GAME #4

“WHO WANTS TO BE A MILLIONAIRE?”

VORTEX ACADEMY
THE ART OF THINKING

VORTEX ACADEMY
THE ART OF THINKING

WELCOME BACK!

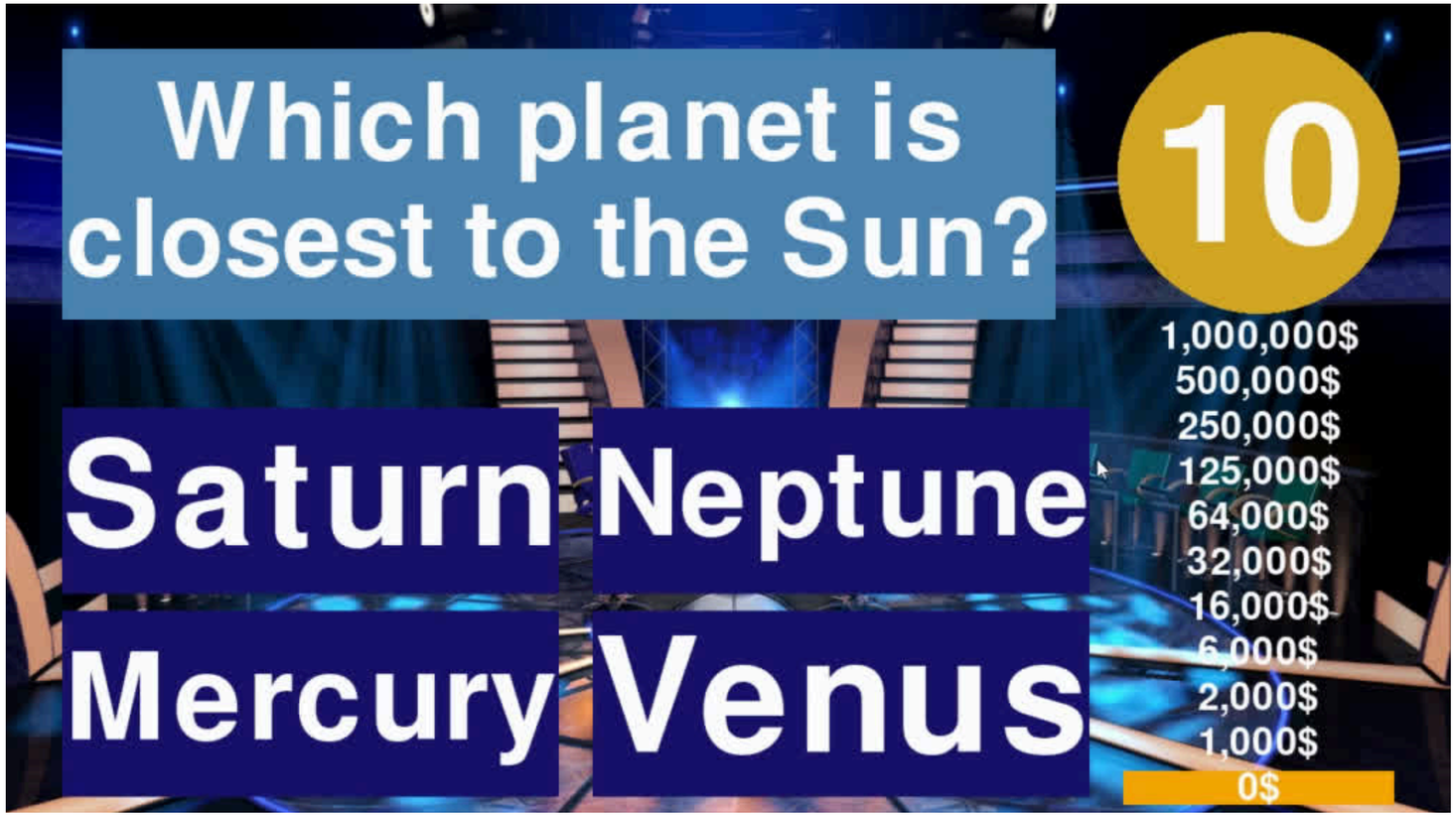
CAN YOU GUESS WHAT WE'LL CREATE TODAY?

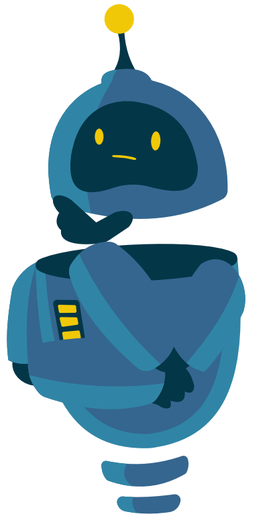
VORTEX ACADEMY
THE ART OF THINKING

HOW DO WE MAKE WHO WANTS TO BE A MILLIONAIRE?

VORTEX ACADEMY
THE ART OF THINKING

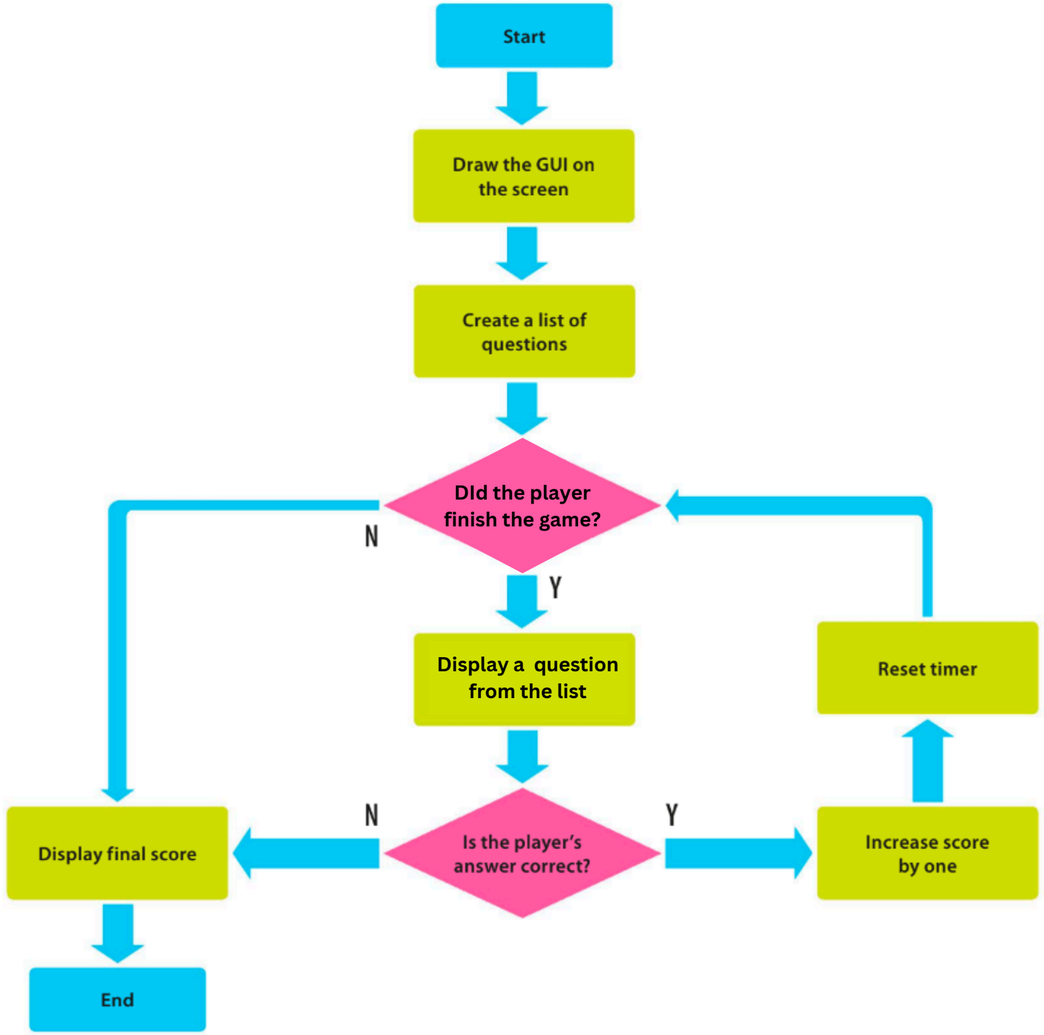
Using what you've learned from the previous sessions, how do you think we could build Who Wants to be a Millionaire?

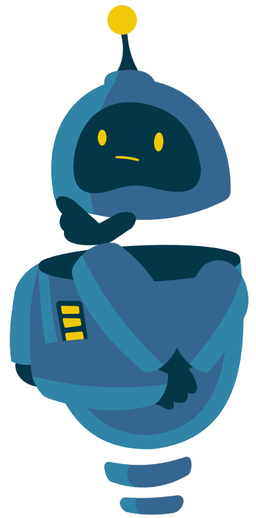




HOW DO WE MAKE WHO WANTS TO BE A MILLIONAIRE?

VORTEX ACADEMY
THE ART OF THINKING





TOPICS

VORTEX ACADEMY
THE ART OF THINKING

01

Screen Elements & GUI design

- How do we draw rectangles and circles on the screen?
- How do we create the game GUI?

02

Enumerate & More Clock Options

- How do we get the items and their index in our for loop at the same time.
- More clock functions

03

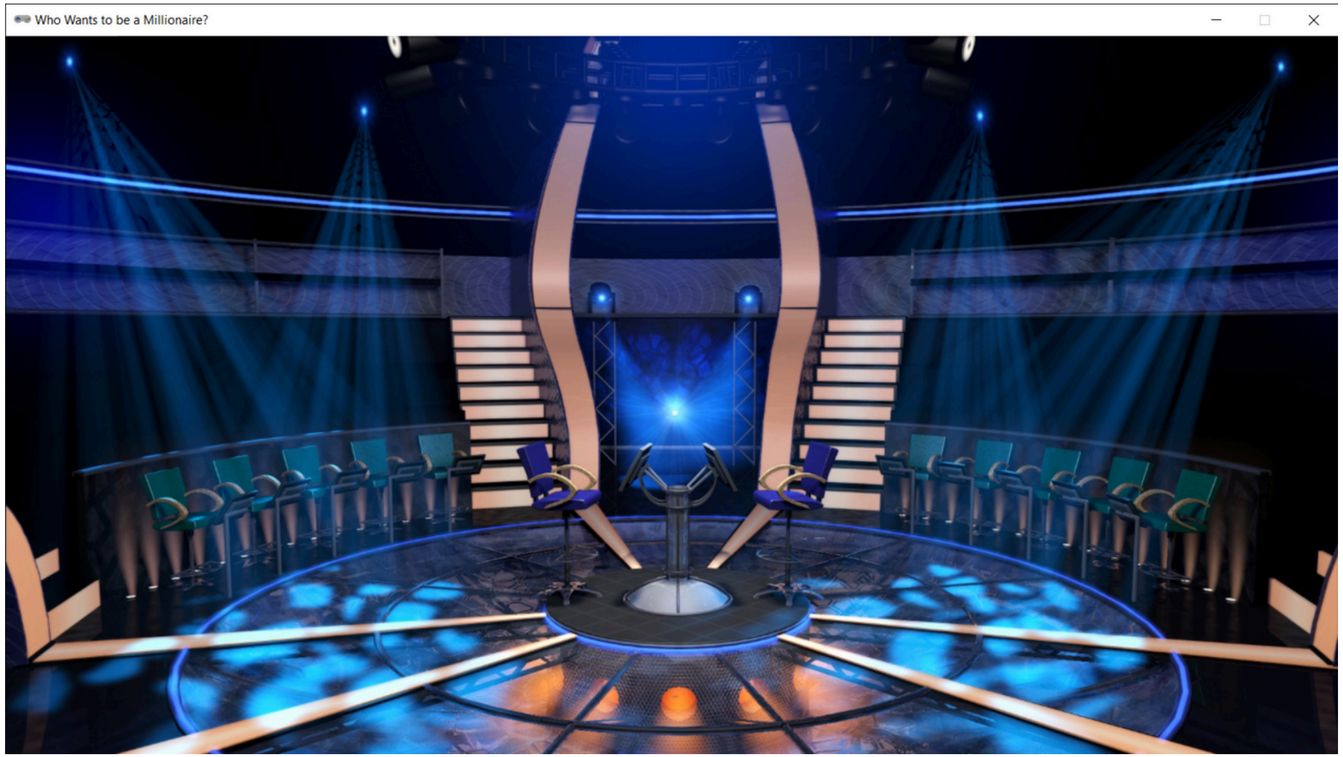
File Handling & Game Logic

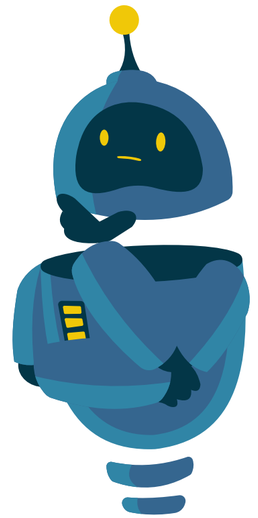
- How do we read and work with files in python?
- Building Who Wants to be a Millionaire?

LET'S START

Use what you've learned to make the background **1280 * 720** the title: *Who Wants to be a Millionaire* then set the show image as the background.

You can look at your previous code and copy from it.





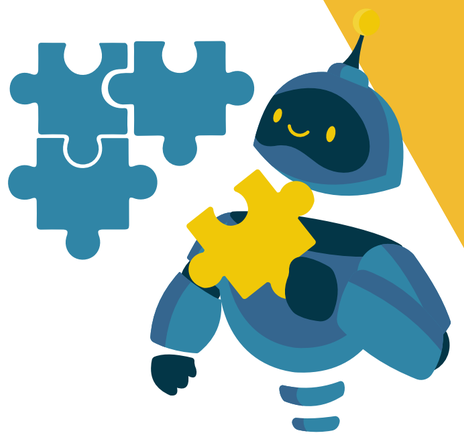
VORTEX ACADEMY
THE ART OF THINKING

VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#1.1

Try the following code

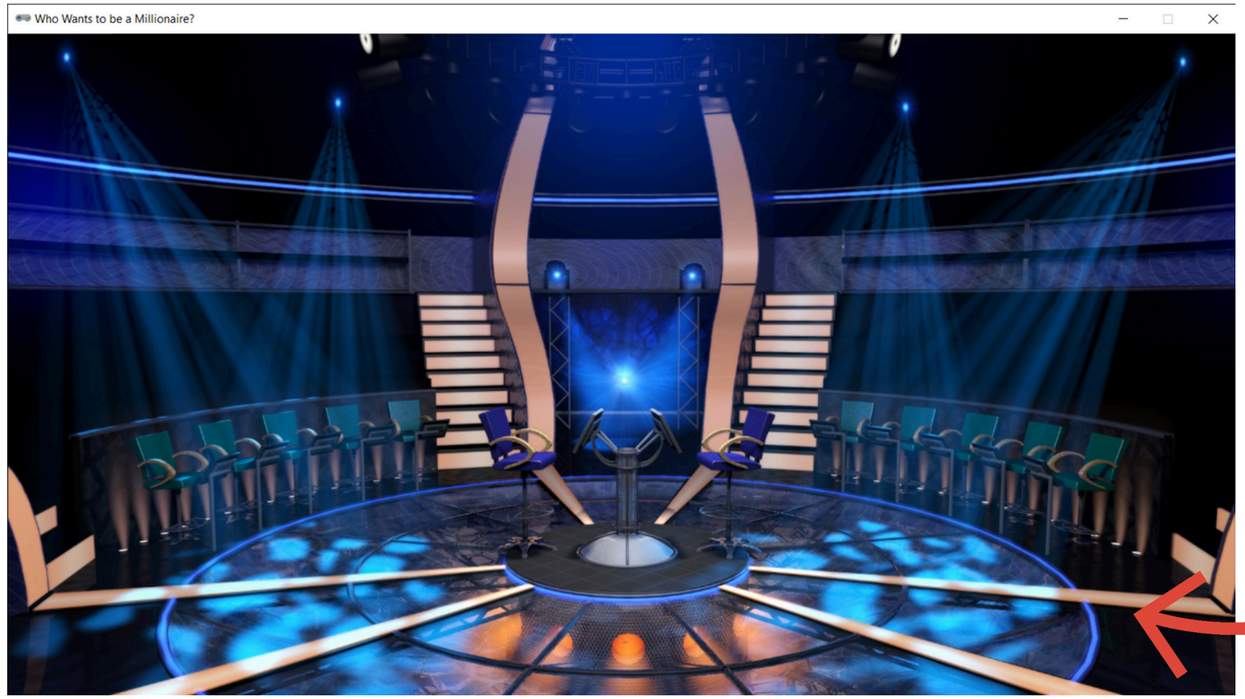
```
import pgzrun
import random

WIDTH = 1280
HEIGHT = 720
TITLE = "Who Wants to be a Millionaire?"

def draw():
    screen.clear()
    screen.blit("background", pos = (0,0))

def update():
    pass

pgzrun.go()
```



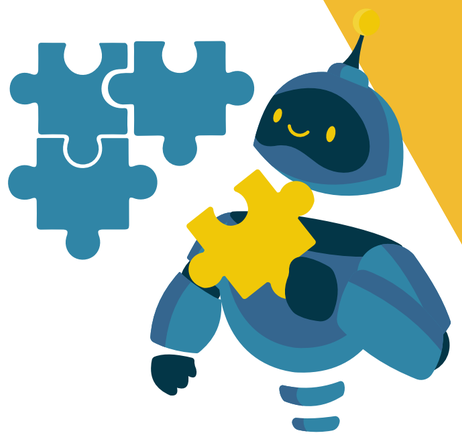
VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#1.1

OUTPUT:



Now we have setup the base for our game, but how do we make the START screen?

