WELCOME TO JOURNEYS WITH PYTHON

An Introductory Workshop for Python

By Anil Duvvuri, Adit Kantak, and Saketh Penumudy

Let’s Get To Know You!

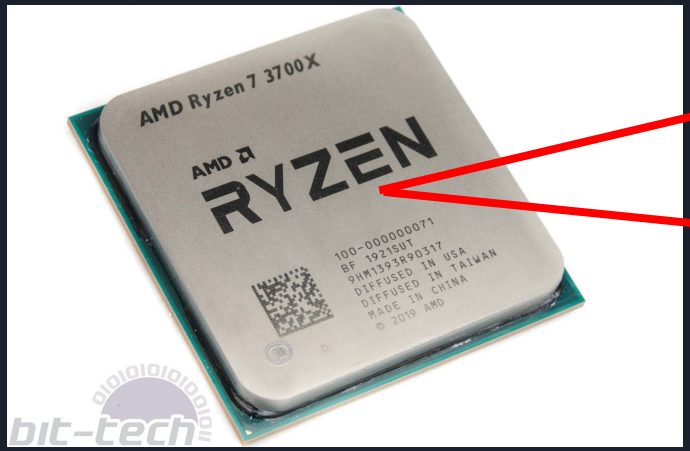
Say your name, what school you go to, what your favorite animal is, and why you like it.

Raise your hand if you want to speak, but if nobody raises their hand, then we will just choose a person for you.

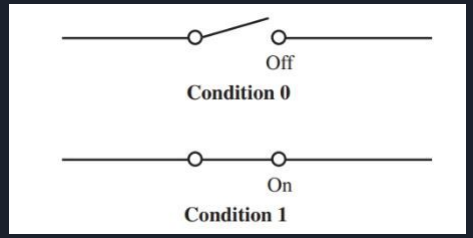
Day 1: **What Even *Is* Programming?**

In order to ***understand*** *what programming* ***is****, we have to* ***ask the question****, “****What*** *is a computer, and* ***what*** *does it do?”*

Inside a computer

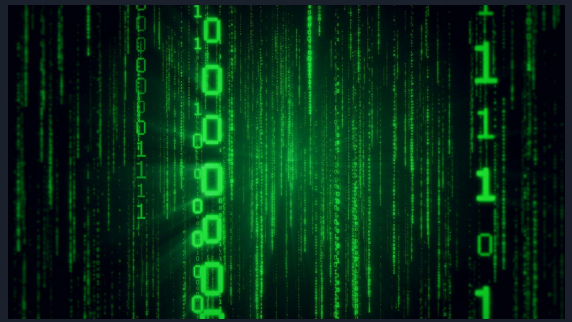


These switches that are in the computer are called **transistors.** They, like light switches in your house, turn on and off, which tells the computer the number 0 when it is off, and 1 when it is on.

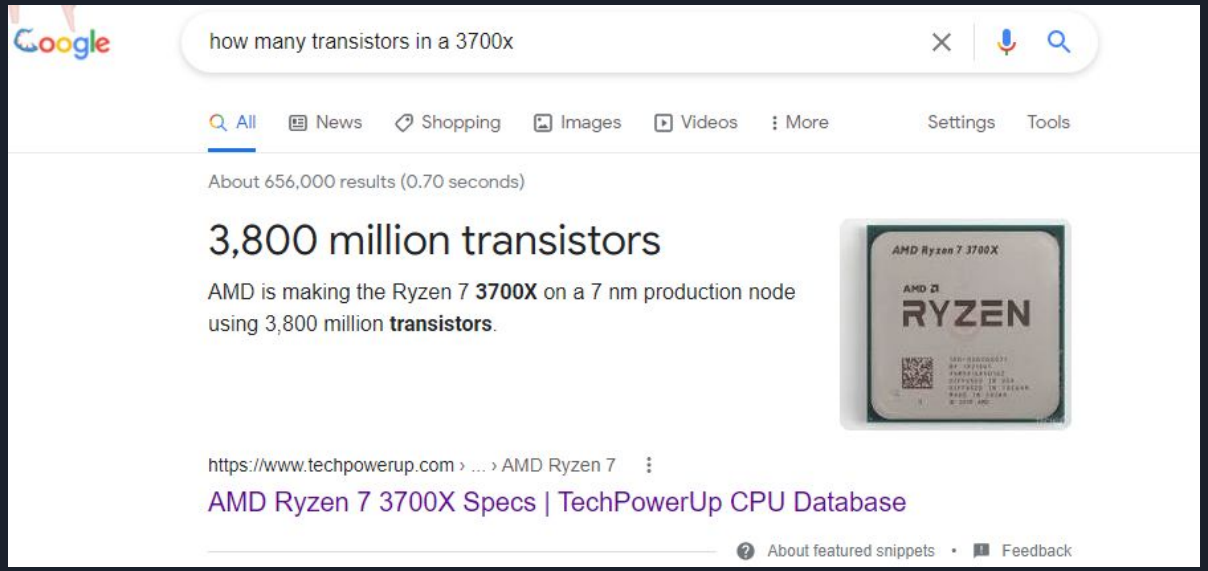


A really simple way to explain what a computer is - is a lot of switches (like the ones you use to turn the lights in your room on and off). These switches turn on and off in certain patterns, which tells the computer what to do.

=



Inside a computer (contd.)



= 3,800,000,000 transistors

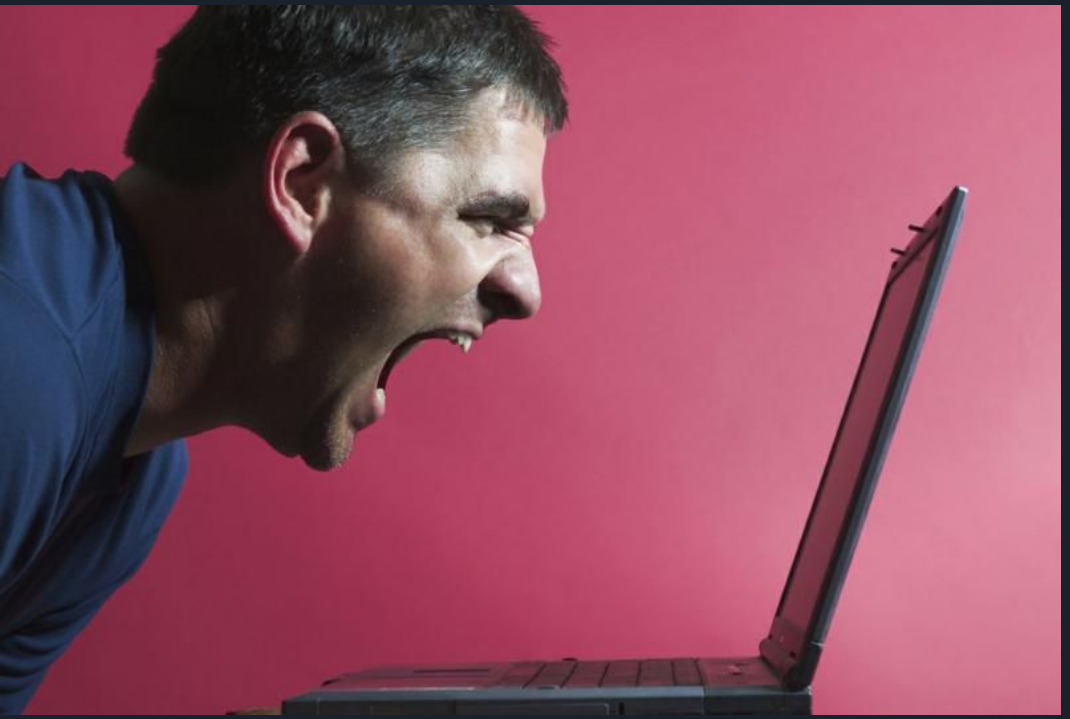
Yes, that's 3.8 **BILLION** transistors

That's more transistors than people in both India and China COMBINED

Now that we **know** how a computer thinks, let's move forward on to how we can **make** it think **for us** and do the things you are used to seeing a computer do!

