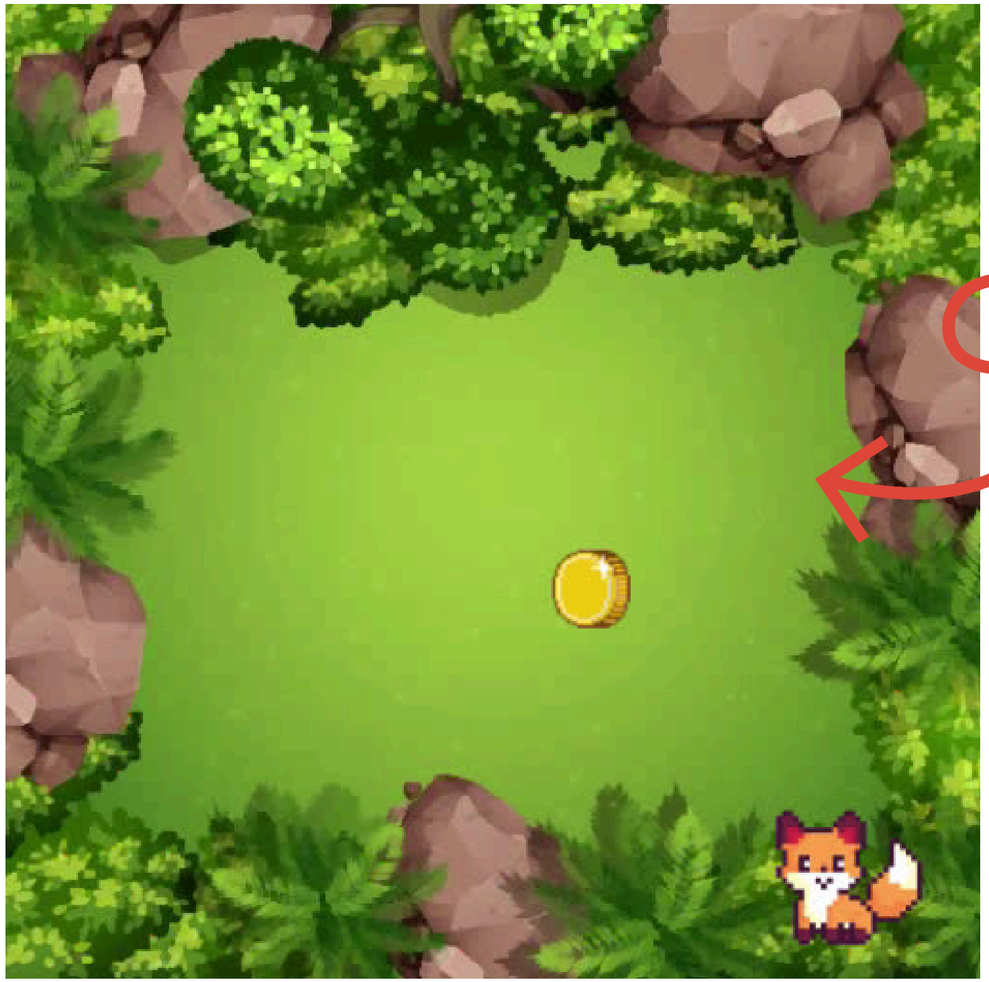
```
def update(): # Add the lines in yellow to the update function.
 if keyboard.Right:
  if player.x < WIDTH:
   player.x += 2
  else:
   player.x = 0
 if keyboard.Left:
  if player.x > 0:
   player.x -= 2
  else:
   player.x = WIDTH
 if keyboard.Up:
  if player.y > 0:
   player.y -= 2
  else:
   player.y = HEIGHT
 if keyboard.Down:
  if player.y < HEIGHT:
   player.y += 2
  else:
   player.y = 0
```



VORTEX ACADEMY
THE ART OF THINKING

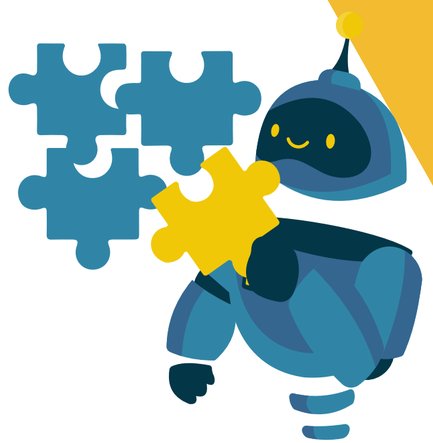
# ACTIVITY#4

## OUTPUT:



**Now our fox can move in all four directions and loop around the screen.**

#HINT: try replacing
keyboard.**Right,** with keyboard.**D**
keyboard.**Left,** with keyboard.**A**
keyboard.**UP** with keyboard.**W**
keyboard.**Down** with keyboard.**S**
What happens when you press the WASD keys on the keyboard?