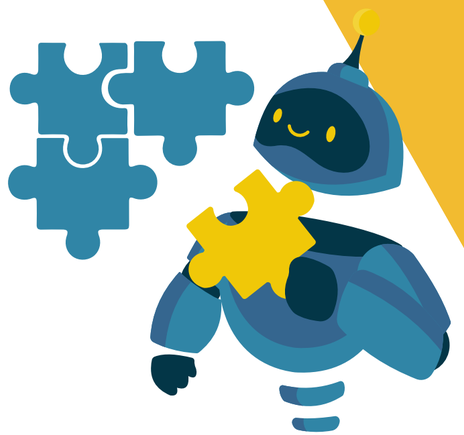


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#1.2

Modify Your Previous Code & Try:

```
# Add to the game variables.
gameStarted = False
# Modify the draw function by adding the yellow lines.
def draw():
 screen.clear()
 screen.blit("background", pos = (0,0))
 if gameStarted:
  pass
 else:
  screen.draw.text("Start?", center = (WIDTH/2, HEIGHT/2),
  fontsize = 400, color = "white")

# Add this function.
def on_mouse_down(pos):
 global gameStarted
 if not gameStarted:
  gameStarted = True
```

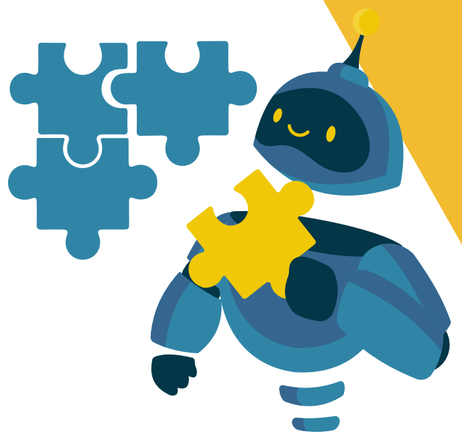


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#1.2

OUTPUT:

Now we have setup the start screen correctly.

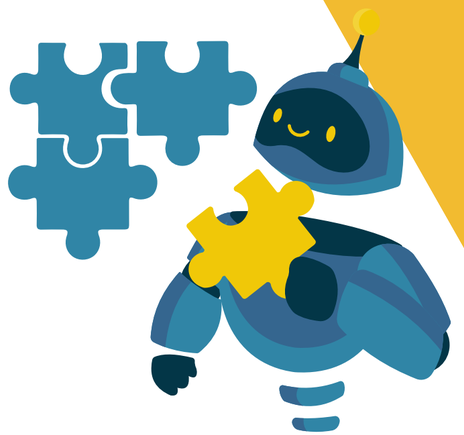


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#2.1

Try in another script the following code

```
import pgzrun

WIDTH = 1280
HEIGHT = 720
TITLE = "Who Wants to be a Millionaire?"

box1 = Rect((0, 0), (100, 200))
box2 = Rect((200, 0), (100, 200))

def draw():
    screen.fill("slate gray")
    screen.draw.rect(box1, "maroon")
    screen.draw.filled_rect(box2, "steel blue")

def update():
    pass

pgzrun.go()
```

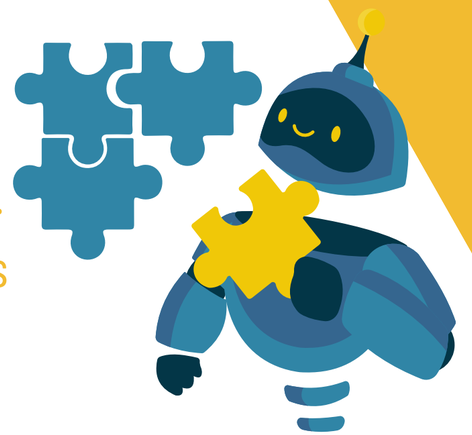


ACTIVITY#2.1

OUTPUT:

The two rectangles are drawn on the screen at the locations we specified

#HINT: try changing the rectangle colors by changing the color passed to the draw.**rect()** function.
try changing the top left and bottom right corners of the rectangles by changing the coordinates given in box = **Rect**((X1,Y1),(X2,Y2)).
try adding another filled rectangle that fills the whole screen. (from (0,0) to (1280, 720)).



Rect()

VORTEX ACADEMY
THE ART OF THINKING

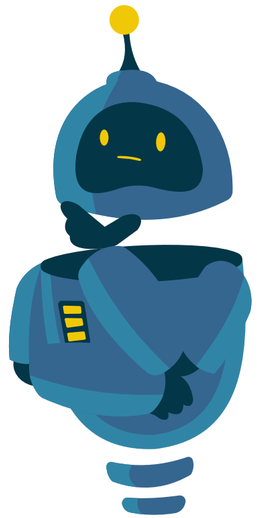
When we want to draw rectangles to the screen we use *RECTANGLE_NAME* = **Rect**(*(LEFT, TOP), (WIDTH, HEIGHT)*) to create the rectangle then screen.draw.**rect**(*RECTANGLE_NAME, RECTANGLE_COLOR*) to draw the rectangle's outline to the screen and screen.draw.**rect_filled**(*RECTANGLE_NAME, RECTANGLE_COLOR*) to draw the filled rectangle to the screen.

In the previous example we used screen.draw.**rect**(*box1, "maroon"*) and screen.draw.**filled_rect**(*box2, "steel blue"*) to draw the maroon rectangle outline and steel blue filled rectangle.

Syntax:

RECTANGLE_NAME = **Rect**(*(LEFT, TOP), (WIDTH, HEIGHT)*)

screen.draw.**rect**(*RECTANGLE_NAME, RECTANGLE_COLOR*)
screen.draw.**rect_filled**(*RECTANGLE_NAME, RECTANGLE_COLOR*)

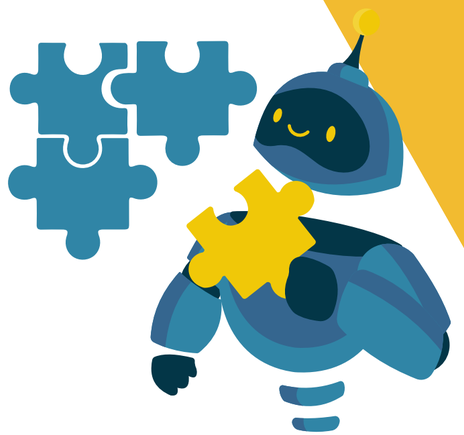


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#2.2

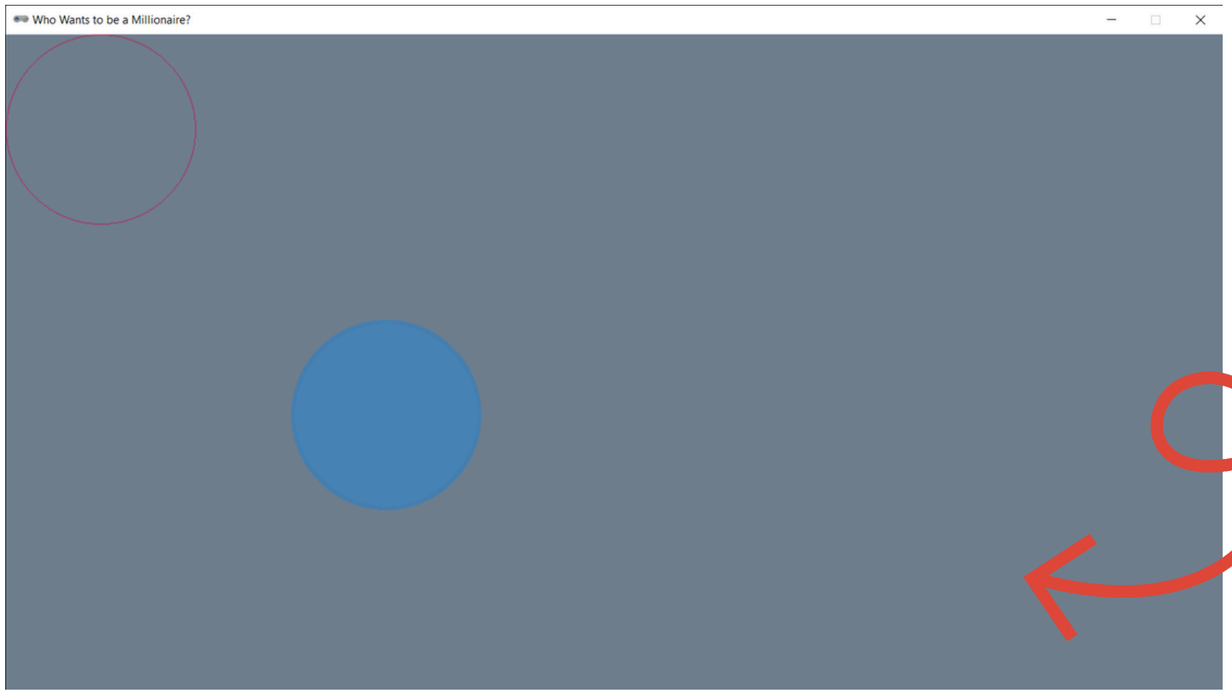
Try in another script the following code

```
import pgzrun

WIDTH = 1280
HEIGHT = 720
TITLE = "Who Wants to be a Millionaire?"

def draw():
    screen.fill("slate gray")
    screen.draw.circle((100,100), 100, "maroon")
    screen.draw.filled_circle((400,400), 100, "steel blue")

def update():
    pass

pgzrun.go()
```



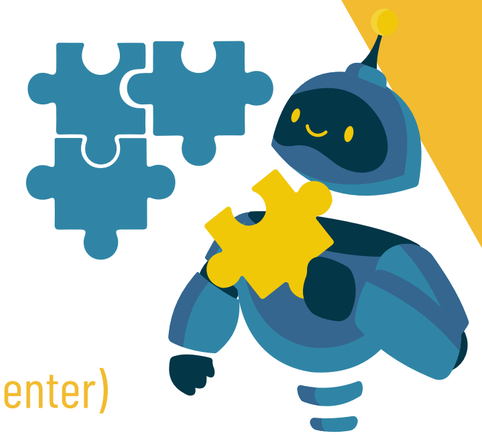
ACTIVITY#2.2

OUTPUT:



The two circles are drawn on the screen at the locations we specified.

#HINT: try changing the circle colors by changing the color passed to the draw.**circle()** function.
try changing the circle radius by changing the number passed to the draw.**circle()** function.
try changing the circle center by changing the tuple passed to the draw.**circle()** function.
try adding another filled circle that fills the whole screen. (radius = 720/2 = 360, center = screen center)



Circles

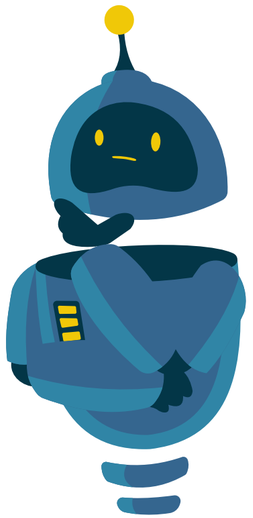
VORTEX ACADEMY
THE ART OF THINKING

When we want to draw circles to the screen we use screen.draw.**circle**((CENTER_X,CENTER_Y), RADIUS, COLOR) to create a circle outline and screen.draw.**filled_circle**((CENTER_X,CENTER_Y), RADIUS, COLOR) to create a filled circle.

In the previous example we used screen.draw.**circle**(*(100,100), 100, "maroon"*) and screen.draw.**filled_circle**(*(400,400), 100, "steel blue"*) to draw the maroon circle outline and steel blue filled circle.