Now that we know what a computer is, ***what even is code****?*

What is code?



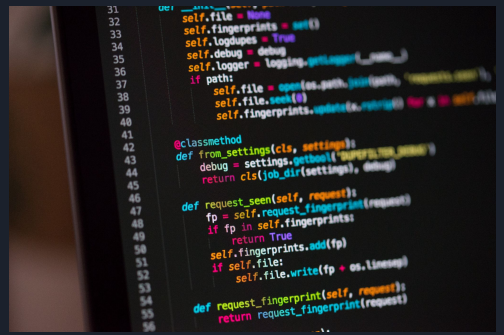
A **language** is a way that you can **communicate** with someone or something else in a way both can **understand**.

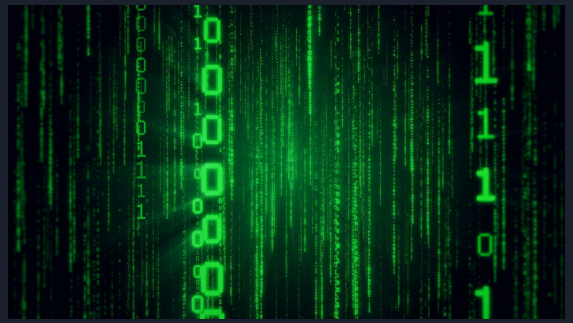
Everybody knows that you can't have a conversation with your computer in a normal language like English, Spanish, or German.

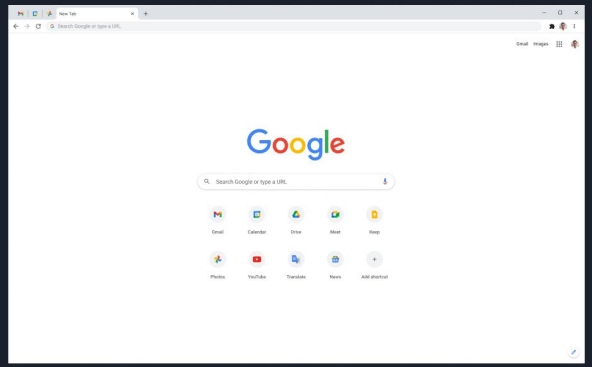
What is code (continued.)?

If you only spoke English to someone who mainly spoke Spanish but was learning English, they would have to translate what you said into Spanish to understand it.

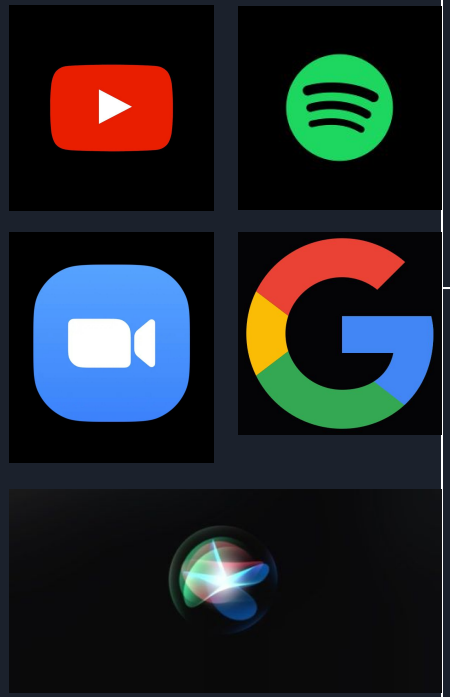
Just like people translate one language to another, computers translate the Python language into Binary to understand it and do what you asked it to do (Binary means the 1s and 0s we talked about earlier)







What can programming do?

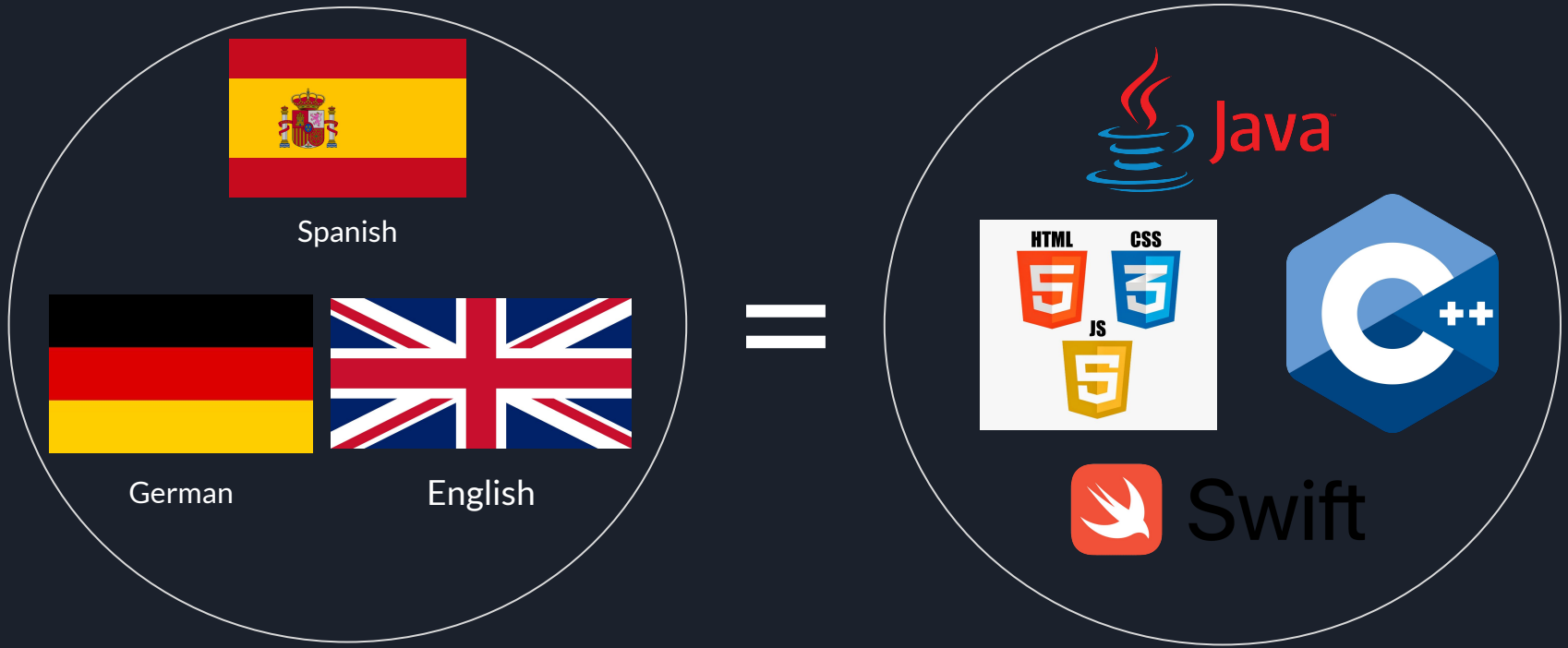


Programming is actually a very, very important part of running day-to-day tasks on your computer that we all use. For example, programming was used to make

- Super Mario
- Minecraft
- YouTube
- Google
- Zoom
- Spotify
- Siri
- And much much more!