# USING THE KEYBOARD

VORTEX ACADEMY
THE ART OF THINKING

The second way we can interact with our game is using the keyboard.

We use keyboard."KEY_NAME" to check if a certain key is pressed.
By adding to our **update()** function the logic to check for which key is pressed we can make our game respond to different keyboard keys.

In the previous example we used **if** with *keyboard.**Up***, *keyboard.**Down***, *keyboard.**Right*** ,*keyboard.**Left*** to check which of the arrow keys is pressed and move our character accordingly.

---

**Syntax:**

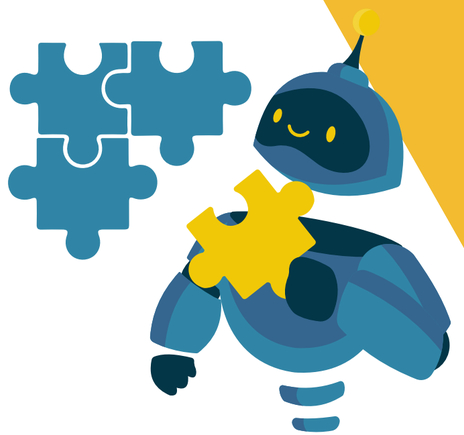
```
def update():
    if keyboard."KEY_NAME":
        # The actions we want to happen when KEY_NAME is pressed.
```

VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#5

## Modify Your Previous Code & Try:

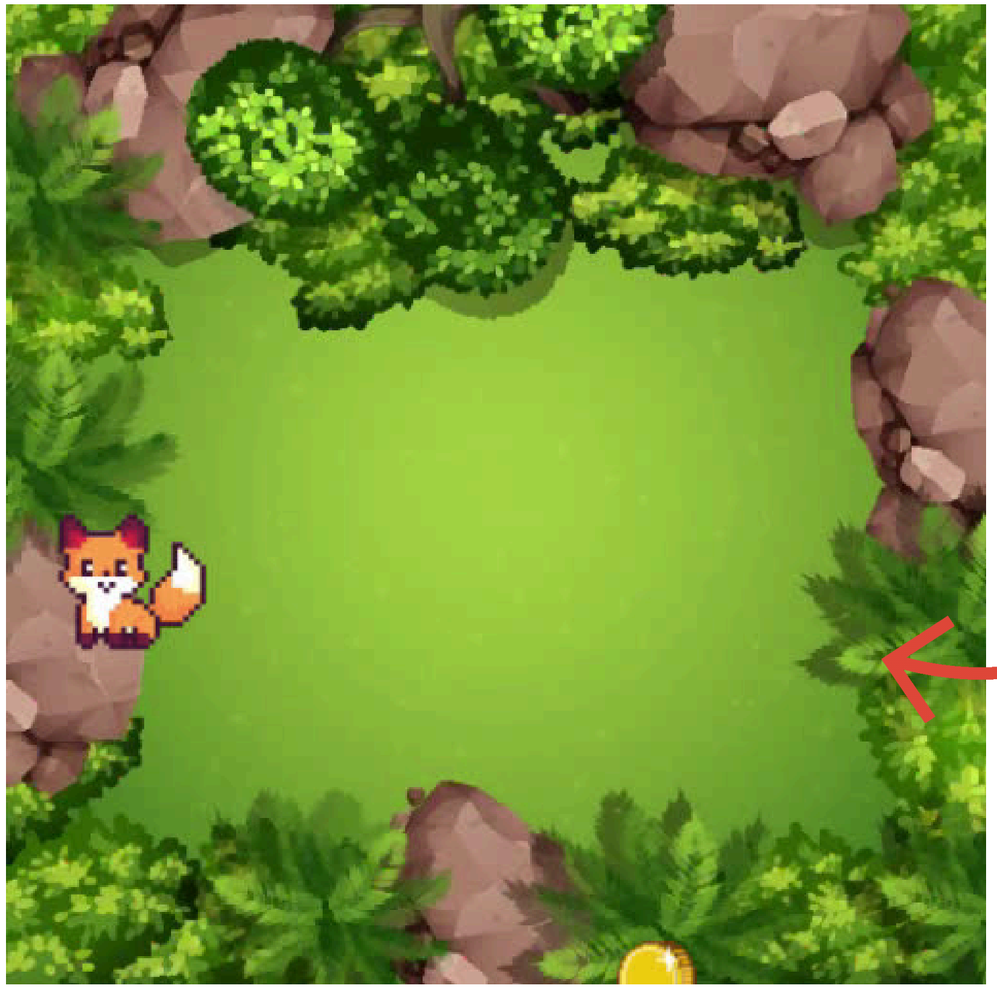
```
# Add this import
from random import randint

# Modify the update function by adding the following lines.
def update():
    #The previous code for keyboard control
    if player.colliderect(coin):
      coin.pos = randint(0, 500), randint(0, 500)
```