Syntax:

screen.draw.**circle**(*(CENTER_X,CENTER_Y), RADIUS, COLOR*)
screen.draw.**filled_circle**(*(CENTER_X,CENTER_Y), RADIUS, COLOR*)

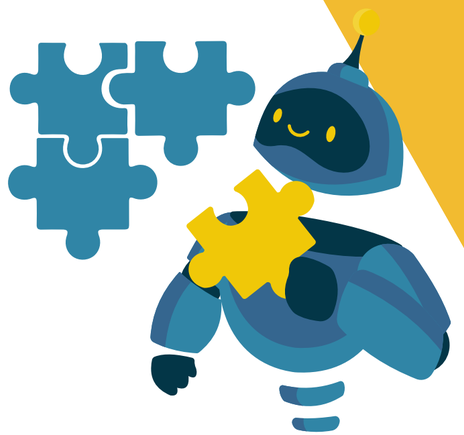


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#2.3

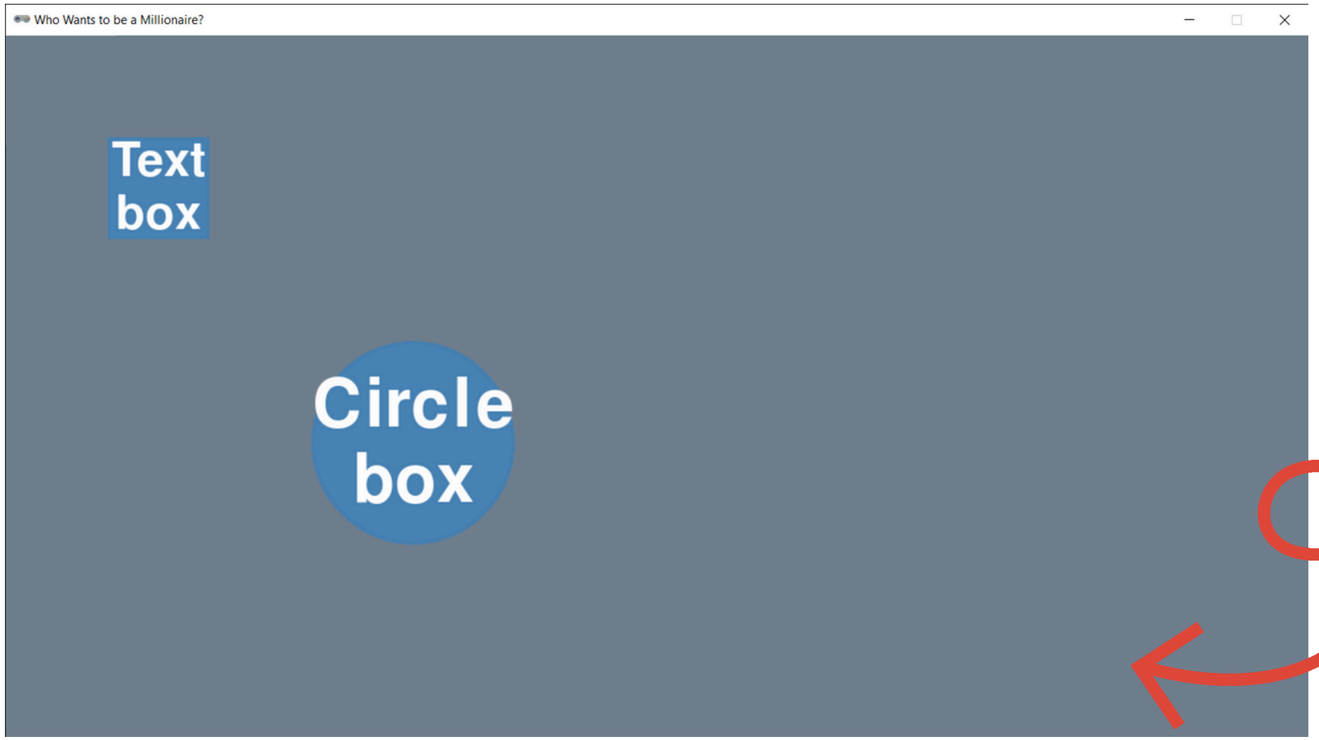
Try in another script the following code

```
import pgzrun
import random

WIDTH = 1280
HEIGHT = 720
TITLE = "Who Wants to be a Millionaire?"

box = Rect((100,100), (100,100))
circle_box = Rect((300,300), (200,200))

def draw():
    screen.fill("slate gray")
    screen.draw.filled_rect(box, "steel blue")
    screen.draw.filled_circle((400,400), 100, "steel blue")
    screen.draw.textbox("Text box", box, color="white")
    screen.draw.textbox("Circle box", circle_box, color="white")

def update():
    pass

pgzrun.go()
```



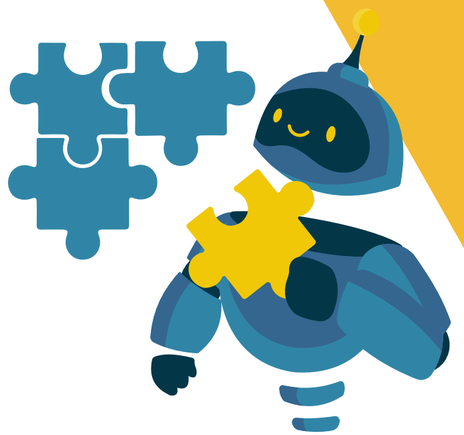
ACTIVITY#2.3

OUTPUT:



The two text boxes wre drawn where we specified on top of the rectangle and circle.

#HINT: try changing the text colors by changing the color passed to the draw.**textbox()** function.
try changing the text by changing the string passed to the draw.**textbox()** function.
try changing the size of the textbox by changing the Rect passed to the draw.**textbox()** function.
try adding another textbox.



Text boxes

VORTEX ACADEMY
THE ART OF THINKING

When we want to draw text boxes to the screen we use
RECTANGLE_NAME = **Rect**(*(LEFT, TOP), (WIDTH, HEIGHT)*) to create the rect for the textbox then screen.draw.**textbox**(“TEXT”, *RECTANGLE_NAME, TEXT_COLOR*).

In the previous example we used screen.draw.**textbox**(*"Text box", box, color="white"*) and screen.draw.**textbox**(*"Circle box", circle_box, color="white"*) to draw the text on the rectangle and the circle.

Syntax:

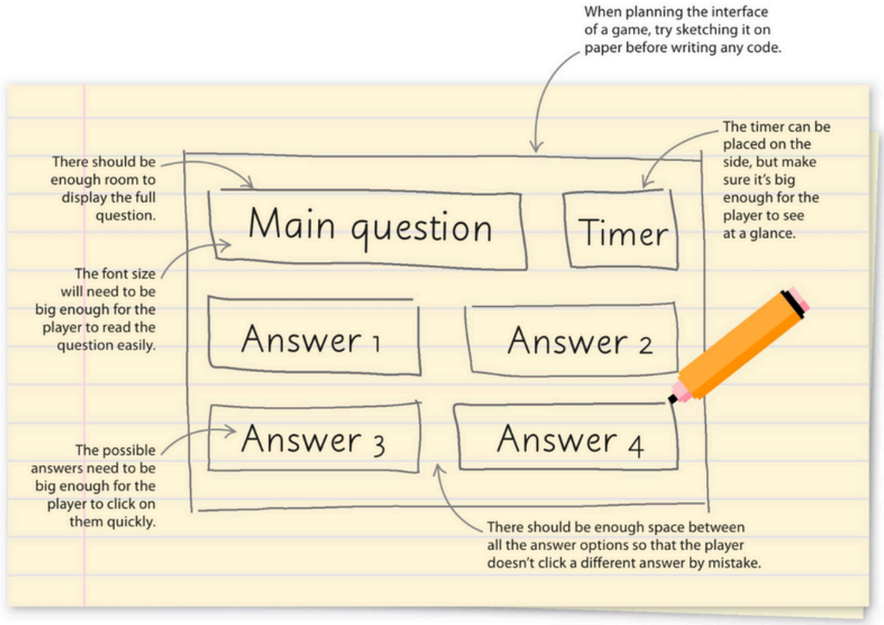
RECTANGLE_NAME = **Rect**(*(LEFT, TOP), (WIDTH, HEIGHT)*)
screen.draw.**textbox**(*“TEXT”, RECTANGLE_NAME, TEXT_COLOR*).

GUI Design

VORTEX ACADEMY
THE ART OF THINKING

When we want to design the Graphical User Interface (GUI) of a program or game we start with sketches of the program either on paper or applications like figma before starting the code implementation.

Can you come up with a sketch for the game?

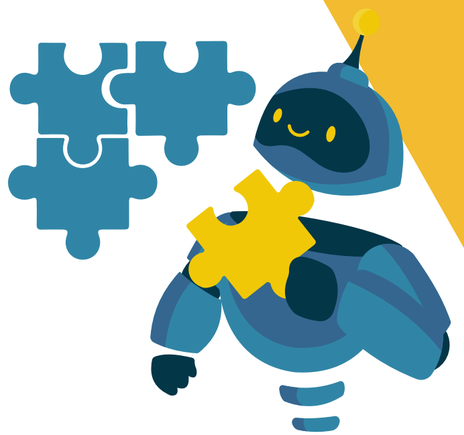


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#3.1

Try in another script the following code

```
list = ["one", "two", "three", "four", "five"]

for index, item in enumerate(list):
    print(index, item)

colors = ["red", "green", "blue", "yellow", "purple"]

for index, color in enumerate(colors):
    print(index, color)

boxes = ["box1", "box2", "box3", "box4", "box5"]

for index, box in enumerate(boxes):
    print(index, box)
```



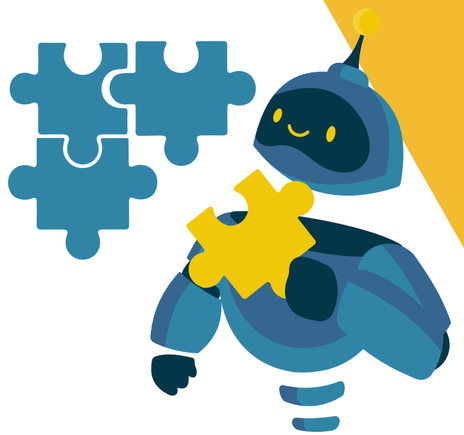
VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#3.1

OUTPUT:

```
0 one
1 two
2 three
3 four
4 five
0 red
1 green
2 blue
3 yellow
4 purple
0 box1
1 box2
2 box3
3 box4
4 box5
```

enumerate(LIST)
allows us to gain
access to the elements
and their index inside
the for loop



VORTEX ACADEMY
THE ART OF THINKING

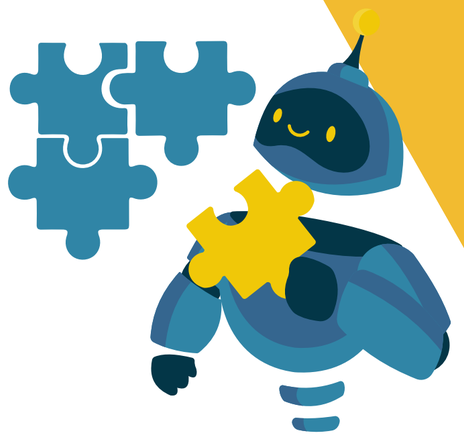
ACTIVITY#3.2

To Continue The Game:

How can we use rectangles and text boxes to add the question box to our game?

Create a rect for the main box then draw it to the screen colored “steel blue” if the game has started, and a text box on top of it to contain the questions.

#Hint: use values (50, 40, 880, 240) for the rectangle.

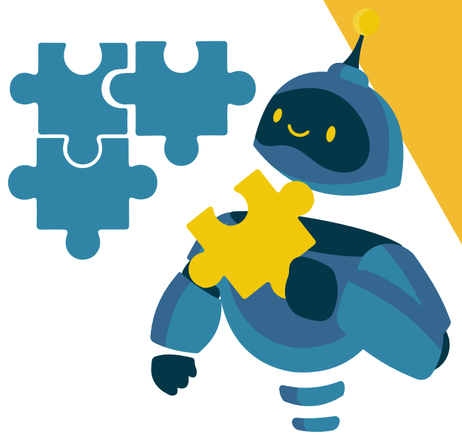


ACTIVITY#3.2

VORTEX ACADEMY
THE ART OF THINKING

Modify Your Previous Code & Try:

```
# Add to the GUI variables below game variables
main_box = Rect(50, 40, 880, 240)
# Modify the draw function with the yellow lines.
def draw():
  screen.clear()
  screen.blit("background", pos = (0,0))
  if gameStarted:
    screen.draw.filled_rect(main_box, "steel blue")
    screen.draw.textbox("question", main_box, color = "white")
  else:
    screen.draw.text("Start?", center = (WIDTH/2, HEIGHT/2),
    fontsize = 400, color = "white")
```

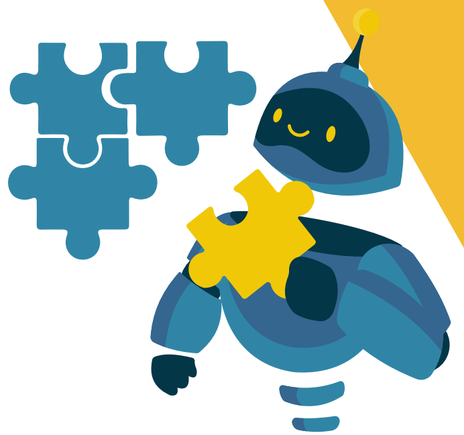


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#3.2

OUTPUT:

The question box has been placed.



VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#3.3

To Continue Our Game:

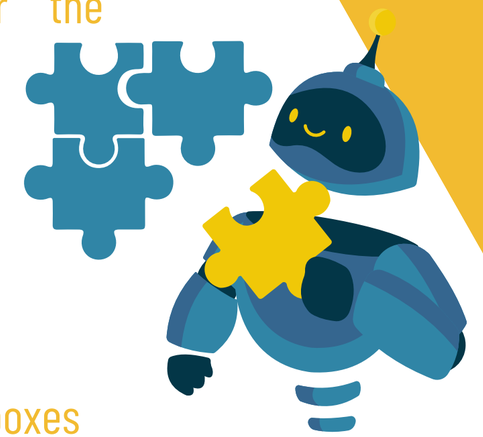
How can we use rectangles and text boxes to add the answer boxes to our game?

Define the answer box rectangles in a list then draw them using a for loop with enumerate() drawing filled rectangles then textboxes.

#Hint: use the following values for the rectangles.

(50, 358, 440, 165),
(520, 358, 440, 165),
(50, 538, 440, 165),
(520, 538, 440, 165)

use the index from enumerate() in the textboxes

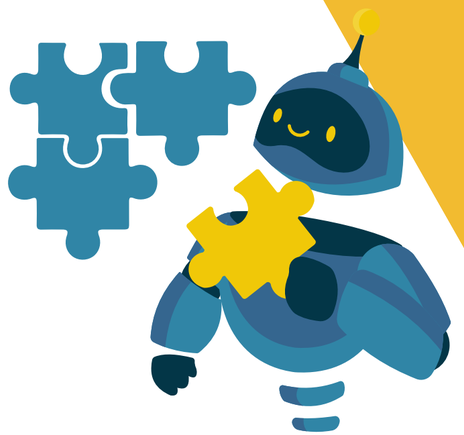


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#3.3

Modify Your Previous Code & Try:

```
# Add to the GUI variables
answer_boxes = [Rect(50, 358, 440, 165),
                Rect(520, 358, 440, 165),
                Rect(50, 538, 440, 165),
                Rect(520, 538, 440, 165)]
 # Add to the draw() function in the if condition where the game has started.
for index, box in enumerate(answer_boxes):
    screen.draw.filled_rect(box, "midnight blue")
    screen.draw.textbox("answer"+str(index), box, color = "white")
```

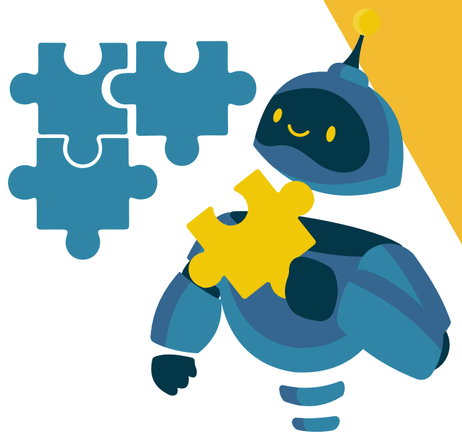


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#3.3

OUTPUT:

Now the answer boxes are drawn correctly.



VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#3.4

To Continue Our Game:

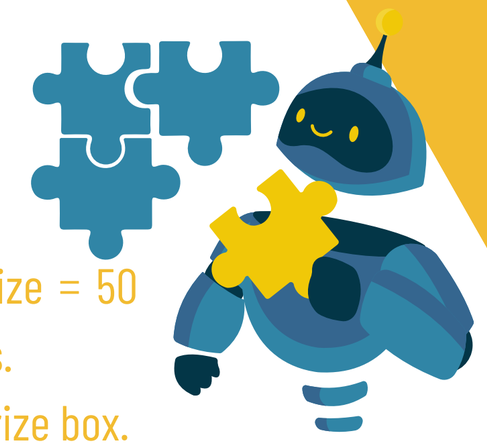
How can we use text and rectangles to add the prizes and prize box to the game?

