

Intro To Python

Hoosier Hacks

intro to python

00000001

strings

Strings are the technical term for text.

They must be surrounded in single or double quotation marks in Python.

You can show them on screen by using `print()`.

```
>>> print('hello world')  
hello world
```

intro to python

00000010

variables

Variables can store data in them, such as numbers or strings.

You can change their values using `=` and do math with them.

```
>>> my_number = 5
>>> my_number = my_number + 1
>>> print(my_number)
6
>>> my_number = 420
>>> print(my_number)
420
```

intro to python

00000011

comparisons

You can check if two numbers are equal by using the equality operator `==`.

```
>>> a = 4
>>> b = 4
>>> print(a == b)
True
```

```
>>> password = '196572b'
>>> guess = '106572b'
>>> print(password == guess)
False
```

intro to python

00000100

more comparisons

You can use `!=` to check if two things are *not* equal.

You can also use the math comparisons `>=`, `>`, `<=`, and `<`.

```
>>> print(5 >= 3)
True
>>> print(4 != 17)
True
```

intro to python

00000101

control flow: if this, else that

Control flow lets us determine what we want to do depending on some condition.

Let's say we're at a restaurant that serves alcoholic drinks and we need to make sure everyone is drinking legally.

```
>>> age = 15
>>> if age >= 21:
...     print('What would you like to order?')
... else:
...     print('Sorry, you are not allowed to drink alcohol yet!')
...
Sorry, you are not allowed to drink alcohol yet!
```

intro to python

00000110

data structures: lists

Lists can store multiple items in one variable.

You can define them using the brackets [and].

You access them with `list[number]`. Be careful, `number` starts at 0!

```
>>> mylist = ['first', 'second', 'third', 'fourth']
>>> print(mylist[0])
first
>>> print(mylist[2])
third
```

intro to python

00000111

data structures: lists

You can also change items inside a list.

```
>>> mylist = [1, 3, 5]
>>> mylist[0] = 2
>>> mylist[1] = 5000
>>> print(mylist)
[2, 5000, 5]
```


intro to python

00001000

data structures: lists

You can add more numbers too.

```
>>> mylist = [5, 4, 3]
>>> mylist.append(2)
>>> mylist.append('ice cream')
>>> print(mylist)
[5, 4, 3, 2, 'ice cream']
```

intro to python

00001001

data structures: tuples

Tuples are almost the same as lists.

But once you make them, *they can never be changed*.

We use (and) to define them.

```
>>> mytuple = (1, 2, 3)
```

```
>>> mytuple[0] = 3
```

```
Traceback (most recent call last):
```

```
  File "<stdin>", line 1, in <module>
```

```
TypeError: 'tuple' object does not support item assignment
```

Error!

intro to python

00001010

iteration

You can use the `for` loop to perform an action for every item in a list or tuple.

```
>>> numbers = (2, 3, 4, 7)
>>> for x in numbers:
...     print(x)
...
2
3
4
7
```

In a `for` loop, we have a loop variable that changes each time we run it.

Here, the loop variable is `x`.

intro to python

00001011

functions

Functions are pieces of code that we write once and can use over and over again.

```
>>> def my_function():  
...     print('a useful function')
```

```
>>> my_function()  
a useful function  
>>> my_function()  
a useful function
```

intro to python

00001100

advanced functions

Functions can sometimes take in parameters.

Trivia: What's the difference between an argument and a parameter?

This lets you pass some information *to* a function.

```
>>> def be_annoying(word):  
...     print(word)
```

```
>>> be_annoying('apple')  
apple
```

```
>>> be_annoying('orange')  
orange
```

intro to python

00001101

advanced functions

Functions can also return a value.

It's like the opposite of parameters: we can get information *from* a function.

```
>>> def give_me_five():  
...     return 5
```

```
>>> number = give_me_five()  
>>> print(number)  
5
```

intro to python

00001110

methods

Advanced functions in python are called *methods*.

You'll have to use a dot to access them.

We'll see a few of these. Don't worry too much about them for now.

```
game_won = self.core.check_victory()
if game_won:
    self.playing = Playing.ENDING
    clear_canvas(self.canvas)
    self.core.handle_victory()
    self.core.game_won = True
```

Questions?

intro to python

00010000

coding challenge

You get a list of numbers. You need to give back a list of numbers. The new list should have each old number, but it should be doubled.