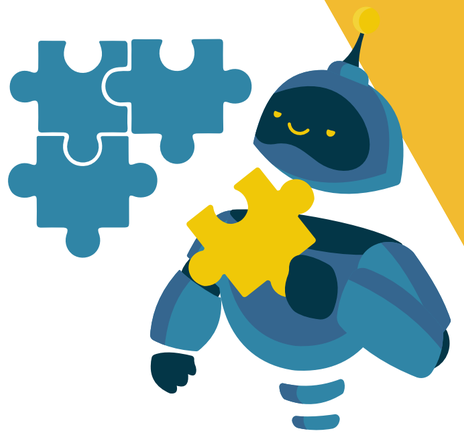
VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#5

## OUTPUT:



Now the coin jumps around the screen randomly every time the fox collects it

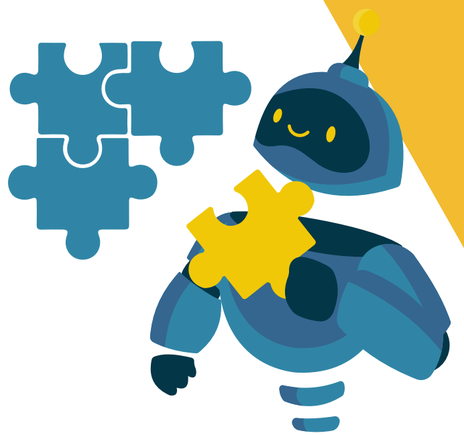


VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#6

## Modify Your Previous Code & Try:

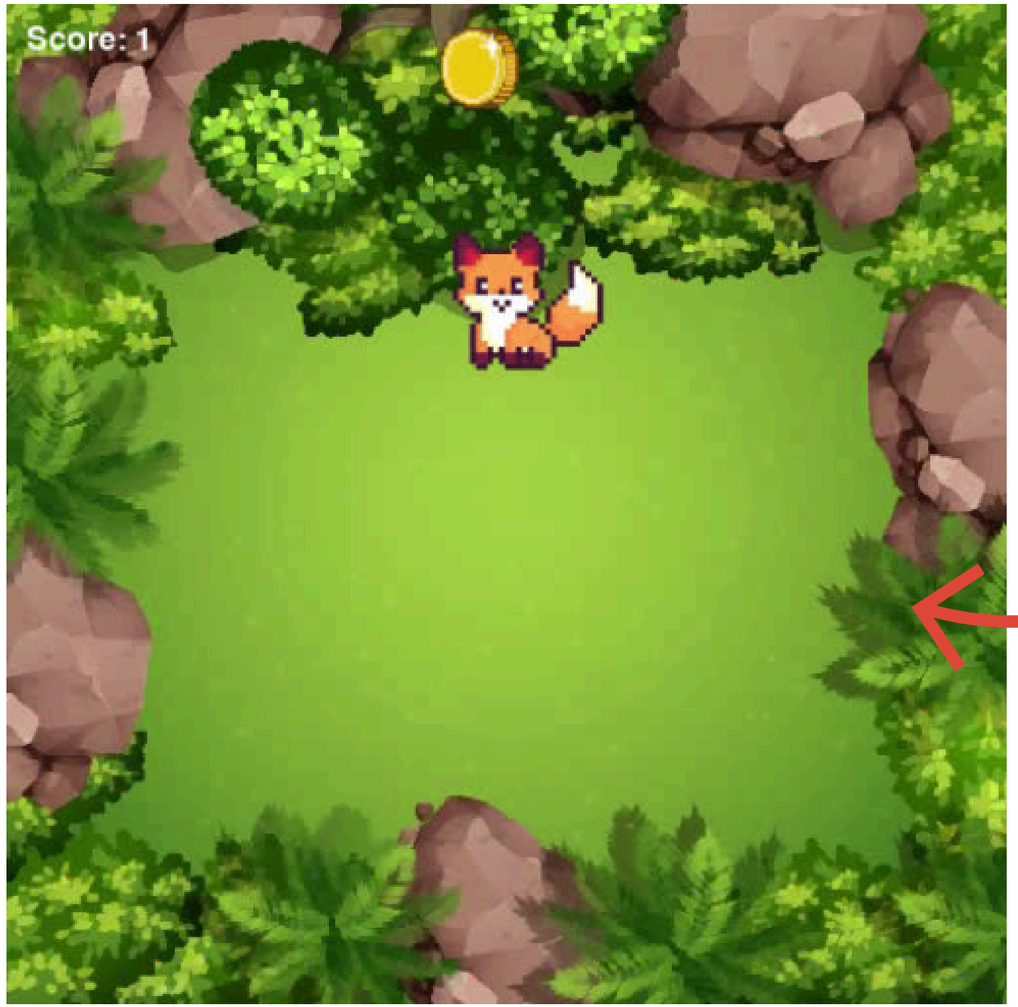
```
# Under WIDTH = 500 HEIGHT = 500 define the following.
score = 0

# Modify the draw function.
def draw():
  global score
 #The previous code for drawing the screen
  screen.draw.text("Score: " + str(score),(10,10))

# Modify the update function with the yellow line.
def update():
 global score
 #The previous code for keyboard control
 if player.colliderect(coin):
  coin.pos = randint(0, 500), randint(0, 500)
  score += 1
```