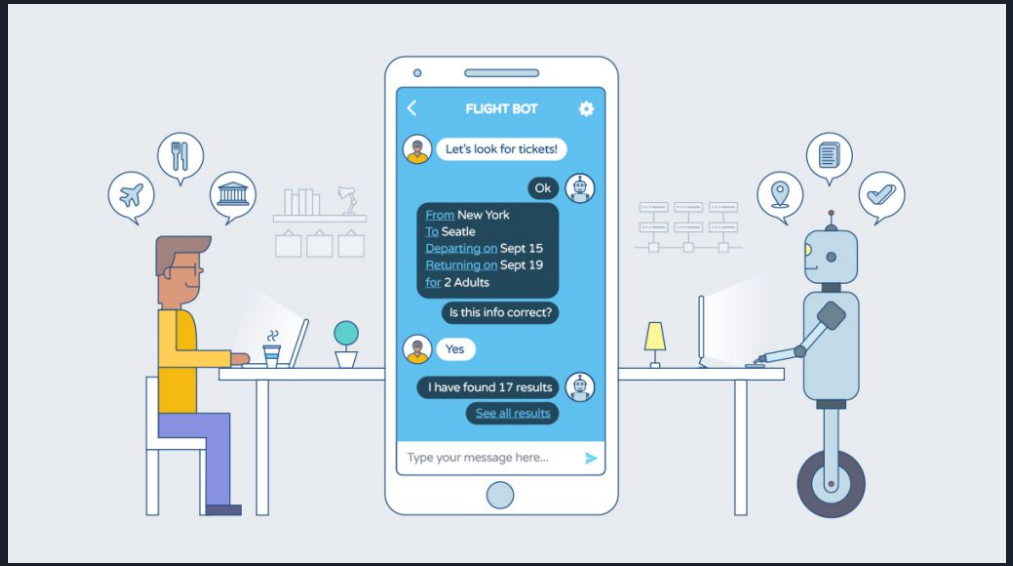
Which other programming languages are there?

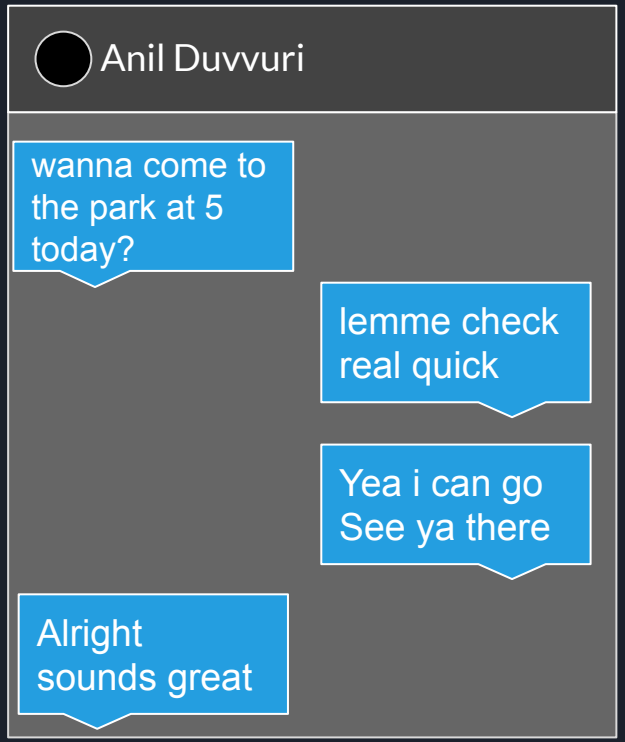
Although we won't learn them in this course, there are many other programming languages that exist, just like there are other languages that people speak. Some famous ones other than Python include Java, the language used to make Minecraft: Java Edition, HTML/CSS/JavaScript, the languages used to make websites on the internet, Scratch, a children's block coding platform, and many others like C#, C++, Swift, and Go.

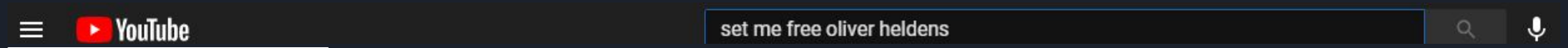


The main parts of a Computer Program are ***Input***, ***Processing***, *and* ***Output***

One way to **visualize** a computer program is a text conversation with a friend.

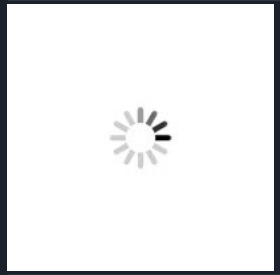






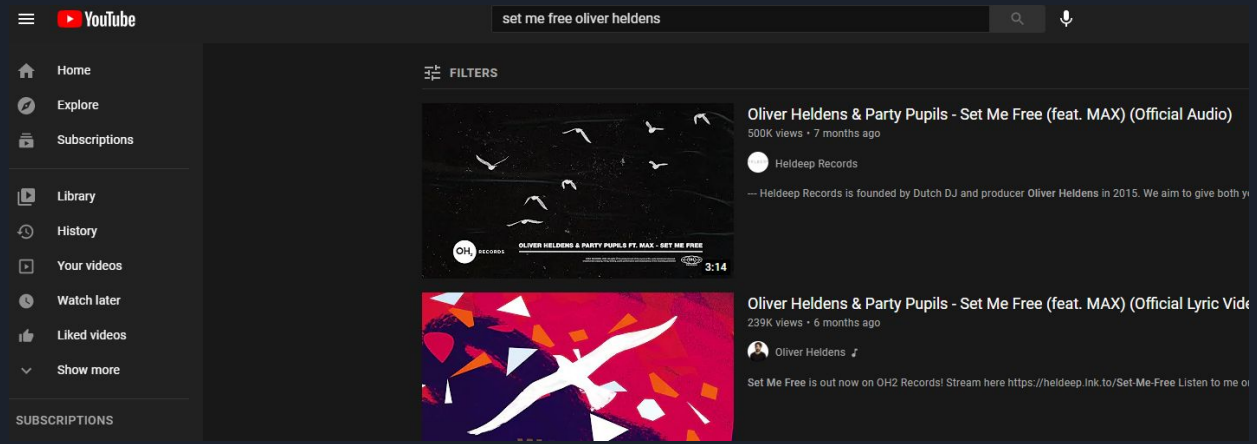
Input

The input in this situation is the words you put into the search bar, which is basically an instruction for the computer to look for your search.



Processing

The computer takes a minute to understand what you're trying to tell it, which is called processing (represented by the loading symbol).



Output

Finally, we reach the Output step, which is the last of these steps, and the Output is what the computer gives you, and in this situation, it's the results of your search.

What we have learnt so far: What a computer is, what code is, and what it can do

5 Minute Break

What we will learn next: How to get Python ready for coding on your computer

Installing Python Language and Pycharm, the code editor we will be using

Now we're going to get the a Python workspace open for you guys so that you can actually code in Python on your computer.

1. Identify what computer you have: Mac, or PC (Windows)
2. PC users stay in this main Zoom room
3. Mac users head over to Breakout Room 1 with Anil
4. Once you are finished, head back to the main room

THANK YOU FOR COMING!

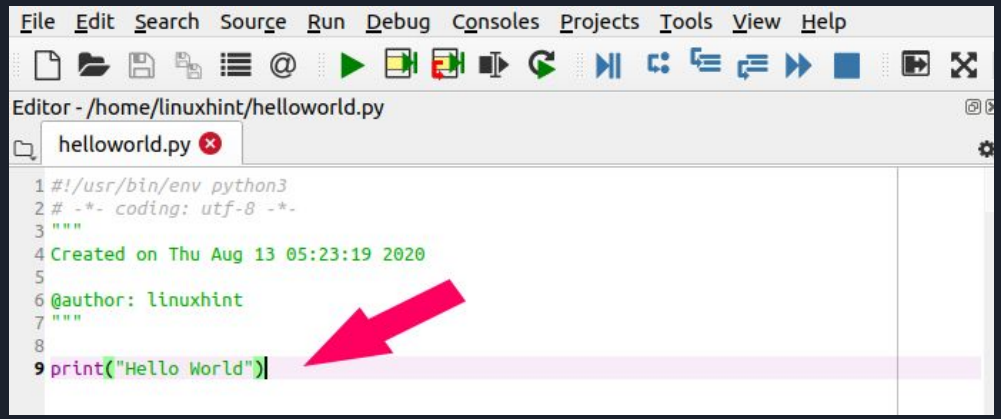
See you guys next Sunday!