Create the prize box and a list the contains the values of the prizes then write them to the screen with the box starting behind 0$.

#Hint: the prizes are:
"0$","1,000$", "2,000$", "6,000$",
"16,000$", "32,000$", "64,000$",
"125,000$", "250,000$", "500,000$",
"1,000,000$"

Use center = (1110, 695-40*counter), fontsize = 50 along with enumerate to position the prizes.

Use the values (990, 680, 240, 30) for the prize box.

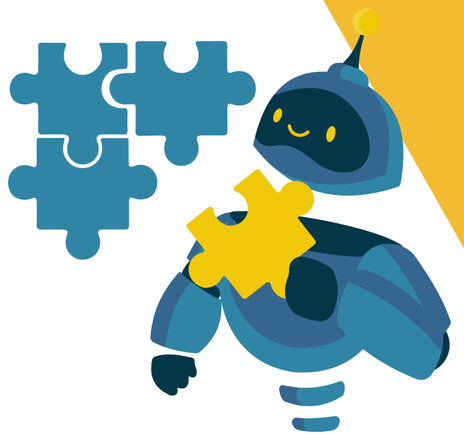


ACTIVITY#3.4

VORTEX ACADEMY
THE ART OF THINKING

Modify Your Previous Code & Try:

```
# Add to the GUI variables.
prize_box = Rect(990, 680, 240, 30)
prizes = ["0$","1,000$", "2,000$", "6,000$",
          "16,000$", "32,000$", "64,000$",
          "125,000$", "250,000$", "500,000$",
          "1,000,000$"]
# Add to the draw() function in the if condition
where the game has started:
screen.draw.filled_rect(prize_box, "orange")
for counter, prize in enumerate(prizes):
    screen.draw.text(prize, center = (1110, 695-
40*counter), fontsize = 50, color = "white")
```

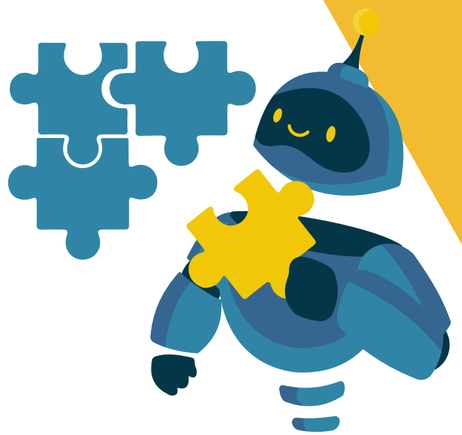


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#3.4

OUTPUT:

Now the prizes are positioned correctly.

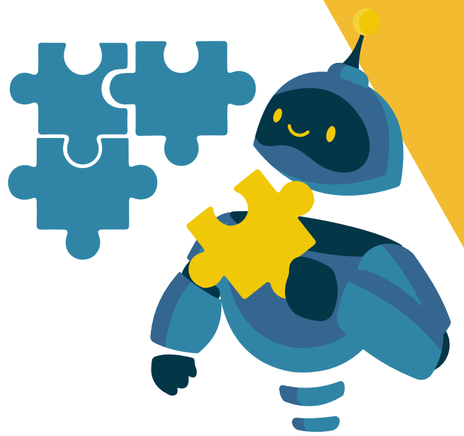


ACTIVITY#4.1

VORTEX ACADEMY
THE ART OF THINKING

Try in another script the following code, then try clicking multiple times.

```
import pgzrun
import random

WIDTH = 1280
HEIGHT = 720
TITLE = "Who Wants to be a Millionaire?"

def sayHi1():
    print("Hello1")
    clock.schedule(sayHi1, 1)

def sayHi2():
    print("Hello2")
    clock.schedule(sayHi2, 1)

clock.schedule(sayHi1, 1)

def draw():
    pass

def on_mouse_down(pos):
    clock.unschedule(sayHi1)
    clock.schedule(sayHi2, 1)

def update():
    pass

pgzrun.go()
```



ACTIVITY#4.1

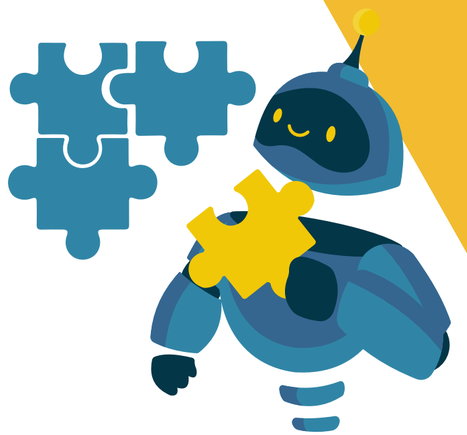
VORTEX ACADEMY
THE ART OF THINKING

OUTPUT:

Hello1
Hello1
Hello1
Hello1
Hello2
Hello2
Hello2
Hello2
Hello2
Hello2

Hello1 is printed to the terminal every second. After clicking Hello2 is printed every second.

If you click multiple times Hello2 will be printed faster and faster.
Why do you think that happens?

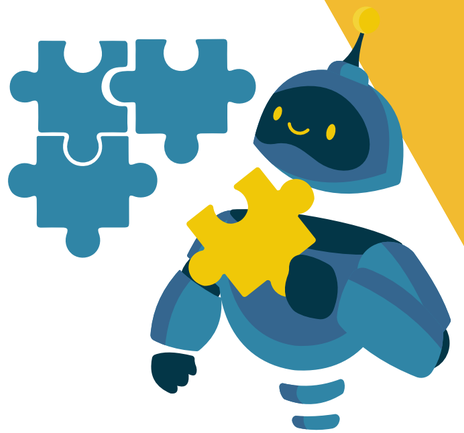


ACTIVITY#4.2

VORTEX ACADEMY
THE ART OF THINKING

Modify in the previous slide's code, then try clicking multiple times.

```
import pgzrun
import random

WIDTH = 1280
HEIGHT = 720
TITLE = "Who Wants to be a Millionaire?"

def sayHi1():
    print("Hello1")
    clock.schedule(sayHi1, 1)

def sayHi2():
    print("Hello2")
    clock.schedule(sayHi2, 1)

clock.schedule(sayHi1, 1)

def draw():
    pass

def on_mouse_down(pos): #Replace the lines in yellow.
    clock.unschedule(sayHi1)
    clock.schedule_unique(sayHi2, 1)

def update():
    pass

pgzrun.go()
```



ACTIVITY#4.2

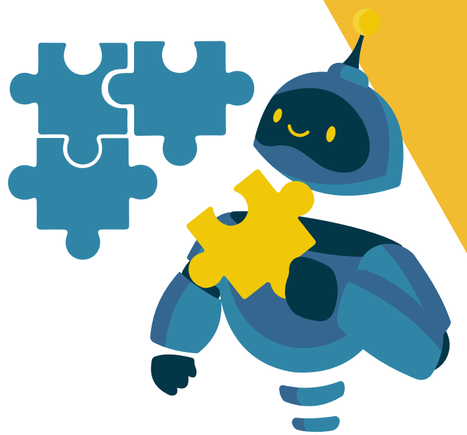
VORTEX ACADEMY
THE ART OF THINKING

OUTPUT:

Hello1
Hello1
Hello1
Hello1
Hello2
Hello2
Hello2
Hello2
Hello2
Hello2

Hello1 is printed to the terminal every second. After clicking Hello2 is printed every second.

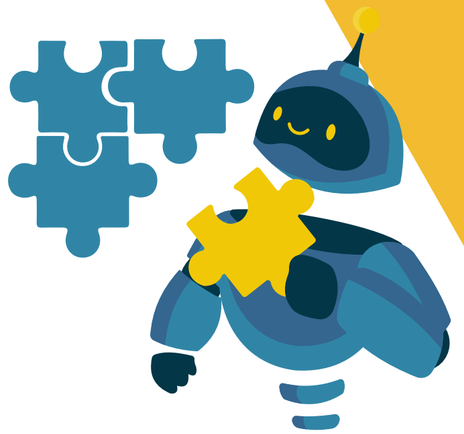
If you click multiple times Hello2 will be printed at the same speed.
Why do you think that is?



ACTIVITY#4.3

Modify in the previous slide's code

```
import pgzrun
import random

WIDTH = 1280
HEIGHT = 720
TITLE = "Who Wants to be a Millionaire?"

def sayHi1():  #Replace the lines in yellow.
    print("Hello1")
def sayHi2():
    print("Hello2")

clock.schedule_interval(sayHi1, 1)

def draw():
    pass

def on_mouse_down(pos): #Replace the lines in yellow.
    clock.unschedule(sayHi1)
    clock.schedule_interval(sayHi2, 1)

def update():
    pass

pgzrun.go()
```



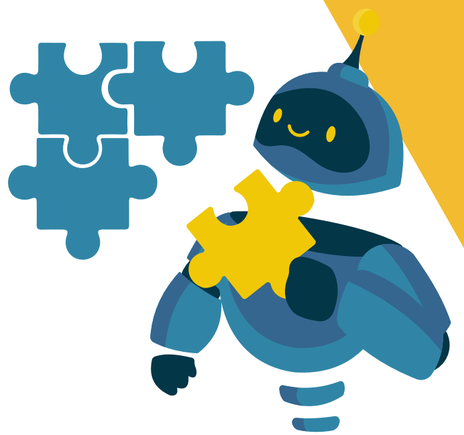
VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#4.3

OUTPUT:

Hello1
Hello1
Hello1
Hello1
Hello2
Hello2
Hello2
Hello2
Hello2
Hello2

Hello1 is printed to the terminal every second. After clicking Hello2 is printed every second.



More Clock Options

VORTEX ACADEMY
THE ART OF THINKING

In order to be able to time the events in the games more accurately and effectively we have a few other clock options which are:

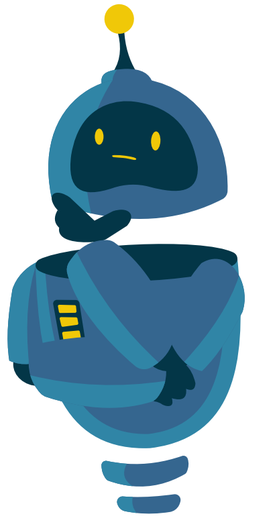
clock.**unschedule**(*FUNCTION_NAME*): to allow us to unschedule an already scheduled event.

clock.**schedule_unique**(*FUNCTION_NAME, TIME*): to allow us to schedule a function to run after TIME has passed such that if the **schedule_unique**() is called on the same function again it wouldn't schedule it twice.

clock.**schedule_interval**(*FUNCTION_NAME, TIME*): to allow us to schedule a function to run every TIME seconds forever.

Syntax:

clock.**unschedule**(*FUNCTION_NAME*)
clock.**schedule_unique**(FUNCTION_NAME, TIME)
clock.**schedule_interval**(FUNCTION_NAME, TIME)



VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#4.4

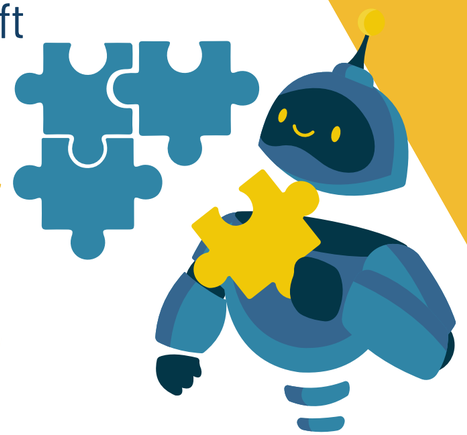
To Continue Our Game:

How can we use clock.schedule_interval() with circles and text to make our timer?

Create a variable to store time left, a function to updateTime that decreases time by 1 if the game has started and time > 0, and calles game_over when the time reaches 0. then schedule_interval(updateTime, 1.0) and draw a filled circle to the screen with the time left as text on it.

#Hint: use ((1110, 150), 125, "golden rod") for the circle.

use (str(timeLeft), center = (1110, 160), fontsize = 250, color = "white") for the text.

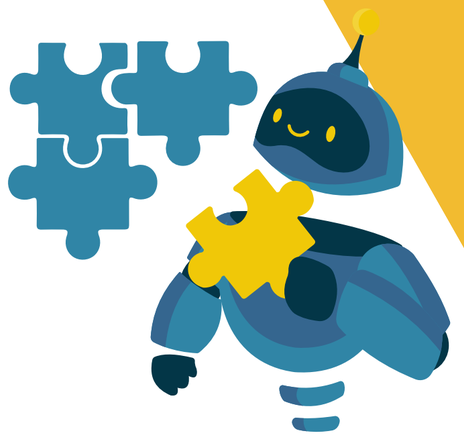


ACTIVITY#4.4

VORTEX ACADEMY
THE ART OF THINKING

Modify Your Previous Code & Try:

```
# Add to the game variables.
timeLeft = 10
# Add this function and the clock scheduling of it.
def updateTime():
    global timeLeft, gameStarted
    if gameStarted and timeLeft > 0:
        timeLeft -= 1
    if timeLeft == 0:
        print("game_over")
clock.schedule_interval(updateTime, 1.0)

#Add to def draw() in the if condition where the game has started.
screen.draw.filled_circle((1110, 150), 125, "golden rod")
screen.draw.text(str(timeLeft), center = (1110, 160), fontsize = 250, color = "white")
```

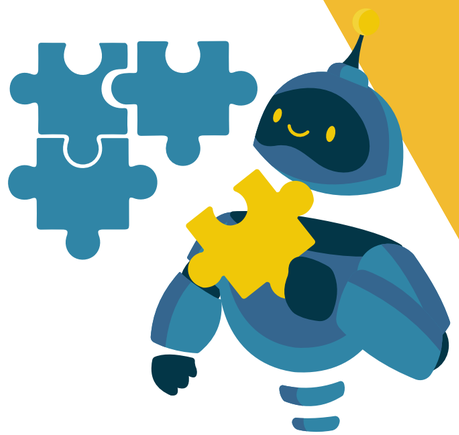


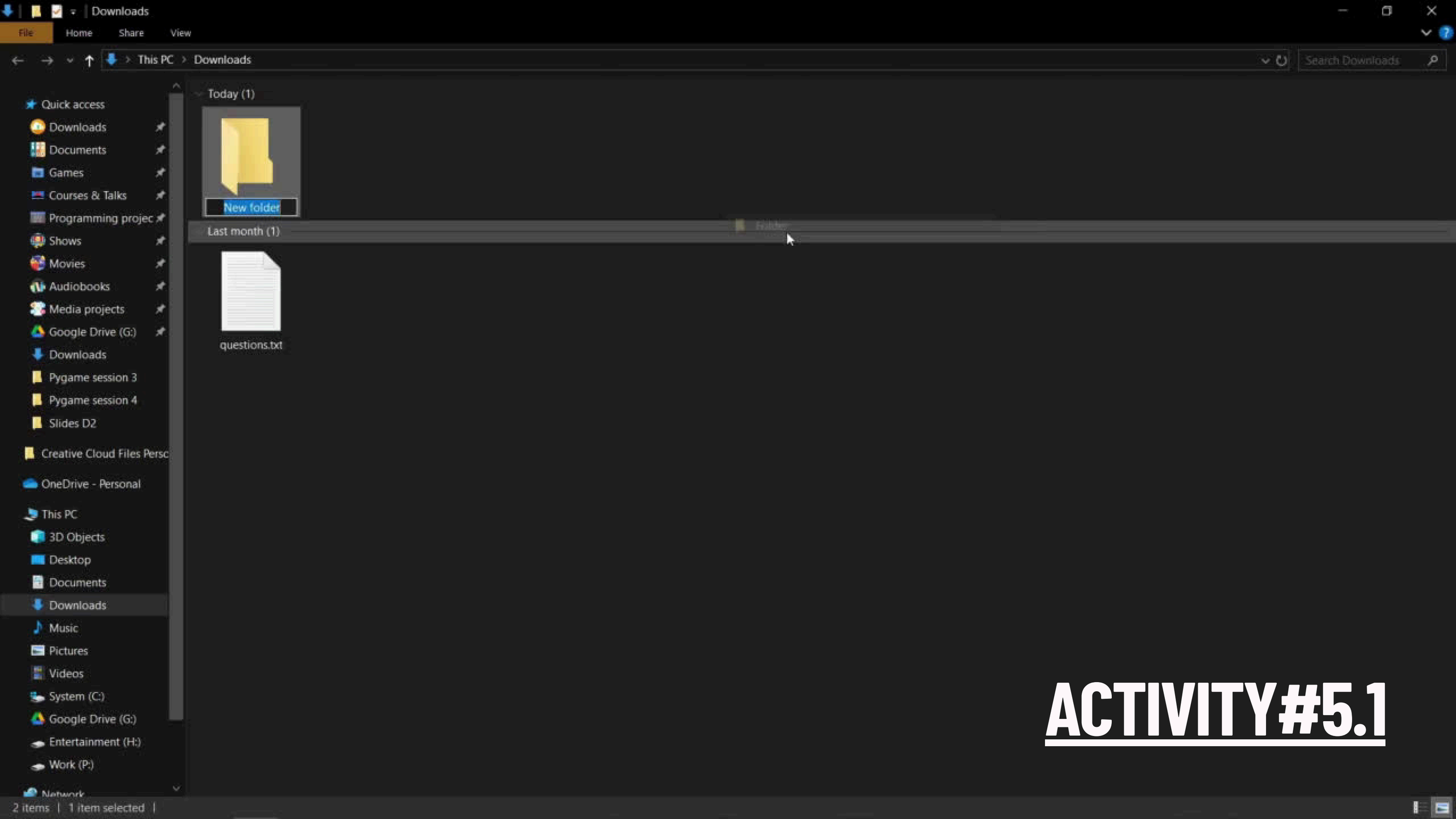
VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#4.4

OUTPUT:

Now the timer works correctly.