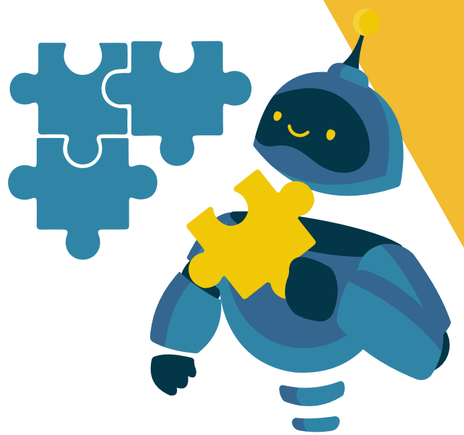
VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#6

## OUTPUT:




**Now we can count the number of coins our fox has collected and show the score**



# COLLISION DETECTION

VORTEX ACADEMY
THE ART OF THINKING

In order to make our actors interact with each other we have to detect when they touch “Collision” which is why we use collision detection.

The function *“ACTOR1_NAME”*.**colliderect**(*“ACTOR2_NAME”*) allows us to detect wether *“ACTOR1_NAME”* and *“ACTOR2_NAME”* are touching which is what we call collision detection.

In the previous example we used **if** player.**colliderect**(*coin*): to move the coin to a new random location when the fox touches it.

---

**Syntax:**

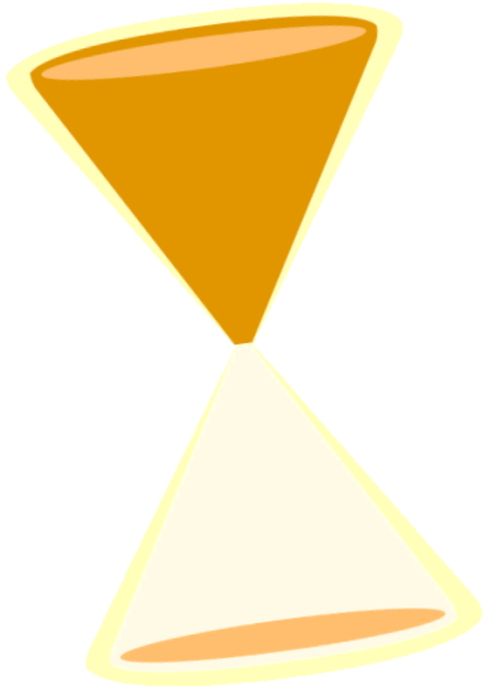
```
def update():
    if “ACTOR1_NAME”.colliderect(“ACTOR2_NAME”):
        # The actions we want to happen when “ACTOR1_NAME” and
          “ACTOR2_NAME” are touching.
```