Input: `[1, 6, 5, 1, 7, 4, 12]`

Required Output: `[2, 12, 10, 2, 14, 8, 24]`

No cheating!

intro to python

00010001

coding challenge solution

```
>>> first_list = [1, 6, 5, 1, 7, 4, 12]
>>> answer = []
>>> for number in first_list:
...     answer.append(number * 2)
```

```
>>> answer
[2, 12, 10, 2, 14, 8, 24]
```

intro to python

00010010

coding challenge solution

Or, if you're a pro:

```
>>> answer = [x*2 for x in first_list]
```

Pygame

WL Hack Club

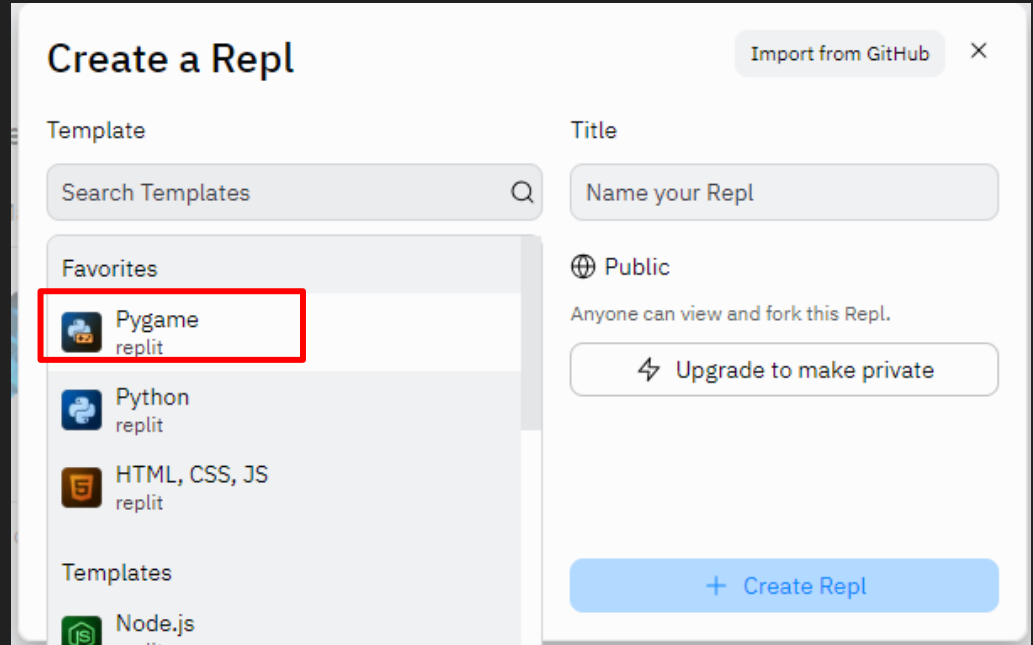
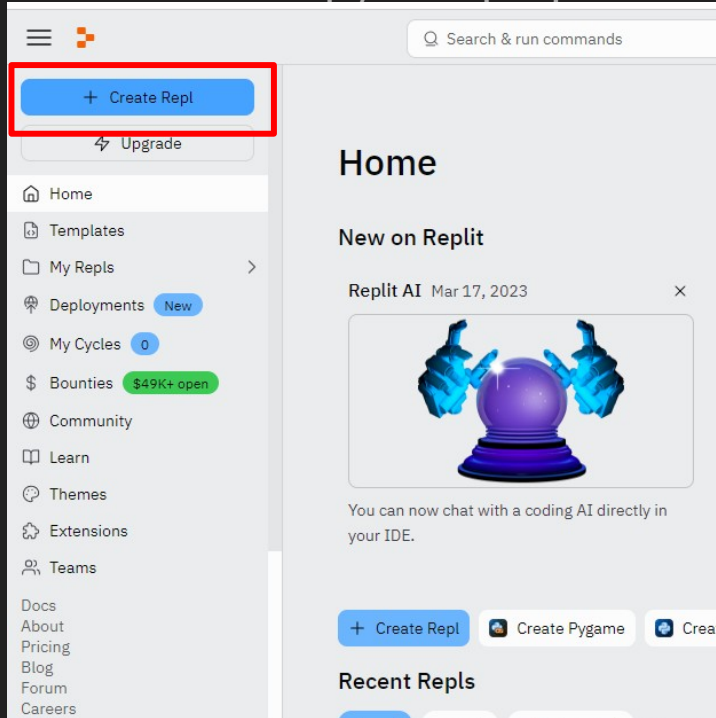
[https://replit.com/@TomasVargas-Ber/Fall-Hackathon-Pygame-Workshop-2023#main.p
y](https://replit.com/@TomasVargas-Ber/Fall-Hackathon-Pygame-Workshop-2023#main.py)

getting started

00010101

create your project in repl.it

Create a new python project in repl.it.

