WhiteHat Jr
Live online Coding for Kids

Topic	INTRODUCTION TO PYTHON	
Class Description	**The student will be introduced to the basics of Python programming language along with its syntax.**	
Class	**PRO C97**	
Class time	**45 mins**	
Goal	● Introduction to Python programming. ● Learn basic syntax of the Python language. ● Numeric Data Type, Conditionals, Functions in Python.	
Resources Required	● Teacher Resources: ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ○ Smartphone ● Student Resources: ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen	
Class structure	**Warm-Up** **Teacher-Led Activity 1** **Student-Led Activity 1** **Wrap-Up**	**10 mins** **10 mins** **20 mins** **05 mins**
Credit	**Python 3.9.6** product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).	
WARM-UP SESSION - 10 mins		

<table>
<tr><td colspan="2">**Teacher Starts Slideshow**
Slide 1 to 4
Refer to speaker notes and follow the instructions on each slide.</td></tr>
<tr><td>Hey &lt;student's name>. How are you? It’s great to see you! Are you excited to learn something new today?
Following are the WARM-UP session deliverables:
● Greet the student.
● Revision of previous class activities.
● Quizzes.</td><td>**ESR**: Hi, thanks!
Yes I am excited about it!
Click on the slide show tab and present the slides</td></tr>
<tr><td colspan="2">**WARM-UP QUIZ**
Click on In-Class Quiz</td></tr>
<tr><td colspan="2">**Continue WARM-UP Session**
Slide 5 to 22</td></tr>
<tr><td colspan="2">**Following are the session deliverables:**
● Appreciate the student.
● Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.</td></tr>
<tr><td>Today we are going to start learning a new programming language,called **Python**.
Do you know what the meaning of the word Python is?
Yeah! That’s right!
You must be wondering why the name of the programming language is similar to the animal name.
Can you make a guess?</td><td>**ESR**: It is the name of a snake.
ESR: Varied.</td></tr>
</table>

Well the person who was developing this language, was reading a script from “**Monty Python's Flying Circus**”, a comedy series in BBC from the 1970s, and that’s when it struck him to keep the name of the language, simple and unique such as “Python”([Check the documentation](#)). So I guess there wasn’t much of a mystery in the name of the language. Python is easy to learn, and we can do almost anything using Python. We will start with learning the basics. Are you excited? Let’s get started.	**ESR**: Yes.

Teacher Ends Slideshow

TEACHER-LED ACTIVITY - 10 mins

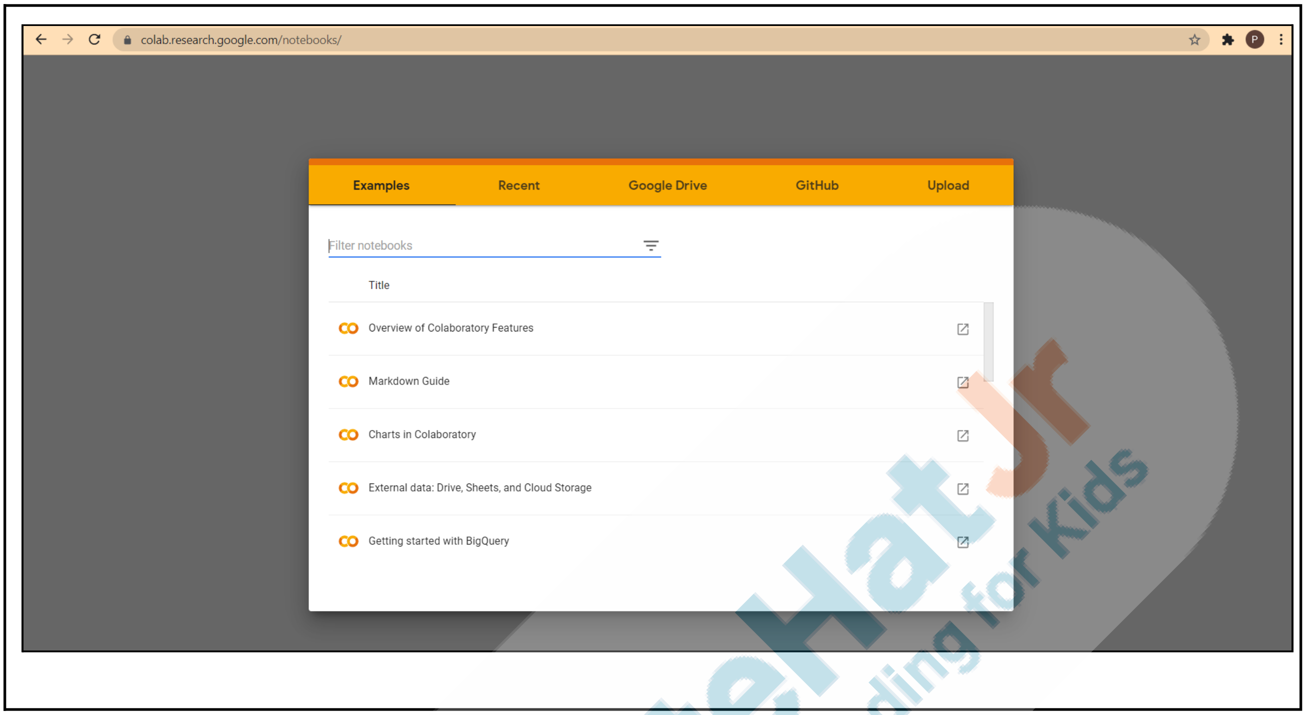
Teacher Initiates Screen Share

ACTIVITY

- **Introduction to Google Colab.**
- **Basic Python syntax.**
- **Python Numeric Data Types.**
- **Functions**
- **Write a function to convert temperature from degree Celsius to Fahrenheit.**

Teacher Action	Student Action
Can you tell me what’s the first thing we need to start coding in any programming language?	**ESR:** We need a platform to run the code.