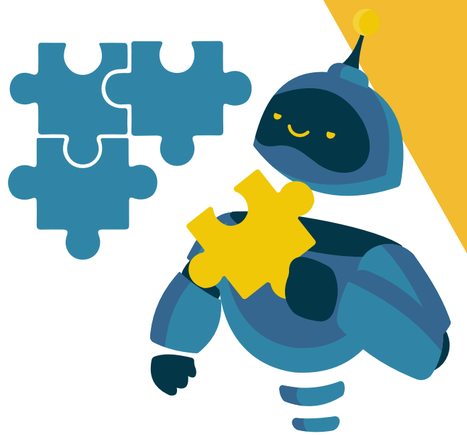
VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#7

## Modify Your Previous Code & Try:

```
# Under player = Actor  coin = Actor Add

def timeUp( ):
  print("Time's up!")

clock.schedule(timeUp, 10)
```

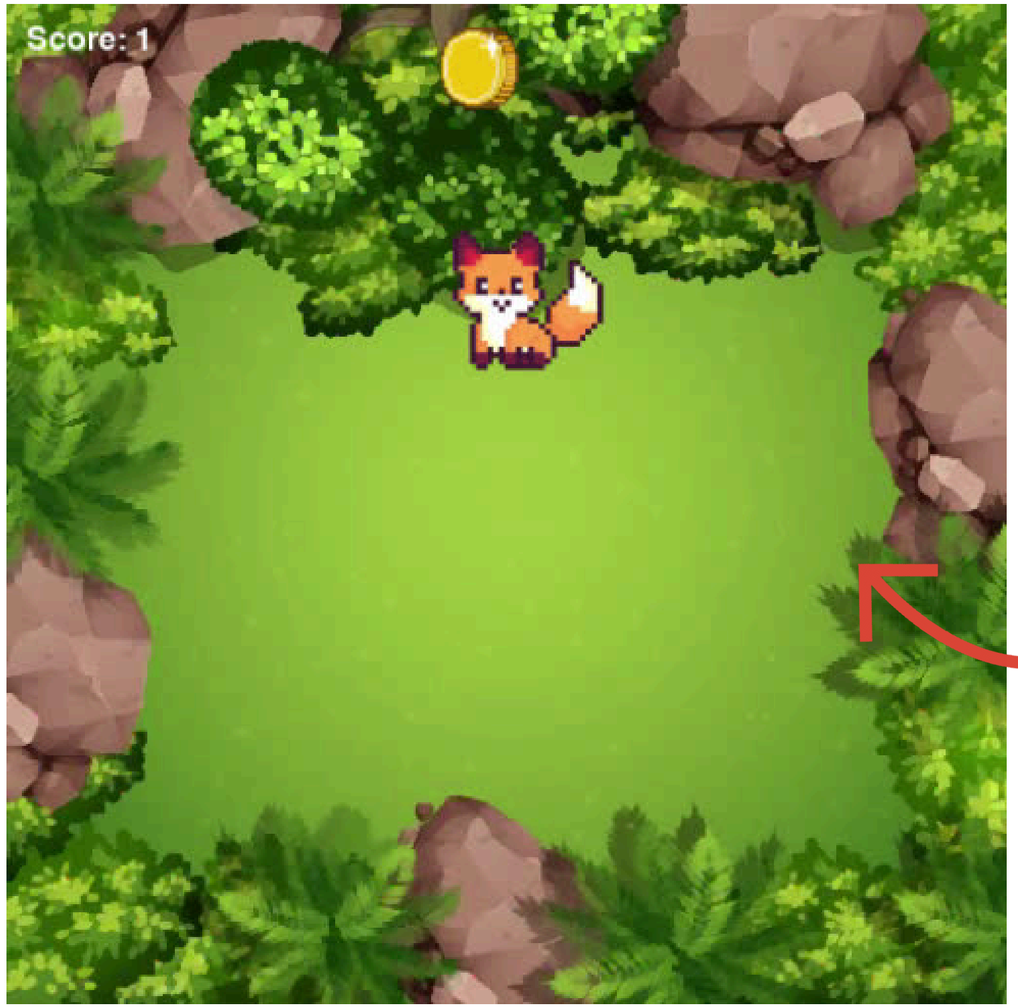




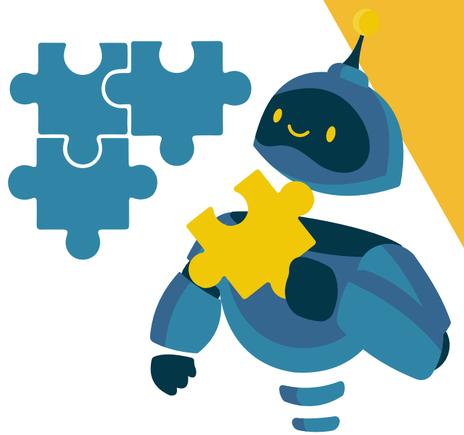
VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#7

## OUTPUT:




**After 10 seconds “Time’s up” will be printed to the terminal**



# CLOCK

Sometimes we want our game to do some actions after a certain amount of time e.g. respawning collectibles, regenerating player health or creating more enemies.