Day 2: Getting started with Python: Basic Variables and the Print Statement

By Anil, Saketh, and Adit

Printing Statements in Python

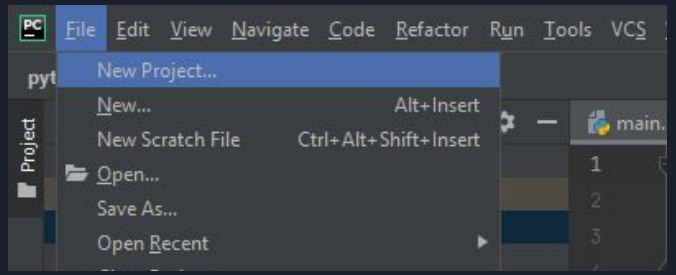
In the last class, we talked about input, processing, and output. One of the ways that we can get output from the computer is by using the print statement, which will output something in the run window

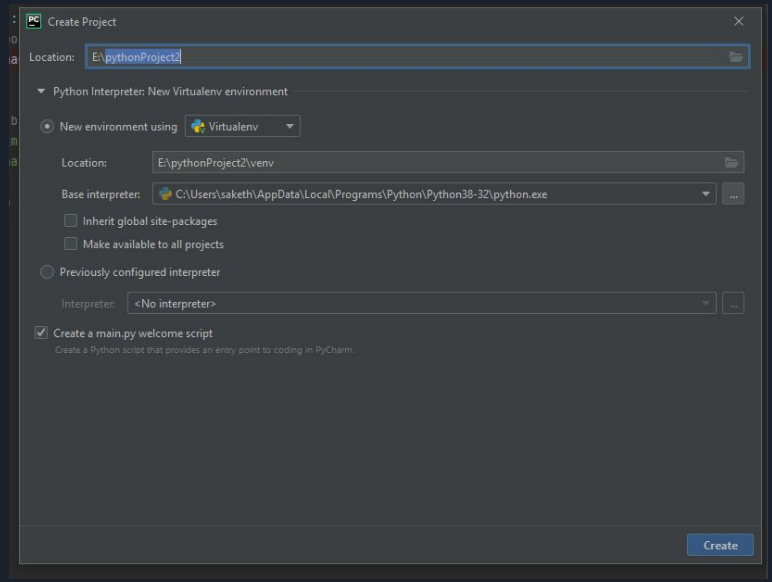


In this situation, the input is the statement print(“Hello World”), and if you were to run this program using the command listed above, then the computer would think and process what you told it, and then print out (or output) the words “Hello World.”

How can you use the Print statement too?

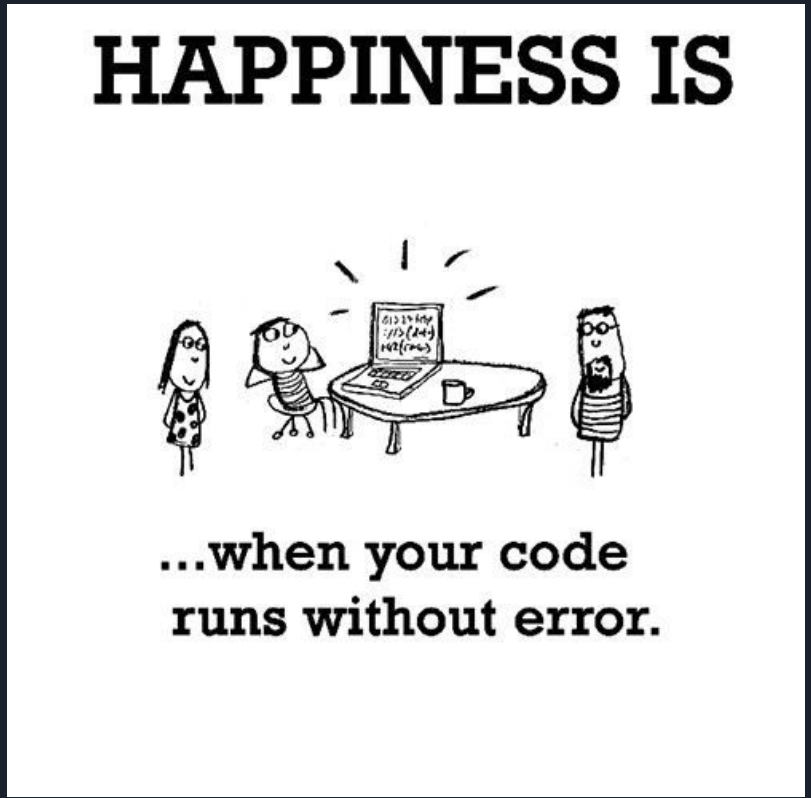
First, open PyCharm by clicking the icon that looks something like this:



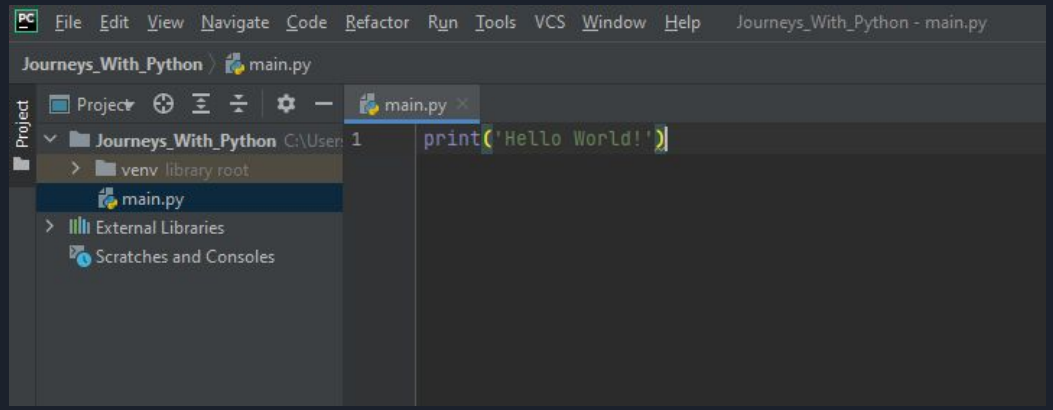


Head to the top left of the screen and click File and then New Project. You will be shown a menu, but all you have to do here is name the project journeys_with_python and you can just click the Create button at the bottom right without changing anything else.

CODING TIME!

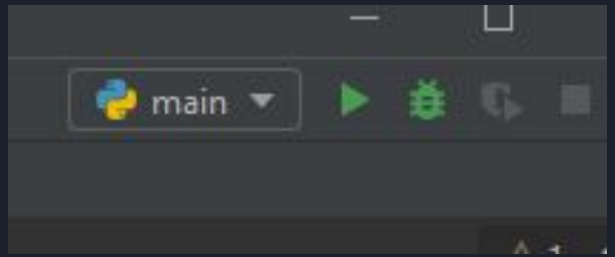


FINALLY we can now actually get coding!