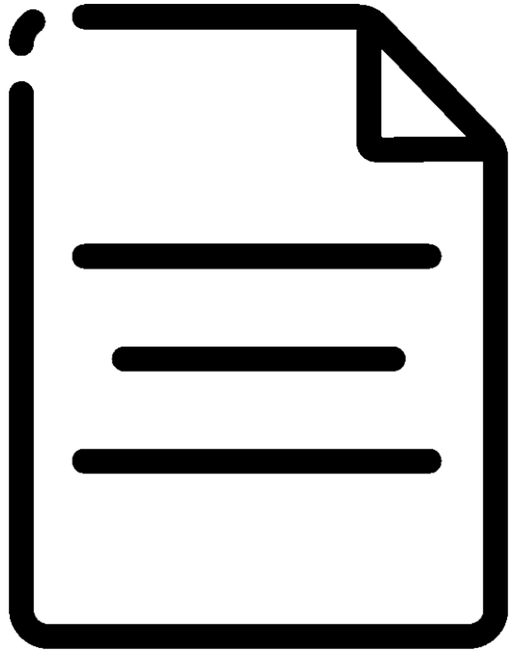

Downloads
File
Home
Share
View
This PC
Downloads
Search Downloads
Quick access
Downloads
Documents
Games
Courses & Talks
Programming projec
Shows
Movies
Audiobooks
Media projects
Google Drive (G:)
Downloads
Pygame session 3
Pygame session 4
Slides D2
Creative Cloud Files Persc
OneDrive - Personal
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
System (C:)
Google Drive (G:)
Entertainment (H:)
Work (P:)
Network
Today (1)
New folder
Last month (1)
questions.txt
2 items | 1 item selected
ACTIVITY#5.1

VORTEX ACADEMY
THE ART OF THINKING

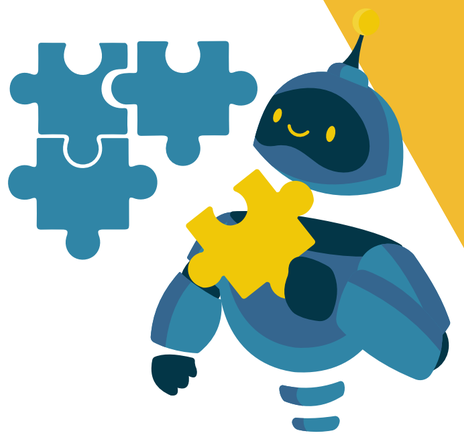
ACTIVITY#5.1

Try in another script the following code

```
file = open(r"questions.txt")
# Replace questions.txt with the path you copied in the previous slide.
print(file.read())
file.close()
#HINT:use right click then copy path on the questions.txt file in vs code to get the path from the script's folder as in the previous slides.
```



#HINT: the r here is to stop python from trying to look for escape sequences as \n or \t and take the string unmodified.



ACTIVITY#5.1

VORTEX ACADEMY
THE ART OF THINKING

OUTPUT:

The text in questions.txt file:

What is the capital of France?,London,Paris,Berlin,Tokyo,1

What is 5+7?,12,10,14,8,0

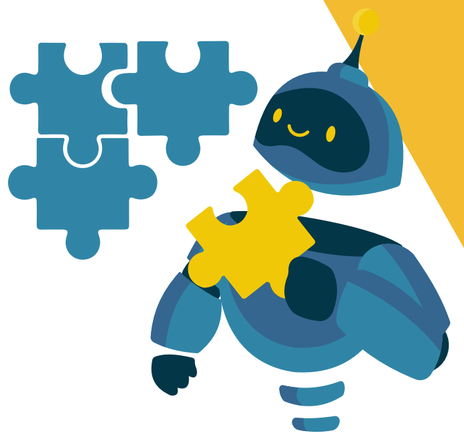
What is the seventh month of the year?,April,May,June,July,3

Which planet is closest to the Sun?,Saturn,Neptune,Mercury,Venus,2

...

What is the value of z in the equation 5z + 10 = 30?,4,5,6,7,0

What is the volume of a cube with side length 4cm?,8 cm^3,16 cm^3,32 cm^3,64 cm^3,3



File Handling

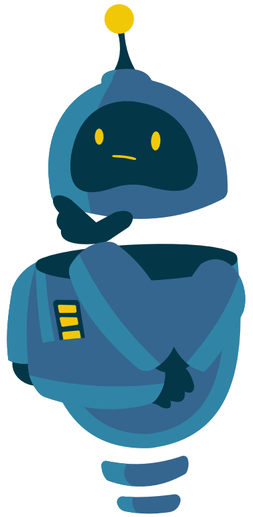
VORTEX ACADEMY
THE ART OF THINKING

When we want to store extra data that our game will use as, the game levels, special skins or items or save the state of the game in a way that is not reset when the game restarts as, checkpoints in the game, we use files.
We use file = **open**(r*"PATH_TO_FILE"*) to open the file and *file*.**close**() to close the file after using it.
NOTE that we must close any files we open during program execution in order to not cause issues for the operating system (windows).

In the previous example we used file = **open**(r*"questions.txt"*) to open the questions.txt file used to store the questions the game uses, their choices and the correct answer, then we used print(file.**read**()) to print the file contents then we closed it with file.**close**()

Syntax:

FILE_NAME = open(r"FILE_PATH")
Use the file
FILE_NAME.**close**()



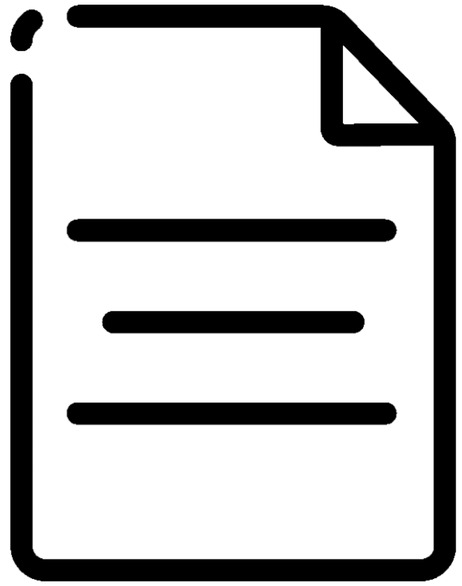
VORTEX ACADEMY
THE ART OF THINKING

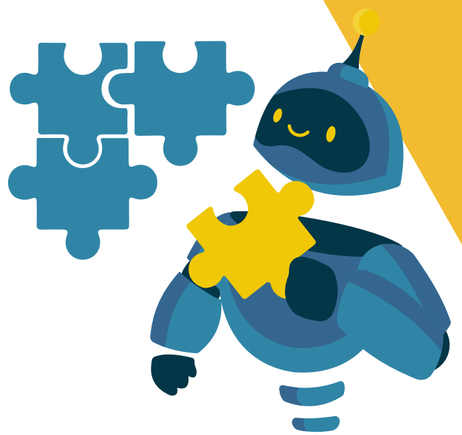
ACTIVITY#5.2

Try in another script the following code

```
# Change the path according  to your device.
with open(r"questions.txt") as file:
    print(file.read())
```

#HINT:use right click then copy path on the questions.txt file in vs code to get the path.





VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#5.2

OUTPUT:

The text in questions.txt file:

What is the capital of France?,London,Paris,Berlin,Tokyo,1

What is 5+7?,12,10,14,8,0

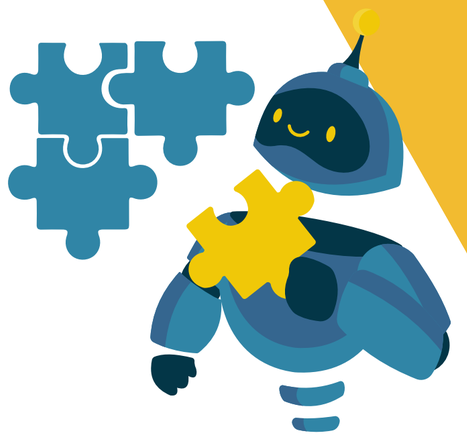
What is the seventh month of the year?,April,May,June,July,3

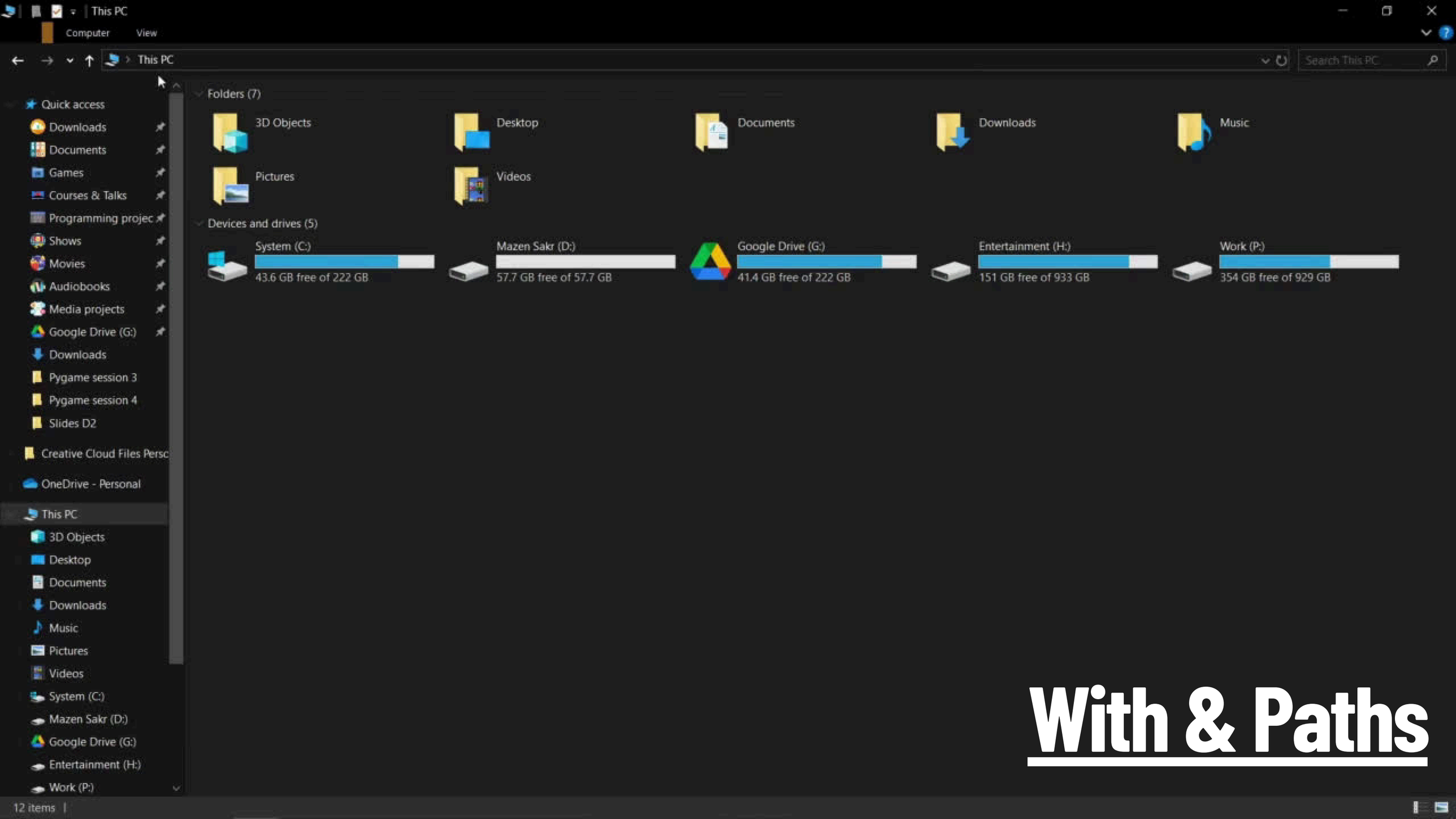
Which planet is closest to the Sun?,Saturn,Neptune,Mercury,Venus,2

...

What is the value of z in the equation 5z + 10 = 30?,4,5,6,7,0

What is the volume of a cube with side length 4cm?,8 cm^3,16 cm^3,32 cm^3,64 cm^3,3




This PC
Computer
View
This PC
Search This PC
Quick access
Downloads
Documents
Games
Courses & Talks
Programming projec
Shows
Movies
Audiobooks
Media projects
Google Drive (G:)
Downloads
Pygame session 3
Pygame session 4
Slides D2
Creative Cloud Files Persc
OneDrive - Personal
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
System (C:)
Mazen Sakr (D:)
Google Drive (G:)
Entertainment (H:)
Work (P:)
Folders (7)
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Devices and drives (5)
System (C:)
43.6 GB free of 222 GB
Mazen Sakr (D:)
57.7 GB free of 57.7 GB
Google Drive (G:)
41.4 GB free of 222 GB
Entertainment (H:)
151 GB free of 933 GB
Work (P:)
354 GB free of 929 GB
With & Paths
12 items

With & Paths

VORTEX ACADEMY
THE ART OF THINKING

In order to use files and be able to tell python which files we want to use we have to use **PATHS** which are where the files are stored on our computer.
They contain the drive name and the name of the folder/folders that contain our file.
ex: C:\Users\Public\Documents\document.txt
Where C:\ is the name of the drive that contains the file document.txt and it is contained in the folder \Documents inside the folder \Public inside the folder \Users in C:\

Instead of using the **open()** and **.close()** functions for file handling. We use
with open(r*"PATH_TO_FILE"*) **as** *FILE_NAME*:
    #What we want to do with the file
Which closes the file for us when python exits the **with** code block for easier use and safety.