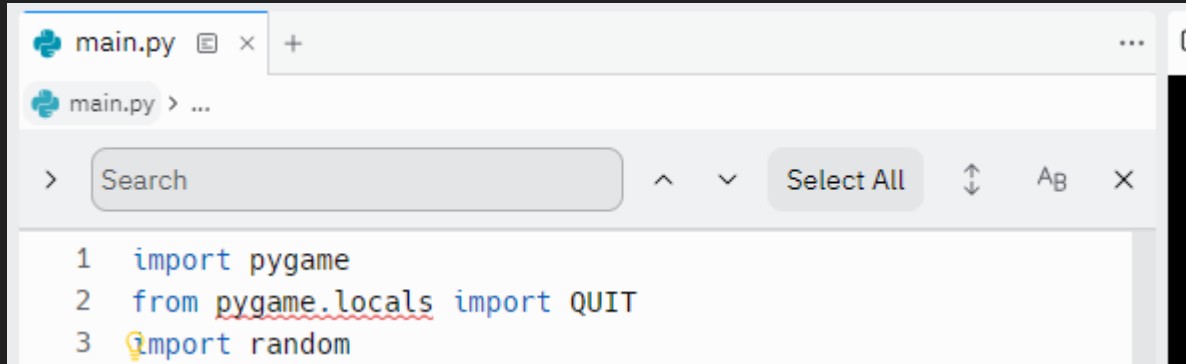


getting started

installing pygame

Type `import pygame` along with a few others into repl.it and run it.

Pygame should install automatically for you.



```
main.py x +
main.py > ...
> Search ^ v Select All ⬆ ⬆ AⒷ X
1 import pygame
2 from pygame.locals import QUIT
3 import random
```

getting started

Defining variables

This first piece of code will give us our first pygame display!

But don't run it yet!

```
6 width, height = 500, 400
7 screen = pygame.display.set_mode((width, height))
8 clock = pygame.time.Clock()
9
10 dvd_height = 100
11 dvd_width = 100
12
13 dvd = pygame.image.load('dvd.png').convert_alpha()
14 dvd = pygame.transform.scale(dvd, (dvd_width, dvd_height))
15
16 dvd_x = width//2
17 dvd_y = height//2
18
19 x_speed = 1
20 y_speed = 1
21
```

getting started

00011010

Helper functions

Updates the color of the dvd image each time it hits a wall

```
23 v def change_color():
24     image = pygame.image.load('dvd.png').convert_alpha()
25     image = pygame.transform.scale(image, (dvd_width, dvd_height))
26     colored = pygame.Surface(image.get_size())
27     colored.fill(random.randint(0, 0xffffffff))
28
29     final = image.copy()
30     final.blit(colored, (0, 0), special_flags = pygame.BLEND_MULT)
31     return final
```


getting started

Helper functions

This will update the position of the dvd image and make it bounce off the borders

```
34 v def move():
35     global dvd_x, dvd_y, x_speed, y_speed, dvd
36     dvd_x += x_speed
37     dvd_y += y_speed
38
39 v if dvd_y + dvd_height - 30 > height or dvd_y + 30 < 0:
40     y_speed *= -1
41     dvd = change_color()
42 v if dvd_x + dvd_width - 5 > width or dvd_x + 5 < 0:
43     x_speed *= -1
44     dvd = change_color()
```

getting started

00011010

Main loop

```
48 v def main():
49 v     while True:
50 v         for event in pygame.event.get():
51 v             if event.type == pygame.QUIT:
52                 pygame.quit()
53
54                 screen.fill((125, 125, 125))
55                 screen.blit(dvd, (dvd_x, dvd_y))
56                 pygame.display.update()
57                 move()
58
59                 clock.tick(60)
60
61
62 v if __name__ == '__main__':
63     main()
```

Questions?

Experiment with the Code!