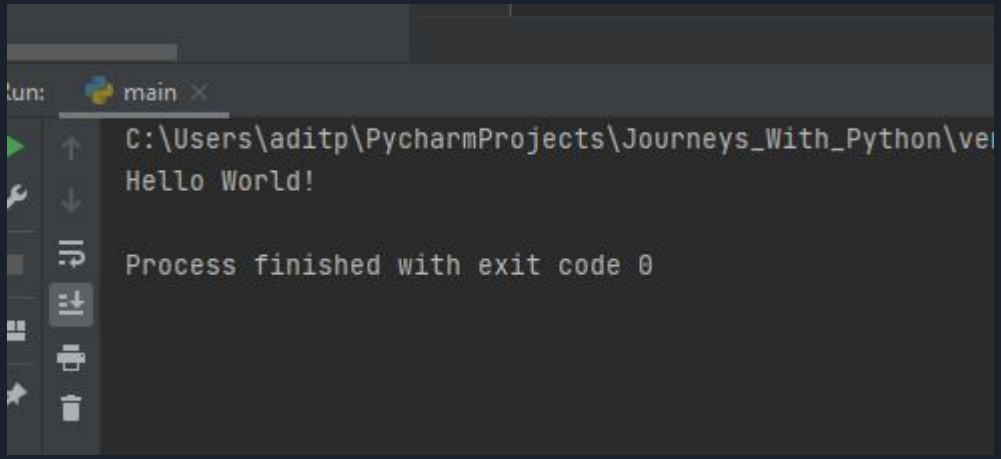


So, for a little bit of backstory, printing out the words Hello World! are the first thing anyone ever does when they learn a new programming language in the programming scene. It functions as a bit of a sign that the language is working properly, and is one of the easiest yet accomplishing thing to do when learning programming!

Lastly, you want to press this green play button in the top right corner of the screen, which will run the program



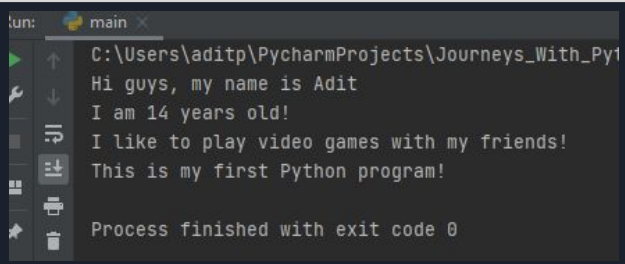
And sure enough, you should see the output of your first computer program with Python in the bottom part of the screen!



If you get this right, feel free to change the words in the quotes to whatever you'd like to make the computer print other things!

Challenge: See if you can figure out the code needed to make this output introducing yourself, telling how old you are, what you like to do, and that this is your first Python program!

Adit's example:



Answer to the Challenge:

```
print('Hi guys, my name is Adit!')
print('I am 14 years old!')
print('I like to play video games with my friends!')
print('This is my first Python program!')
```

Now let’s break this code down:

- The print command will print whatever is in the parentheses, so if we put the words ‘Hello World!’ in it, it will print just that, and ***move to the next line***
- Due to this, if we put multiple print statements like we did above, we get multiple sentences on different lines.

That's really it!

Now that you guys know how to use the print statement, let's move forward to variables, a concept that is very important in programming

5 MINUTE BREAK TIME

Variables in Code

Variables are one of the most important parts of Python and coding in general, so it is very important we learn about them, but they can also be a little complicated to understand, so here's an example: a variable is a BOX in which you can put anything in, whether that be a word, or a number, or a true or false value.

Now, we can put anything in a variable, but that anything has to fall in four categories, which are called data types.

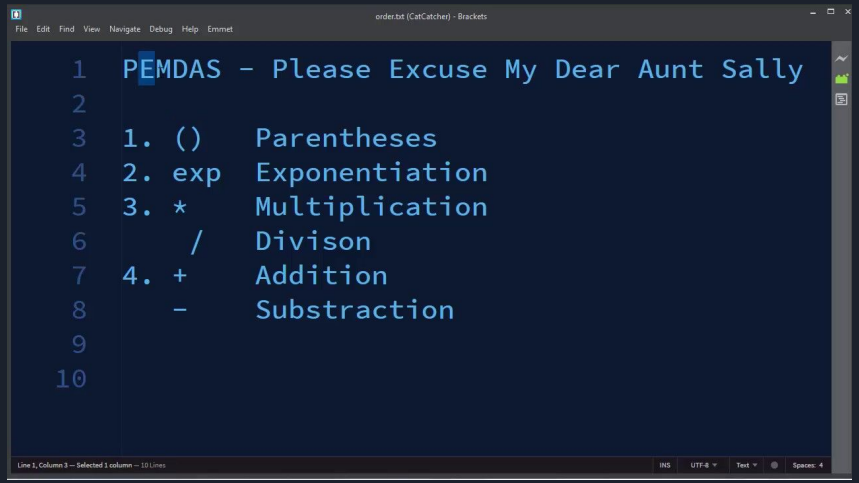
The four data types are called Integer, String, Boolean, and Float

Integers

Integers are any standard numbers, without decimal points or fractions, just like you would count.

Ex. 1, 100000, 6969699, 10239, -121

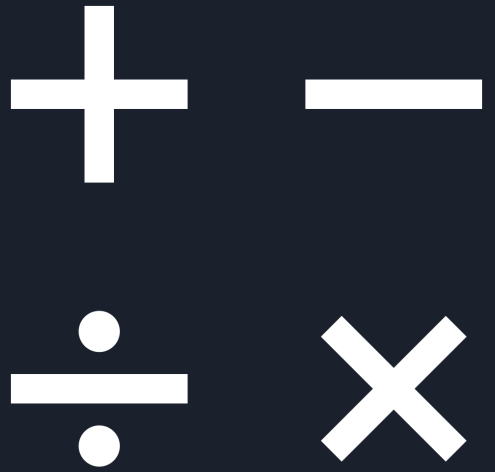
Integers are used for calculations that do not require a lot of pinpoint accuracy, but are still very useful in programs.

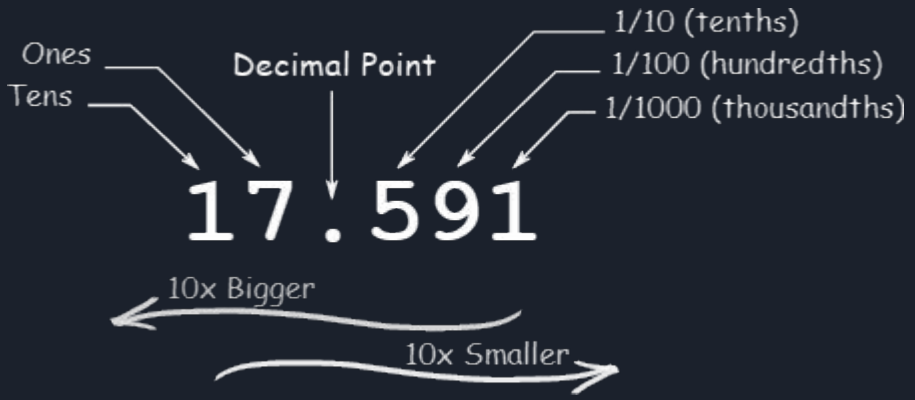


Float

Float, or floating point numbers are just like integers in that they are numbers, but the biggest thing that separates them is the fact that they have a decimal point in them

Ex: 1.1, 69.69, 456.908, 435345345346.54764





Strings

<- (Wrong string)

Strings are quite literally ANYTHING in quotes.