The clock.**schedule(***"FUNCTION_NAME","DELAY_TIME"***)** function allows us to tell our game to do the function *"FUNCTION_NAME" after "DELAY_TIME" has passed.*

In the previous example we used clock.**schedule**(*timeUp, 10*) to tell our game to do the function **timeUp( )** after *10* seconds.

---

**Syntax:**

```
def "FUNCTION_NAME":
    # The function we want to happen after "DELAY_TIME"

clock.schedule("FUNCTION_NAME", "DELAY_TIME")
```

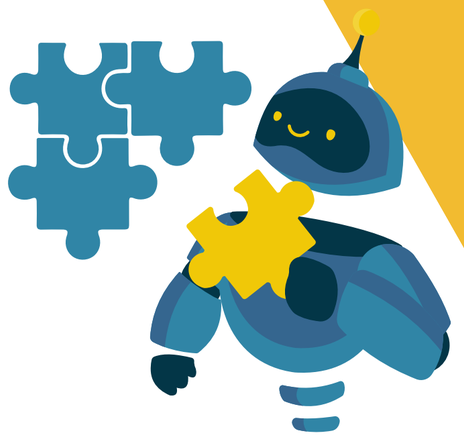
VORTEX ACADEMY
THE ART OF THINKING

VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#8

## Modify Your Previous Code & Try:

```
#Add to the game variables
gameOver = False

#Modify in def timeUp()
def timeUp():
  global gameOver
  gameOver = True

clock.schedule(timeUp, 10)

# Modify in def draw
def draw():
  global gameOver, score
  if not gameOver:
 #Previous code to draw the screen

  else :
    screen.clear()
    screen.fill("maroon")
    screen.draw.text("GAME OVER", (200, 200))
    screen.draw.text("Score: " + str(score), (200, 250))
```

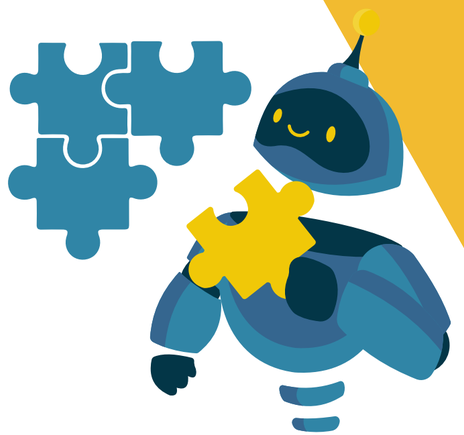


VORTEX ACADEMY
THE ART OF THINKING

# ACTIVITY#8

## OUTPUT:


Now after 10 seconds the screen turns red and "GAME OVER" is displayed along with the score



# COIN COLLECTOR

VORTEX ACADEMY
THE ART OF THINKING

Now we have completed the game COIN COLLECTOR, congrats on finishing your second game.

What's next? what additions do you want to add to the game?
Can you make the game use WASD instead of the arrow keys?

---

**Syntax:**

```
def update():
    if keyboard."KEY_NAME":
        # The actions we want to happen when KEY_NAME is pressed.
```

# ASSIGNMENT

| | | |
|---|---|---|
| 1 | NOTES | Write session notes |
| 2 | SOLVE ASSIGNMENT | |

# ASSIGNMENT

**Continuing on the code that you wrote during the session.**

**Task1:** Modify your code so that it displays in the top left corner of the screen the time left until the game is over.

#HINT make use of the clock.schedule() function to count time.

**Task2:** Continue on your code so that it gives the player an extra second of time when a coin is collected.

**Task3:** Continue on your code so that it makes the player have a speed of 4 for one second when a coin is collected.

#HINT make use of the clock.schedule() function to count time.

VORTEX ACADEMY
THE ART OF THINKING

# THANK YOU

## FOR YOUR ATTENTION