Syntax:

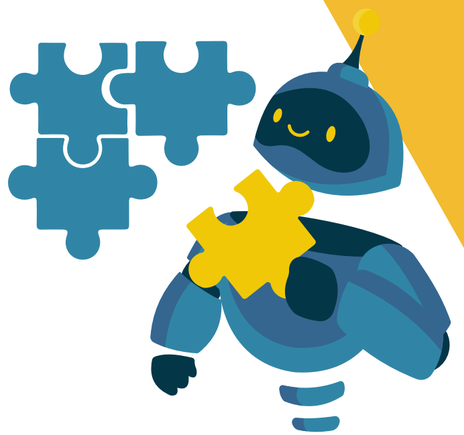
with open(r*"PATH_TO_FILE"*) **as** *FILE_NAME*:
 #What we want to do with the file

ACTIVITY#5.3

VORTEX ACADEMY
THE ART OF THINKING

Modify Your Previous Code & Try:

```
# Add to the game variables.
currentQuestion = ""
# Add this section.
with open(r"questions.txt") as file: #Replace the PATH correctly.
    questions = file.readlines() # This transforms the file lines into list items.
    for index, line in enumerate(questions):
        questions[index] = line.strip().split(",") # This transforms each line from the
list into a list that contains the question, it's answers and the correct answer.
    currentQuestion = random.choice(questions)
    questions.remove(currentQuestion)

# Replace the question and answer box drawing in the draw() function.
        screen.draw.filled_rect(main_box, "steel blue")
        screen.draw.textbox(currentQuestion[0], main_box, color = "white")

        for index, box in enumerate(answer_boxes):
            screen.draw.filled_rect(box, "midnight blue")
            screen.draw.textbox(currentQuestion[index+1], box, color = "white")
```

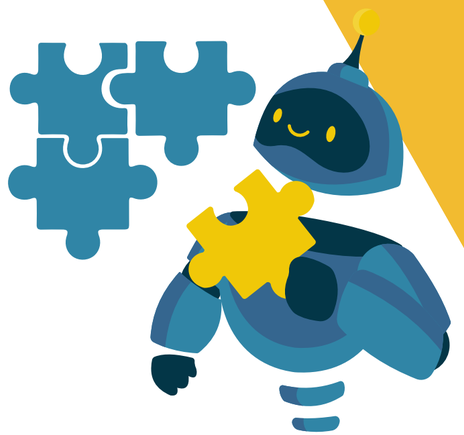


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#5.3

OUTPUT:

Now the game loads questions from the questions.txt file.

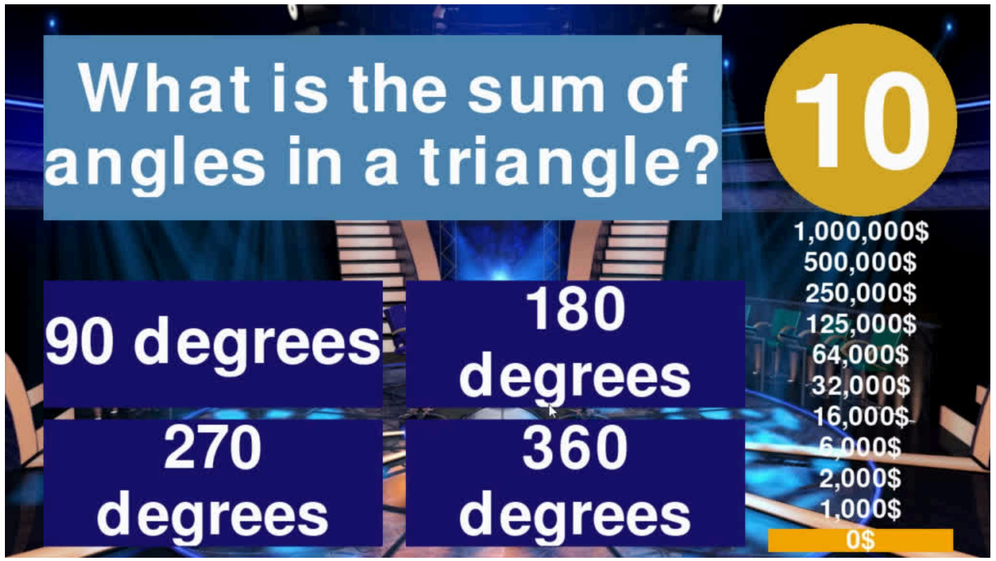


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#6.1

To Continue Our Game:

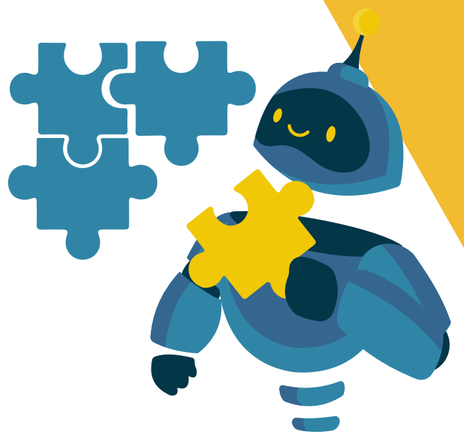
How can we make the mouse clicks trigger a response from the game? How can we make it indicate when we click the correct answer or the wrong answer?



Check in **on_mouse_down()** which of the boxes was clicked and wether it's index in the **answer_boxes** list matches the index of the correct answer.

#Hint: use enumerate().

The questions are read from the questions.txt file to become a list in the format [Question, ans1, ans2, ans3, ans4, correctAns] where correctAns indicates the index of the correct answer where ans1 has an index of 0 and so on.

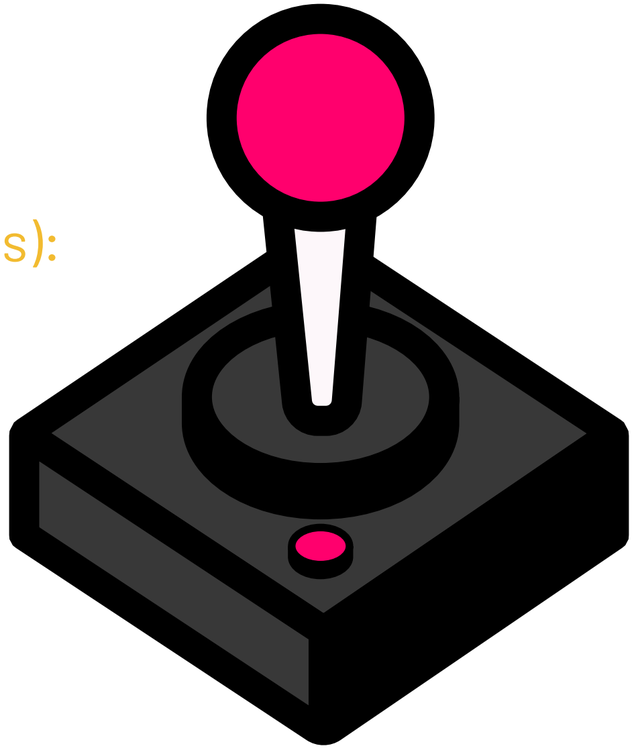


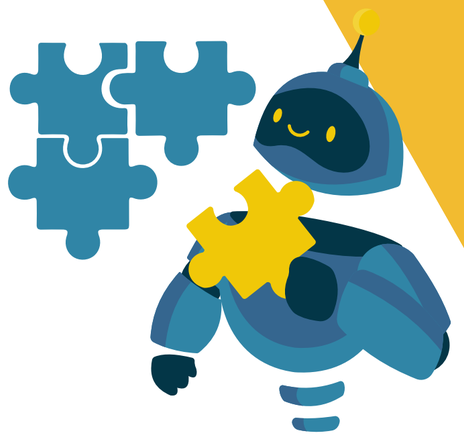
VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#6.1

Modify Your Previous Code & Try:

```
# Modify the on_mouse_down() function by adding the lines in yellow.
def on_mouse_down(pos):
    global gameStarted, currentQuestion, timeLeft, answer_boxes
    if not gameStarted:
        gameStarted = True
    else:
        for index, box in enumerate(answer_boxes):
            if box.collidepoint(pos):
                if index == int(currentQuestion[5]):
                    print("correct_answer")
                else:
                    print("game_over")
```

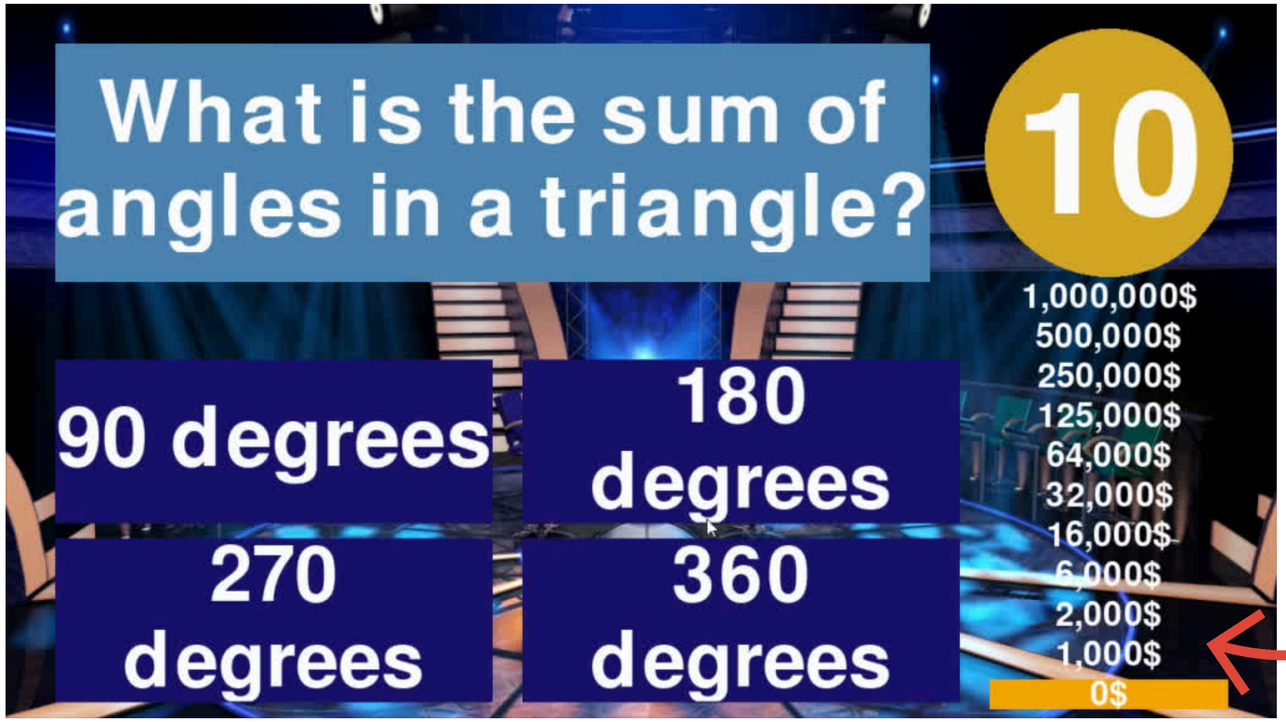




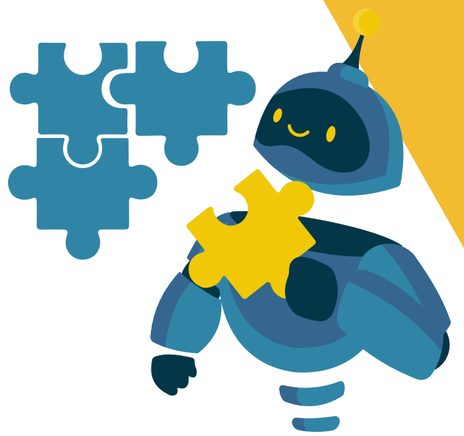
VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#6.1

OUTPUT:



Now correct_answer is printed when we click on the correct answer, and game_over is printed when we click on the wrong answer or the time runs out.

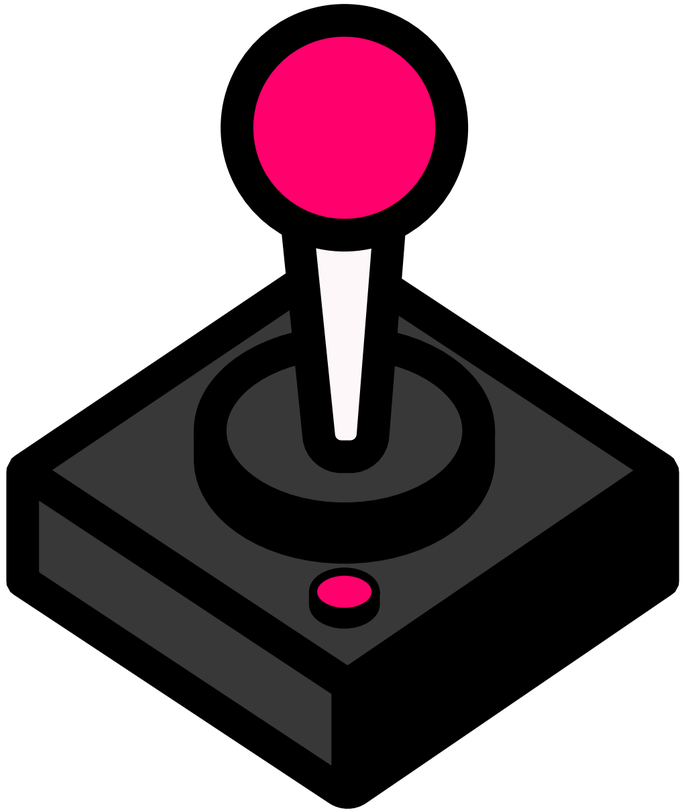


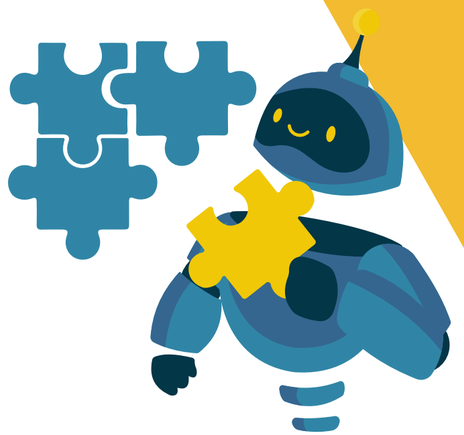
VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#6.2

Try in another script the following code

```
import pgzrun

WIDTH = 1280
HEIGHT = 720
TITLE = "Who Wants to be a Millionaire?"

box1 = Rect((0, 0), (100, 200))

def draw():
    screen.fill("slate gray")
    screen.draw.filled_rect(box1, "steel blue")

def on_mouse_down(pos):
    box1.move_ip(0,40)

def update():
    pass

pgzrun.go()
```



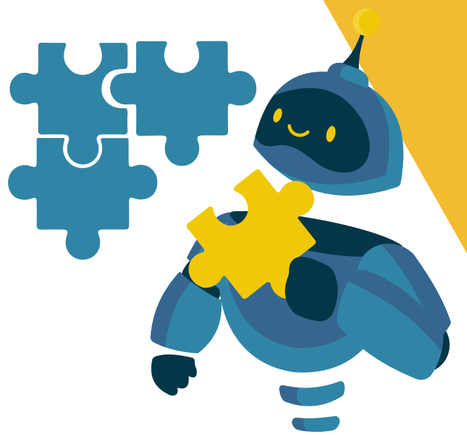


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#6.2

OUTPUT:

The rectangle moves by (0,40) to the left when we click anywhere on the screen.