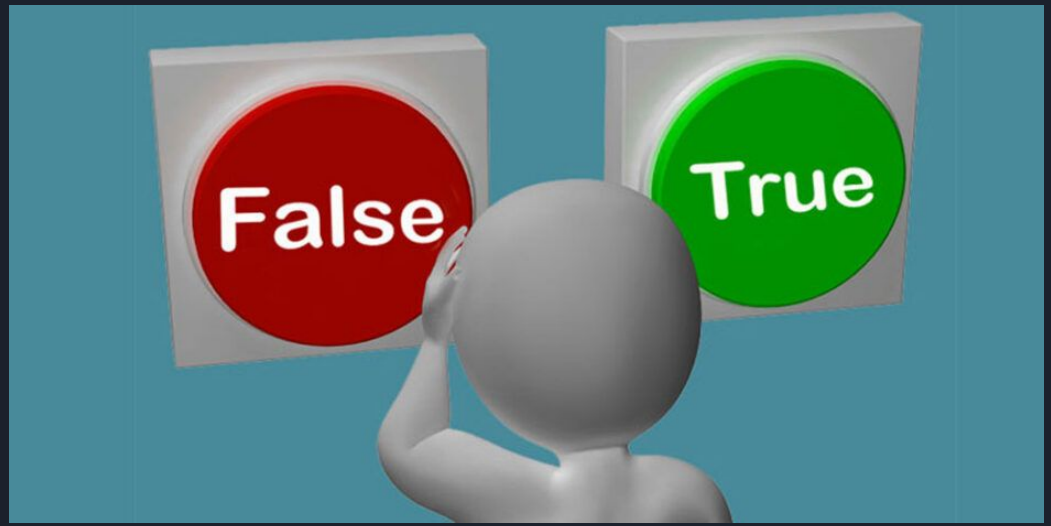
Ex. “Hello”, “1234”. “True”, “ASDF”, “12.24”, “#$%^@#$”, “c”

The reason that strings are called by their name is because just like a word is made up of different letters, a string is made up of different letters, numbers , or characters strung together

This can be in the form of words, phrases, numbers, decimal numbers, characters, or symbols

Boolean (pronounced boo-lee-uhn)

Boolean variables are a little interesting in that they only have two possible values: True and False. You can use booleans to test if something is true or false, and start working from there. We will not use boolean variables too often in this course, however once you start programming a little more complex programs, they will come to a lot of use



Creating your own variables

Follow these examples and try creating your own variables and printing them too!

You can use any appropriate name, and assign any value based on the rule for each data type.

*NOTE - Some names may not work, because they may be built in python commands. (For those using pycharm (most of the time), if the name of a new variable is not auto suggested, then you are okay)

main.py

```
integer = 3
floatingpoint = 6.9
string = 'This is a string'
boolean = True
```

```
1  integer1 = 3
2  str = "Hello World"
3  fl0at = 32.7
4  bool = True
```

Concatenating Strings

We know that the word Concatenation can sound a little bit intimidating and strange, but it is not nearly as complicated as you think it is!

Essentially, concatenation is adding two strings together and connecting them to make one long string. Think of it as taping one piece of thread that is blue to another piece of thread that is red, and the long piece of thread you get is the result of concatenation.

```
x = "Python is "
y = "Awesome!"
print(x + y)
```

```
a = "Python is "
b = "Awesome!"
c = a + b
print(c)
```

```
e = "Python is"
f = "Awesome!"
print(e + " " + f)
```

All these methods will print out the words Python is Awesome!

Methods 1 and 2

```
x = "Python is "
y = "Awesome!"
print(x + y)
```

```
a = "Python is "
b = "Awesome!"
c = a + b
print(c)
```

As we learned earlier, a variable is a little box that we can store different numbers, letters, or characters in. Here, the letters x, ,y, a, b, and c are basically labels that are put on the boxes so that we know what they have in them.

In our examples, we have two boxes that contain the words “Python is “ in the first box and “Awesome!” in the second box.

As you can see, if you combine what is in the two boxes, then you get a sentence that reads: Python is Awesome!

One way that we can do that is by “adding” the two boxes together with the + sign and printing that out. The second example is basically the same as the first one except we create a new box with the sum of the two boxes a and b called c and print that one out.

Method 3

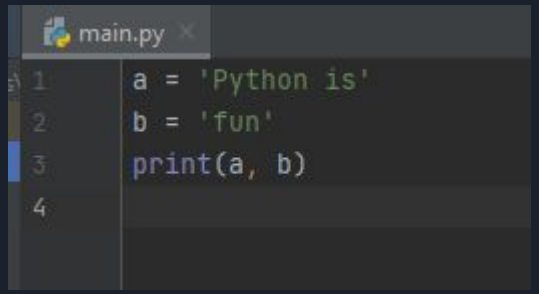
```
e = "Python is"
f = "Awesome!"
print(e + " " + f)
```

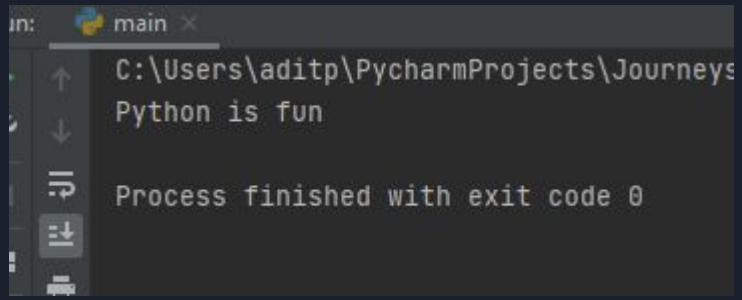
Right off the bat, we can see that method three is a little bit different. The first thing that comes to mind is how on the third line, we are also adding a space in between our two boxes. Why is that?

Well, in the last example, one of our boxes already had a space inside of it, but in this case, we don't have one, so to take care of that, we use one more + sign and add a space in between the two boxes. If we didn't add that space and just wrote print (e + f), then Python would print out "Python isAwesome!" and that is because you haven't told the computer to separate the words is and Awesome.

Since we added the space, Python knows that it has to separate the words is and Awesome so if we add that space, we get the correct print-out, which is "Python is Awesome!

Concatenation with Commas





Most of the time when we put strings and variables together though, we use commas. Commas in Python are useful for putting in between strings, since they automatically add a space between the two things being added together, removing the need for adding janky spaces after each string.

Even though we have covered using plus signs, commas are usually the more used way, since it allows for easy ability to print multiple things

GAME TIME!

Join this kahoot id for a quick game to see if you guys know your data types! The winner gets to play a YouTube video or song of their choice for the rest of the students !

Day 3: Concatenation and Quiz

Time to Start off with a quick quiz game!

Join the kahoot to test your understanding of data types, a concept we learned last class!

Concatenating Strings

We know that the word Concatenation can sound a little bit intimidating and strange, but it is not nearly as complicated as you think it is!

Essentially, concatenation is adding two strings together and connecting them to make one long string. Think of it as taping one piece of thread that is blue to another piece of thread that is red, and the long piece of thread you get is the result of concatenation.

```
x = "Python is "
y = "Awesome!"
print(x + y)
```

```
a = "Python is "
b = "Awesome!"
c = a + b
print(c)
```

```
e = "Python is"
f = "Awesome!"
print(e + " " + f)
```

All these methods will print out the words Python is Awesome!

Methods 1 and 2

```
x = "Python is "
y = "Awesome!"
print(x + y)
```

```
a = "Python is "
b = "Awesome!"
c = a + b
print(c)
```

As we learned earlier, a variable is a little box that we can store different numbers, letters, or characters in. Here, the letters x, ,y, a, b, and c are basically labels that are put on the boxes so that we know what they have in them.

In our examples, we have two boxes that contain the words “Python is “ in the variable x and “Awesome!” in the variable y.

As you can see, if you combine what is in the two boxes, then you get a sentence that reads: Python is Awesome!

One way that we can do that is by “adding” the two boxes together with the + sign and printing that out. The second example is basically the same as the first one except we create a new box with the sum of the two boxes a and b called c and print that one out.

Method 3

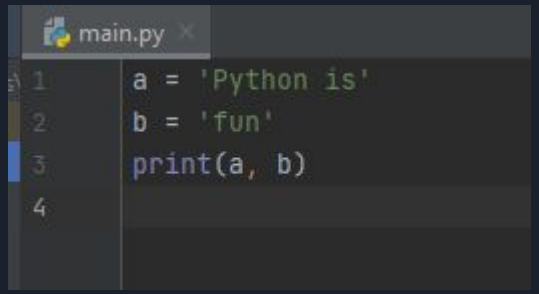
```
e = "Python is"
f = "Awesome!"
print(e + " " + f)
```

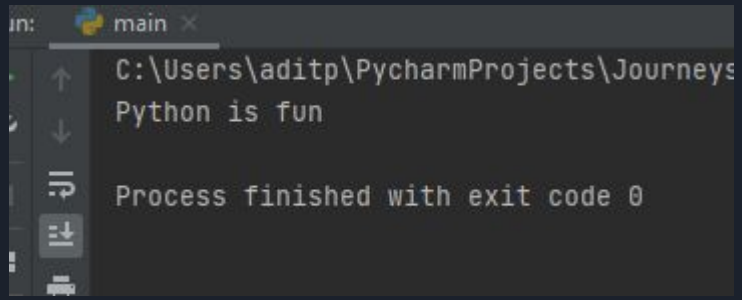
Right off the bat, we can see that method three is a little bit different. The first thing that comes to mind is how on the third line, we are also adding a space in between our two boxes. Why is that?