#HINT: try changing the value by which the rectangle moves by changing the coordinates given to move_ip(X,Y)

VORTEX ACADEMY
THE ART OF THINKING

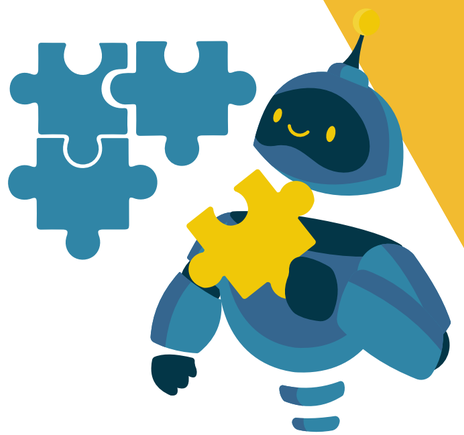
ACTIVITY#6.3

To Continue Our Game:

How can we make our mouse clicks move us to the next level if we click on the correct answer?

Implement the score game variable and **correct_answer**() function to move the player to the next level when they click on the correct answer.

#Hint: It should move the prize_box up by (40,0). Increase the score by 1, choose a new random question, remove it from the questions list and assign it to the currentQuestion variable if the player's score is less than the maximum score of the game, otherwise it should end the game.

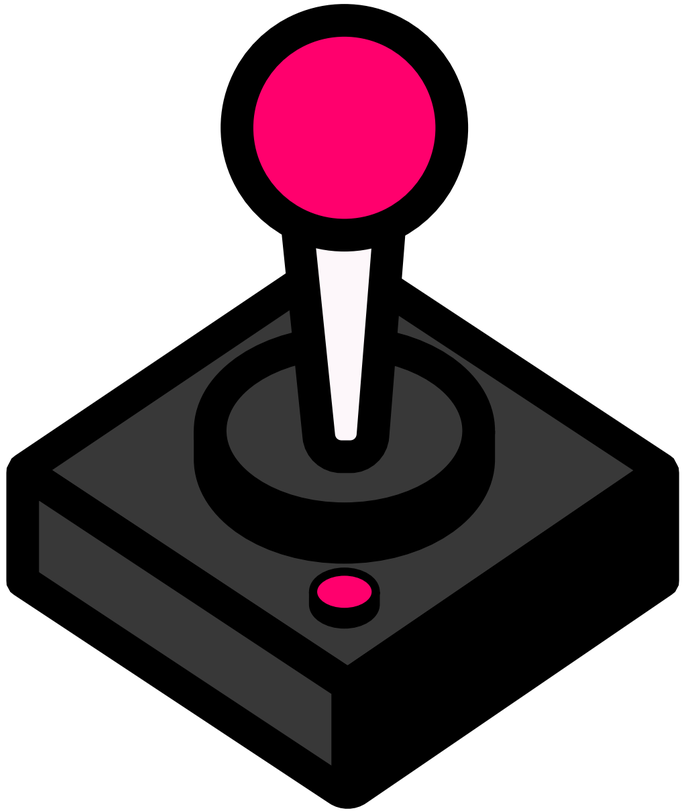


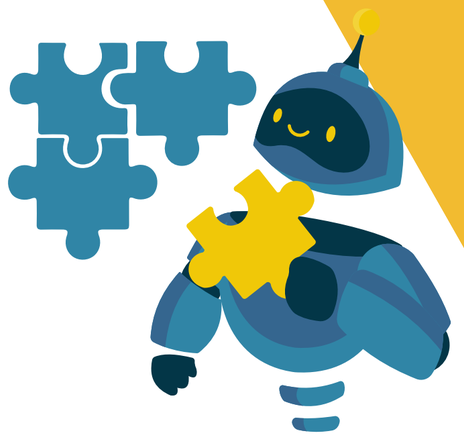
ACTIVITY#6.3

VORTEX ACADEMY
THE ART OF THINKING

Modify Your Previous Code & Try:

```
# Add to game variables.
score = 0
# In on_mouse_down(pos): replace print("correct_answer") with the lines in yellow.
    else:
        for index, box in enumerate(answer_boxes):
            if box.collidepoint(pos):
                if index == int(currentQuestion[5]):
                    correct_answer()
                else:
                    print("game_over")
# Add this function above the update function.
def correct_answer():
    global currentQuestion, timeLeft, score, questions
    score += 1
    prize_box.move_ip(0,-40)
    if score < 10:
        currentQuestion = random.choice(questions)
        questions.remove(currentQuestion)
        timeLeft = 10
    else:
        print("game_over")
```



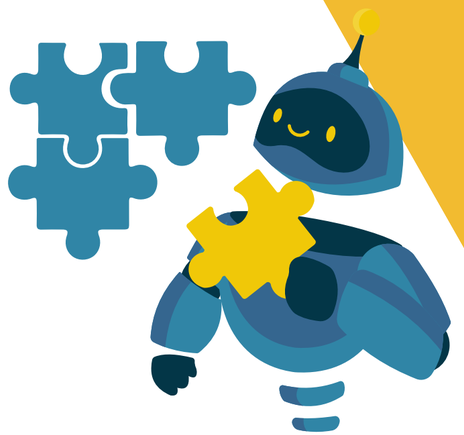


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#6.3

OUTPUT:

Now our level increases when we click on the correct answer and game over is printed when we reach the maximum score.

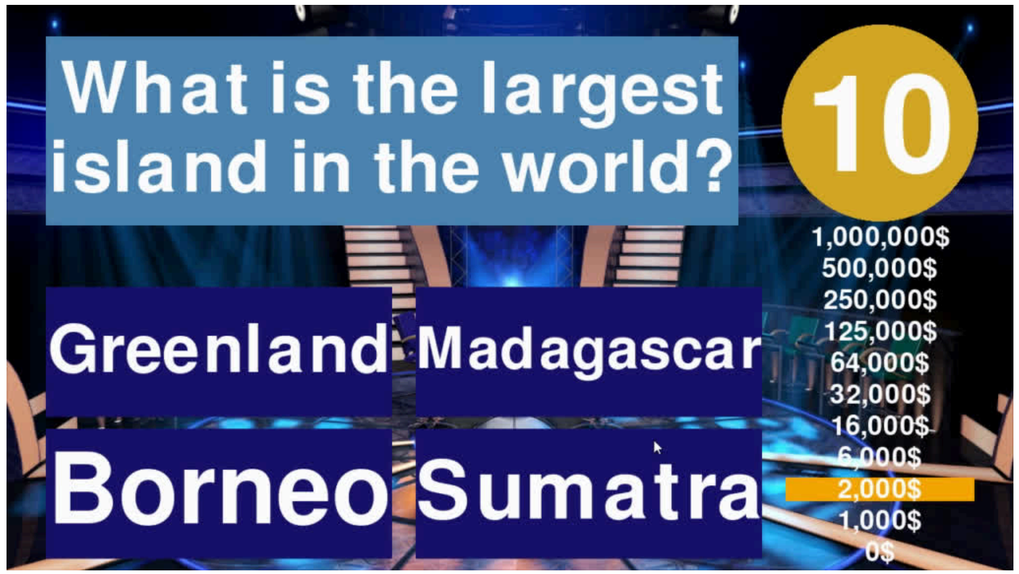


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#6.4

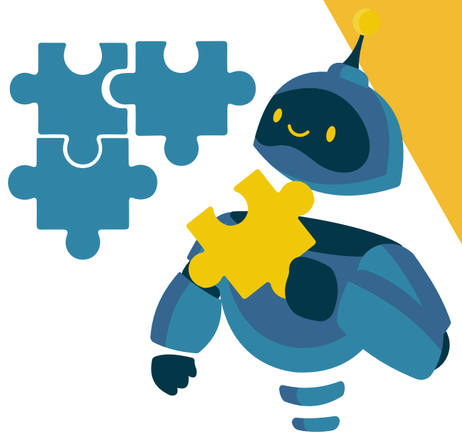
To Continue Our Game:

How can we make mouse clicks on the wrong answer end the game and show how much money the player has won? How can we end the game when the player has won a million?



Implement the **game_over()** function that sets the time left to 0, the main box to how much money the player has won. and the answer boxes to "-" and unschedules the updateTime() function.

#Hint: write a custom question to contain how much money the player has won and the "-" for the answers, assign it to currentQuestion and draw() will display it.



VORTEX ACADEMY
THE ART OF THINKING

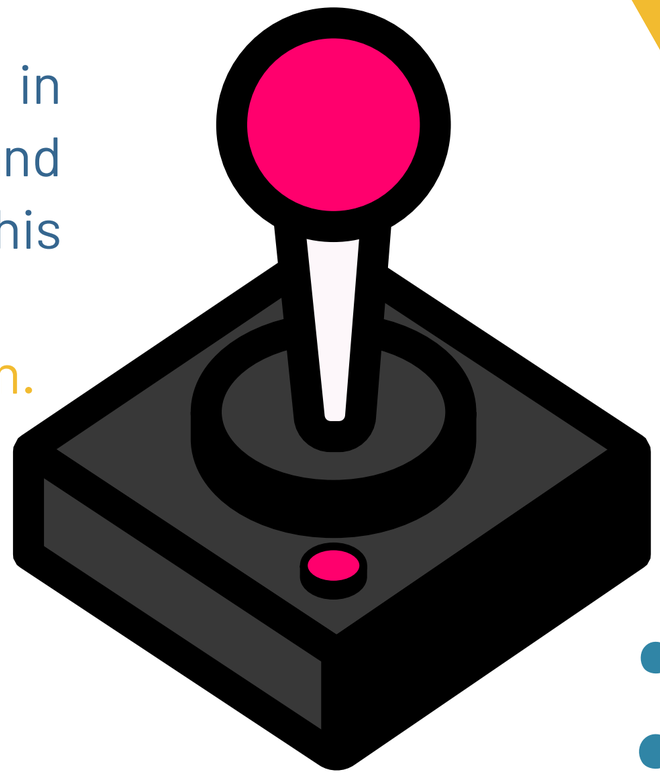
ACTIVITY#6.4

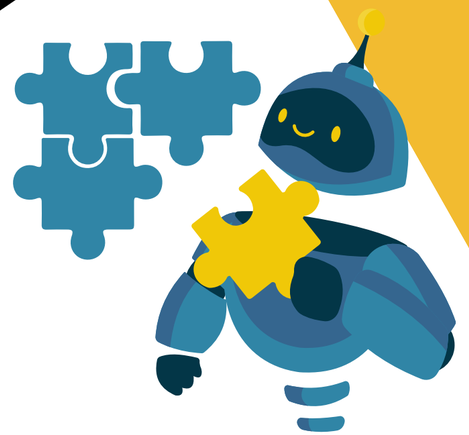
Modify Your Previous Code & Try:

Replace the print(“game_over”) in correct_answer() , update_time() and on_mouse_down() with game_over() and add this function.

Place this function above the update function.

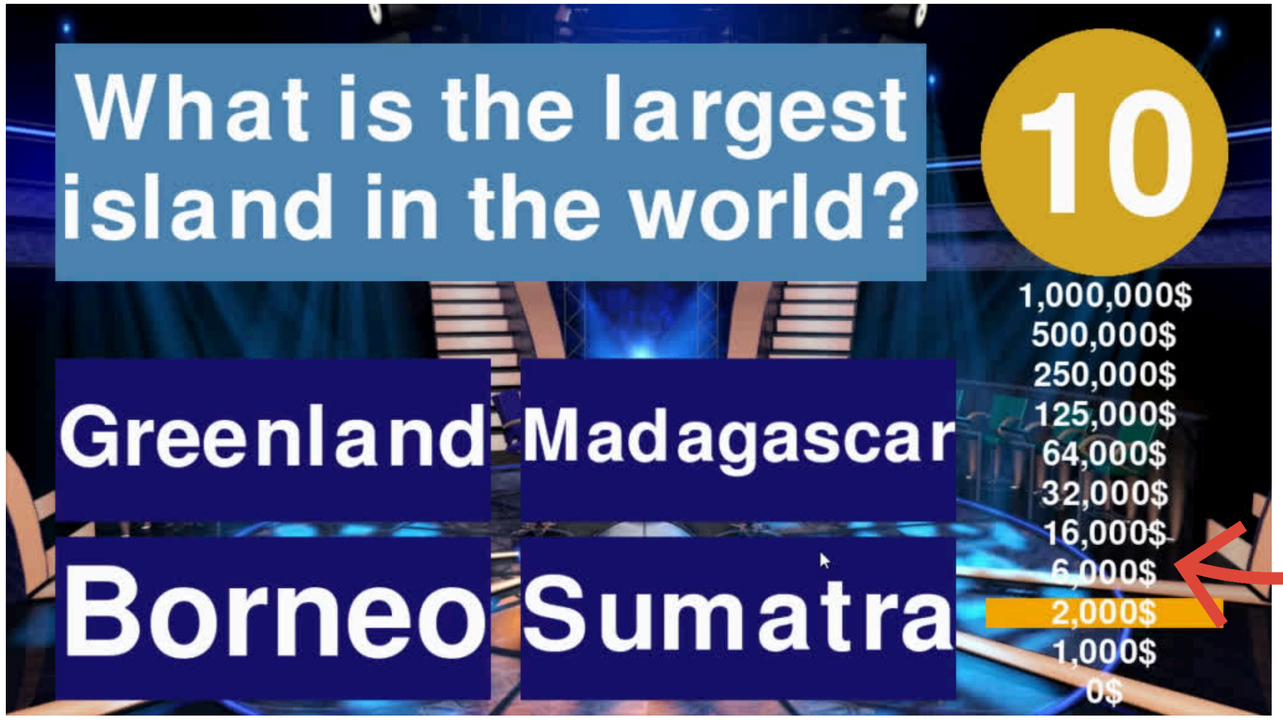
```
def game_over():
    global currentQuestion, timeLeft
    message = "You won" + prizes[score]
    currentQuestion = [message, "-", "-", "-", "-", 5]
    timeLeft = 0
    clock.unschedule(updateTime)
```



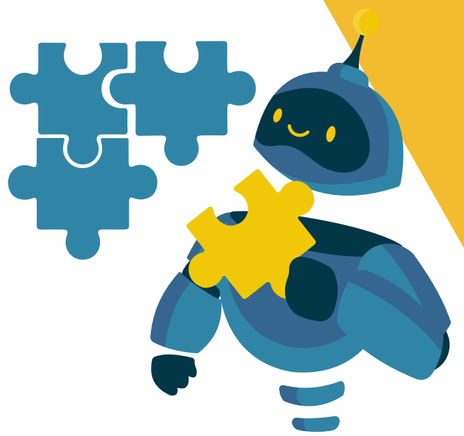


ACTIVITY#6.4

OUTPUT:



Now the game is finished.