Well, in the last example, one of our boxes already had a space inside of it, but in this case, we don't have one, so to take care of that, we use one more + sign and add a space in between the two boxes. If we didn't add that space and just wrote print (e + f), then Python would print out "Python isAwesome!" and that is because you haven't told the computer to separate the words is and Awesome.

Since we added the space, Python knows that it has to separate the words is and Awesome so if we add that space, we get the correct print-out, which is "Python is Awesome!

Concatenation with Commas





Most of the time when we put strings and variables together though, we use commas. Commas in Python are useful for putting in between strings, since they automatically add a space between the two things being added together, and this gets rid of the space at the end of a string.

Even though we have covered using plus signs, commas are normally the more popular method, since it allows for easy ability to print multiple things.

The way that we use this method is simply by just taking the two variables, indicating that we want them together with a comma and a space in the print statement, just as shown above.

Now try this Challenge Problem!

Create a program which outputs something like this:

```
Hi my name is Anil and I am 15 years old
```

You must use string concatenation to concatenate your name and age to the rest of the sentence.

Hint: Everything instead of the name and age can just be printed in a print statement.

Hint 2: Make the age a string, not an int.

Bonus: Try making the age an int, and run the project. What happens? Why did that happen?

Break

5 MINUTE BREAK

Quiz Time!

https://forms.gle/s3BALCGpxsXLSphB9

Day 4: Loops

By Anil, Adit and Saketh

Before we talk about loops, lets review the quiz from last class

What does a computer understand *

- ( ) Numbers
- ( ) Only the alphabet
- (•) 1 and 0
- ( ) The alphabet and numbers

What are the 3 steps of how a computer works? *

- ( ) Question, Work, Answer
- (•) Input, Processing, Output
- ( ) Code, Computation, Result
- ( ) Hamburger, Fortnite, Among us
- ( ) Initialization, Decipher, Ejection

How would you print "The sky is blue." in python? *

print("The sky is blue.")

What are all the data types? *

- [ ] Char
- [x] Integer
- [x] Boolean
- [ ] Function
- [x] Float
- [ ] Random Data
- [x] Sring <- Should be String
- [ ] Symbol
- [ ] Hardware Accelerator Data
- [ ] Termalpayste Data

Which of these variables will NOT work? *

- [ ] ProGamer = "Adit"
- [x] #1Coder = "Saketh"
- [x] 15yearsold = "Anil"
- [ ] GoodStudents = "You guys!"
- [x] print = "this is my variable"

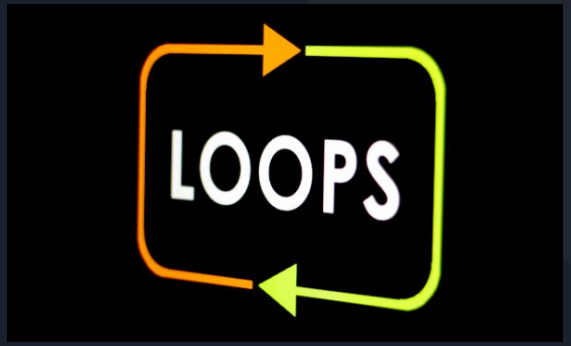
What will the following code output? *

```
name = "Jeff"
age = "11"
print("Hi my name is "+name+" and I am "+age+" years old!")
```

Hi my name is Jeff and I am 11 years old!

Break Time

Loops



First of all what even are loops?

Loops are simply put, any code that repeats itself multiple times.

So far, we have learned about print statements and variables, but those only help with outputting something once

For Loops

To first understand why we use for loops, we'll have to understand the struggles of not being able to use loops.

See if you can write code in PyCharm to get the end result shown in the screenshot right below.

Obviously, you should replace both your name and your age with your own instead of printing out my information.

```
E:\pythonProject3\venv\Scripts\python.exe E:/pythonProject3/main.py
Hi! My Name is Saketh and I am 14 years old.
Hi! My Name is Saketh and I am 14 years old.
Hi! My Name is Saketh and I am 14 years old.
Hi! My Name is Saketh and I am 14 years old.
Hi! My Name is Saketh and I am 14 years old.

Process finished with exit code 0
```

Answer

If you did this right, then your code should look something like this:

```
print("Hi my name is Saketh and I am 14 years old.")
print("Hi my name is Saketh and I am 14 years old.")
print("Hi my name is Saketh and I am 14 years old.")
print("Hi my name is Saketh and I am 14 years old.")
print("Hi my name is Saketh and I am 14 years old.")
```

As you can see, this code is very repetitive to type and takes too much time to do a very simple task.

Thankfully, there is a much easier way to get this job done called the loop.

Loops

In Python, there are two types of loops.

The for loop:

```
for x in range(5):
    print("Hi! My name is Saketh and I am 14 years old.")
```

The for loop here is pretty simple, and the way it works is even simpler.
Essentially, we are telling python to repeat whatever is in the for loop 5 times. That is what the “range(5)” does. The “for x in” part is just how to initialize a for loop.

IMPORTANT!!!: Do not forget the “:” at the end of the beginning of the loop. If you do not add this, there will be an error.

And the while loop:

```
y = 0
while y < 5:
    print("Hi! My name is Saketh and I am 14 years old.")
    y = y + 1
```

This loop is a bit different than the for loop, considering that we have the variable y active. Before setting up the while loop, we have to initialize the variable y as 0, whereas in the for loop this step would just be done with the x in “for x in range”. This is important because every time that the while loop finishes one cycle, it makes the value of x one number higher, so by the time it is finished, the value of x is 5. The reason we say “while y < 5”is because by the time y has reached 5, the code inside of the loop has run five times, and the job has been finished. Simply put, the while loop will keep continue running until it reaches its stop condition.

Some rules about loops

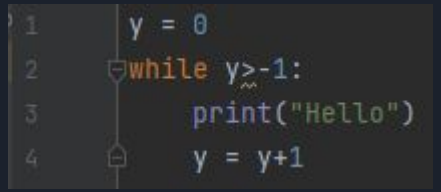
FOR LOOPS

The range value can be 0 or a negative number, but nothing will output.

WHILE LOOPs

Watch out for something called “infinite looping”

That is basically when the stop condition is never reached, so the while loop just keeps on going and going. If you ever accidentally do this, press this red box at the top right of pycharm



Try it for Yourself!

Try copying the loops into Pycharm, and run it. Next, try changing the range value in the for loop. What happens?

Try changing values of numbers in the while loop. What happens?

Without trying the code and based on what you learnt about how loops work on the last slide, tell us one of the instructors in the private chat what you think is going to happen if you make the numbers in for x in range or while < lower or higher.

Conditionals and User Input

By Adit, Saketh and Anil

What are Conditionals?

Conditionals are what they sound like: Conditions!

Conditionals are a condition that has to be fulfilled for a certain thing to happen, with a backup plan in case that thing does not happen

How do you write conditionals?

Conditionals are written with IF and Else Statements

```
X = 3
If X >2:
        print("X is greater than 2")
Else:
        print("X is not greater than 2")
```

This is a very simple example.

It is basically saying: If x is greater than 2 then print("X is greater than 2") but else, print("X is not greater than 2")

Again, you need the colon, just like with loops, for both the if and the else.