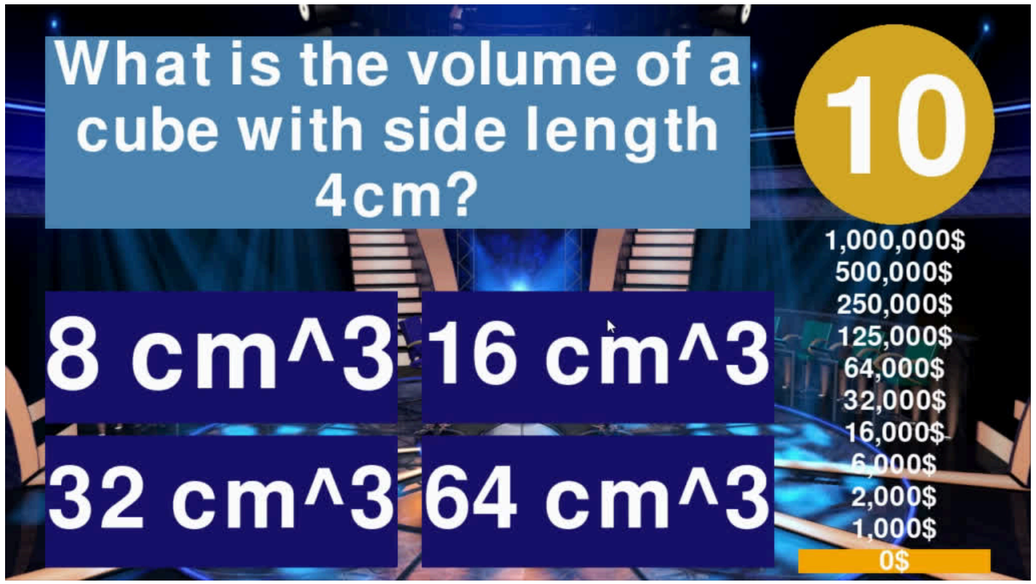


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#7.1

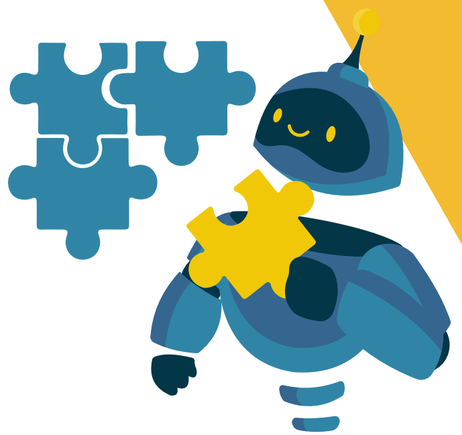
To Continue Our Game:

How can we add background music to the game?



Add background music to the game.

#Hint: you can look at your previous code and copy form it



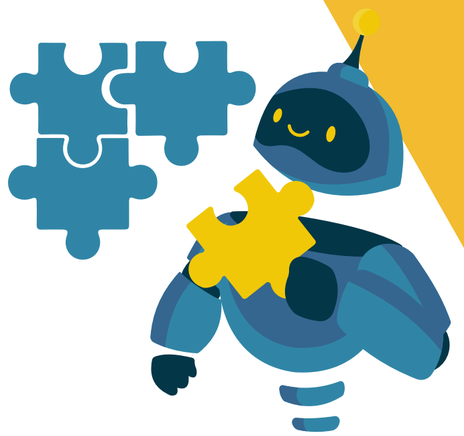
VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#7.1

Modify Your Previous Code & Try:

Add the following line any where in your code

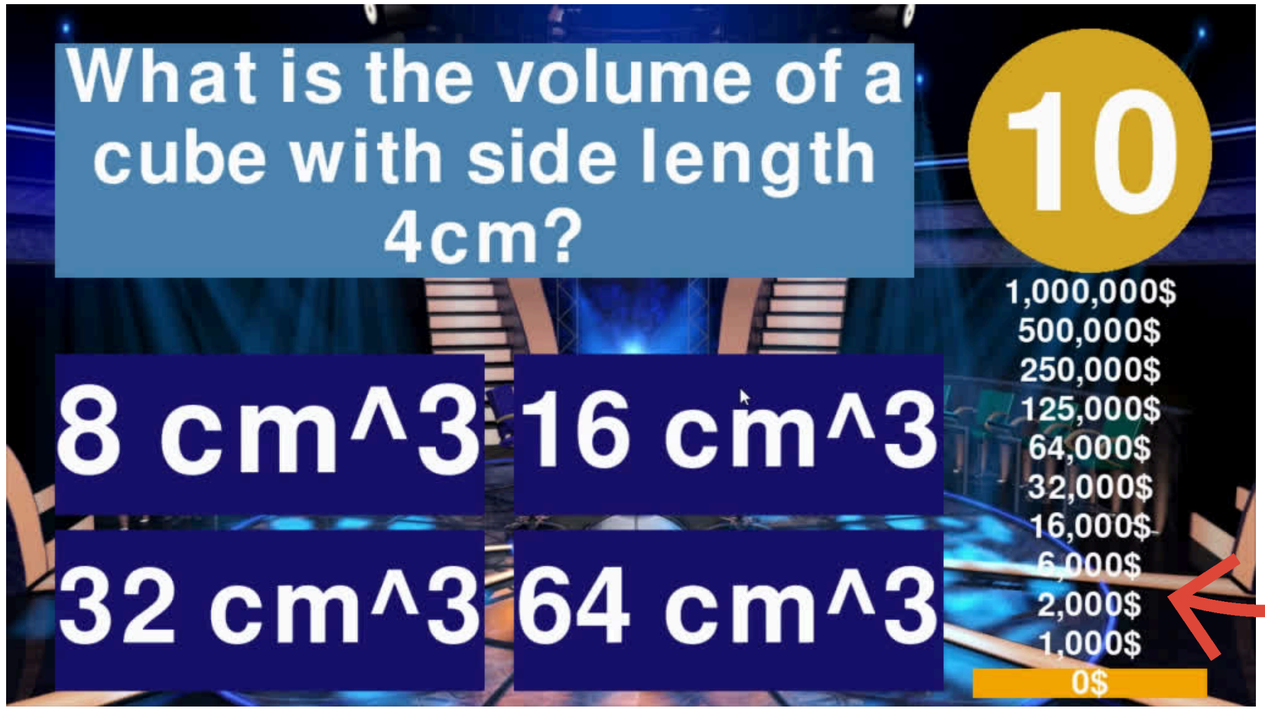
music.**play**("background")



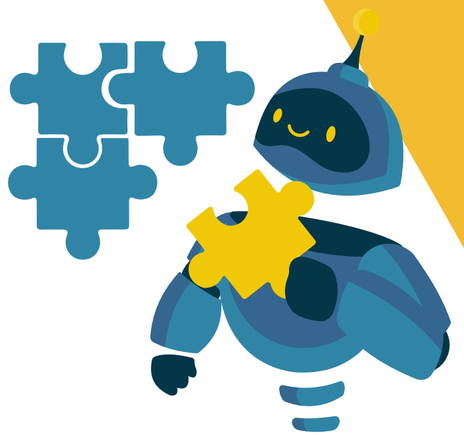
VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#7.1

OUTPUT:



Now background music plays when we start the game.

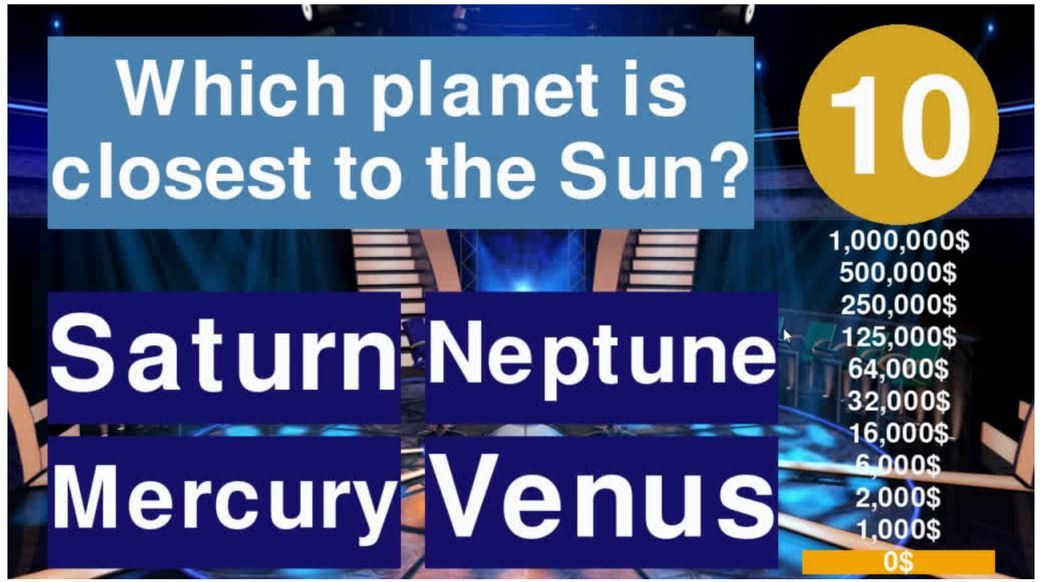


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#7.2

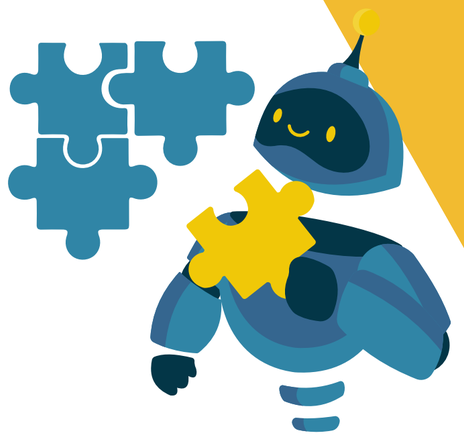
To Continue Our Game:

How can we add sound effects to the game when the player gets an answer correct or loses the game?



Add sound effects to the **correct_answer**() and the **game_over**() functions.

#Hint: the sound effects names are "*correct*" and "gameover"

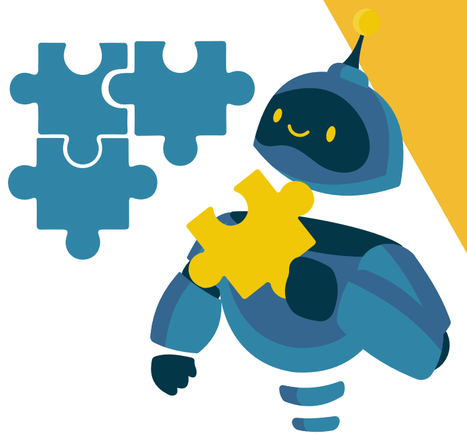


VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#7.2

Modify Your Previous Code & Try:

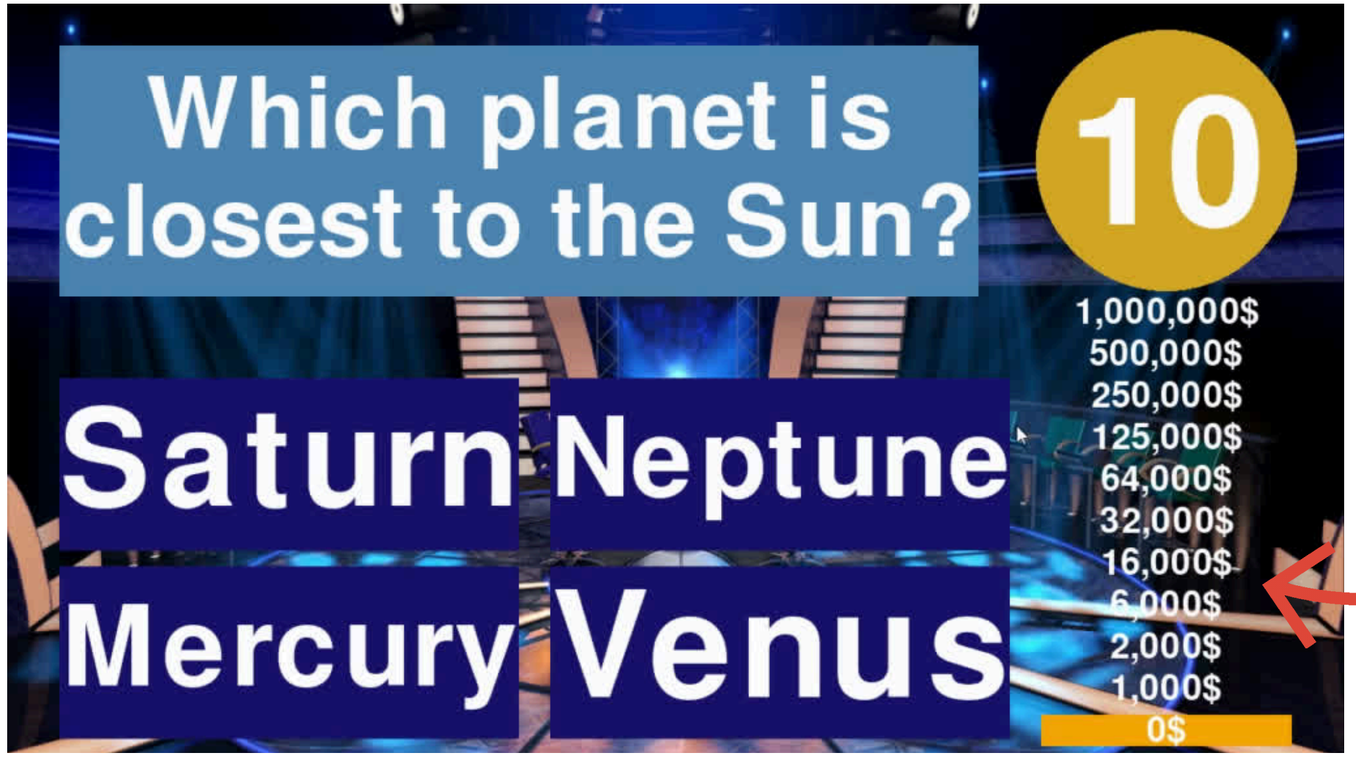
```
def correct_answer(): #Add the lines in yellow.
    global currentQuestion, timeLeft, score, questions
    score += 1
    sounds.correct.play()
    prize_box.move_ip(0,-40)
    if score < 10:
        currentQuestion = random.choice(questions)
        questions.remove(currentQuestion)
        timeLeft = 10
    else:
        game_over()
def game_over():
    global currentQuestion, timeLeft
    message = "You won" + prizes[score]
    sounds.gameover.play()
    currentQuestion = [message, "-", "-", "-", "-", 5]
    timeLeft = 0
    clock.unschedule(updateTime)
```



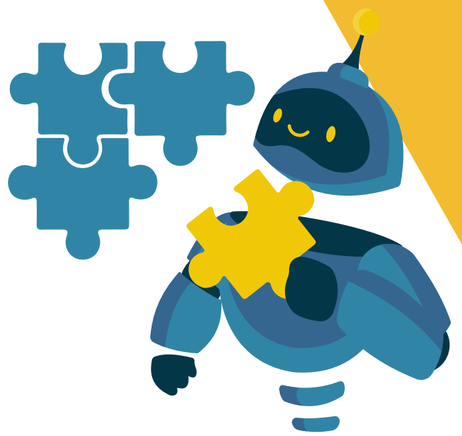
VORTEX ACADEMY
THE ART OF THINKING

ACTIVITY#7.2

OUTPUT:



Now the game has background music and sound effects.



ASSIGNMENT

1	NOTES	Write session notes
2	SOLVE ASSIGNMENT	

ASSIGNMENT

Continuing on the code that you wrote during the session.

Task1: Modify your code to make the answer buttons smaller and move them to the side then move the prizes and prize box to the side to make room for hint buttons, then implement the following buttons:

a. A button that adds 10 seconds to the timer that can be used only once per game.

b. A button that removes 2 of the answers that can be used only once per game.

#HINT choose a random index that isn't the correct answer's index and remove all other answers

c. A button that skips the current question that can be used only once per game.

#HINT change the current question and reset the timer.

Task2: Add more questions to the game.

#HINT add more questions to the questions file in the same format it contains, which is:

question,answer1,answer2,answer3,answer4,index_of_correct_answer_starting_from_0.

VORTEX ACADEMY
THE ART OF THINKING

THANK YOU

FOR YOUR ATTENTION