Things to remember about Conditionals

IMPORTANT: You DON'T have to use an else statement if you don't need it. A program can work with just an if statement if that is all that is needed.

```
X = 3
If X >2:
        print(“X is greater than 2”)
```

This program will work fine with no errors.

What do you think this program will do if X = 5?
What do you think this program will do if X = 1?

This is where **boolean** variables come into play

For example, take the following program:

```
person = 'Adit'

person1 = 'Anil'

if  person == 'Adit':

        print ('hello', person)

else:

        print ('hello', person1)
```

Remember: You need to use '==' when comparing things, but '=' when defining a variables. You will get an error if you compare things with only '='

Since Python is pretty close to normal speaking language, try to guess what this program will do!

That’s right! It will check if the value of the `person` variable is ‘Adit’, and print hello, Adit if it is, and if it isn't, it will print hello, Anil, since that is the value of the other variable

Now let’s try a condition with numbers

```
x=6
if x>5:
    print ('x is greater than 5')
else:
    print ('x is less than or equal to 5')
```

Operators in Python

<→ less than

> → greater than

>= → greater than or equal to

<= → less than or equal to

== → equal to

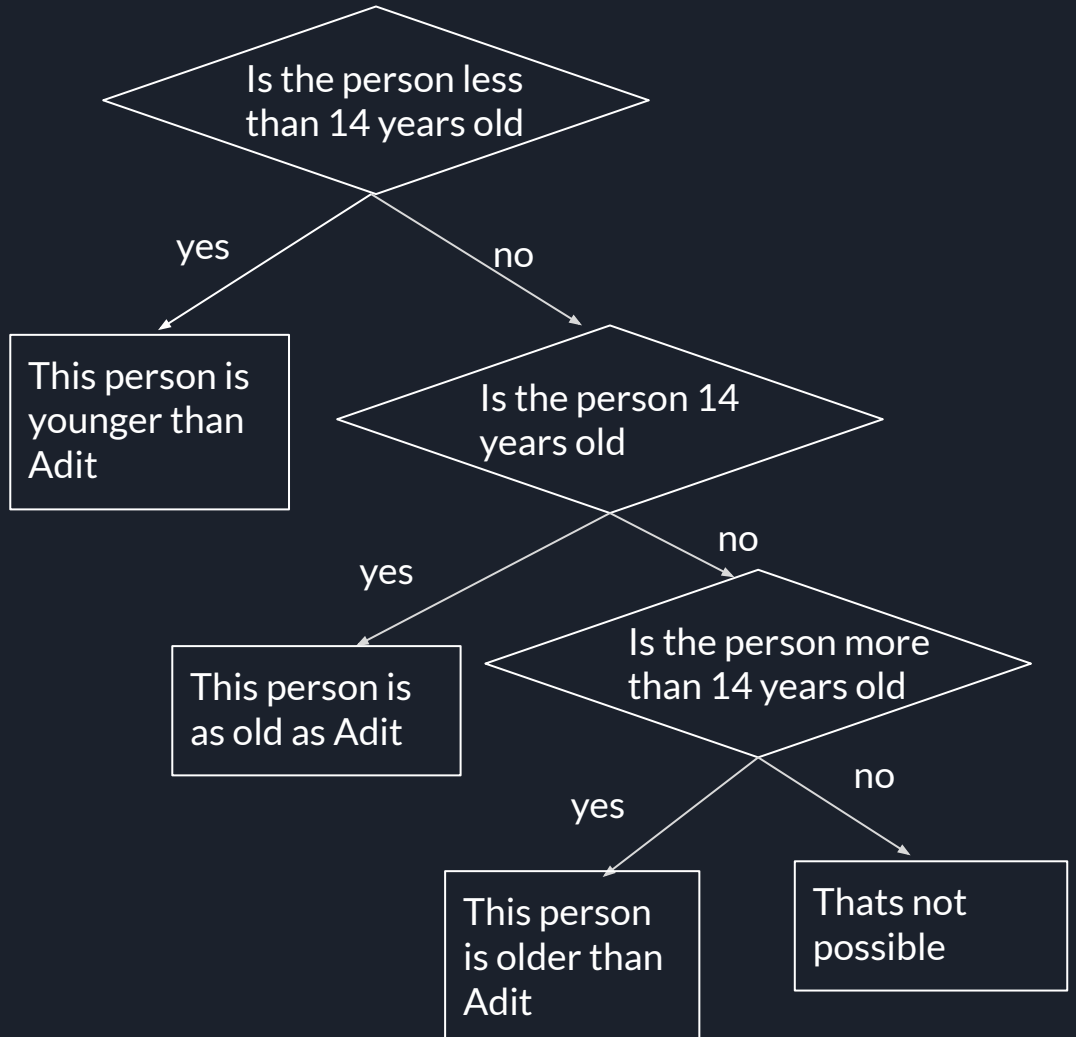
!= → not equal to

We can use these operators to help us create conditions that will help us check things and do things based on that
KEEP IN MIND that you can only compare numbers to numbers and strings to strings, and not strings to numbers.

Nested Conditions

```
age = 14
if age < 14:
    print ('this person is younger than Adit' )
elif age == 14:
    print ('this person is just as old as Adit' )
elif age > 14:
    print ('this person is older than Adit' )
else:
    print ('thats not possible' )
```

What would that code print, and why?


Is the person less than 14 years old
yes
no
This person is younger than Adit
Is the person 14 years old
yes
no
This person is as old as Adit
Is the person more than 14 years old
yes
no
This person is older than Adit
Thats not possible

Any questions?
If not, 5 minute break

Time to get into user input!!

The Python input function

Getting user input in Python is really easy!

All you have to do is create a variable, and set its value to the in-built input function, and set the parameter of the function to be your question!

For example:

```
name = input ('Please enter your name:')

print ('Hello, ', name)
```

Run this code and tell one of the teachers what it does!

Bonus: Explain how

KEEP IN MIND that the input() function by default will create a string variable!!!! If you want to take in a integer or a float from a user, use the int() and float() functions like so:

```
age = int (input ('Please enter your age: '))
```

This program will get the age of the user, and store it in the integer variable age

Now, club all we learnt today together to make a password checker!

Write a program that has a pre-set 5 letter password, then takes a input from the user, and checks if it is the password

Feel free to ask for help!

Any Questions?

Feel free to ask!

If you don’t have any more questions, goodbye!
See you next class, you did really well today!

Day 6: Final Project

Alright guys, that's it!

You've made it through the Journeys with Python beginner Python workshop!

Just before you go, though, we thought you should put your new skills together and make something you can be proud of.

Project Prompt

You will be making a random number quiz game for your final project. The basic requirements are as listed:

- Should make the user guess a random number between 1 and 100
- If the guess is too high, the program should tell the user to guess lower
- If the guess is too low, the program should tell the user to guess higher
- If the number is outside of the range of 1 to 100, then the program should tell the user that the number is outside the range.
- Give the user multiple tries
- Use at least one type of loop, conditional, variable, user input

You can also add bonus to the project if you are finished with the basic early, which can include, but is not limited to:

- Letting the user input the range of the random number
- Letting the user input how many tries they would like to use
- Making the program more user friendly and interesting

How do you even generate a random number?

```
import random

x = random.randint(0,100)

print (x)
```

Try running the code above!

What it does

1. Import the in-built random library
    a. A library is a like a folder with many functions and commands that can be done.
2. Define a variable called x and assign it a value of a random integer between 0 and 100
3. Print that number so you can see it

The “ . “ is used to tell the computer to access the “randint” function from the “random library”

Now that you know how to generate a random number, you can use this to start your final project

Feel free to ask any of the instructors any questions you have, and we will help you to the best of our ability

Links to all slides, if needed.

Day 1 - Intro

Day 2 - Variables & Print

Day 3 - Concatenation

Day 4 - Loops

Day 5 - Conditionals & Input

GOOD LUCK!