***Note**: Let the student think and come up with his/her own answer.* Superb! We need an environment/platform, either online or offline, to run the code. There are many platforms available to run the Python programming language code. We will begin with an online platform that runs on the browser while you are connected to the internet, called "**Google Colaboratory**" developed by **Google**. In short, we will call it **Colab**. It is easy to use and comes with many pre-installed features, which we will explore as we learn. Let's first understand **Colab**. *The teacher opens the link **[Teacher Activity 1]**.* 1. Open the Google Colab link. ***Note 1**: This pop up comes with a list of these reference notebooks for users.* ***Note 2**: If the same list is not visible, open the link in the incognito, to see the same list.*	



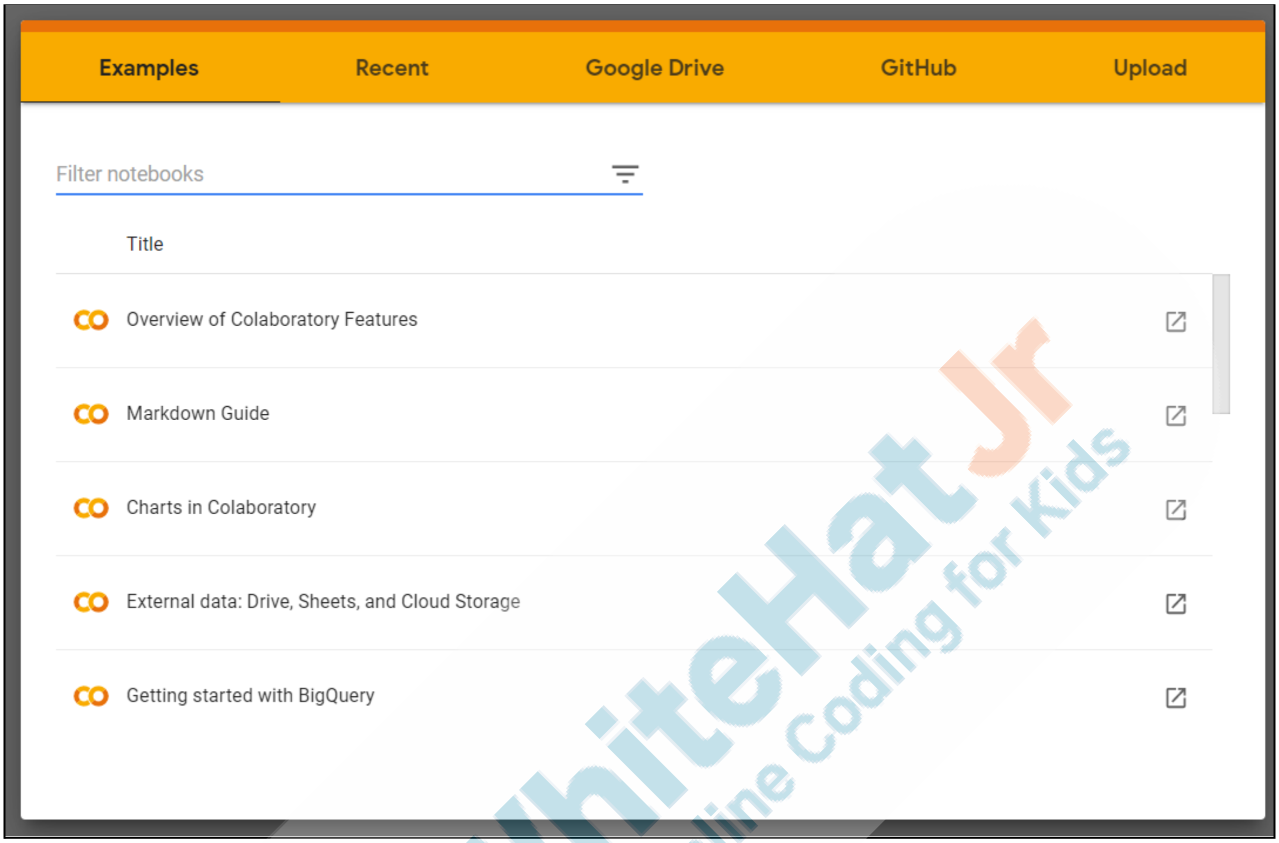
Overview of Colaboratory Features

Markdown Guide

Charts in Colaboratory

External data: Drive, Sheets, and Cloud Storage

Getting started with BigQuery



Title

Overview of Colaboratory Features

Markdown Guide

Charts in Colaboratory

External data: Drive, Sheets, and Cloud Storage

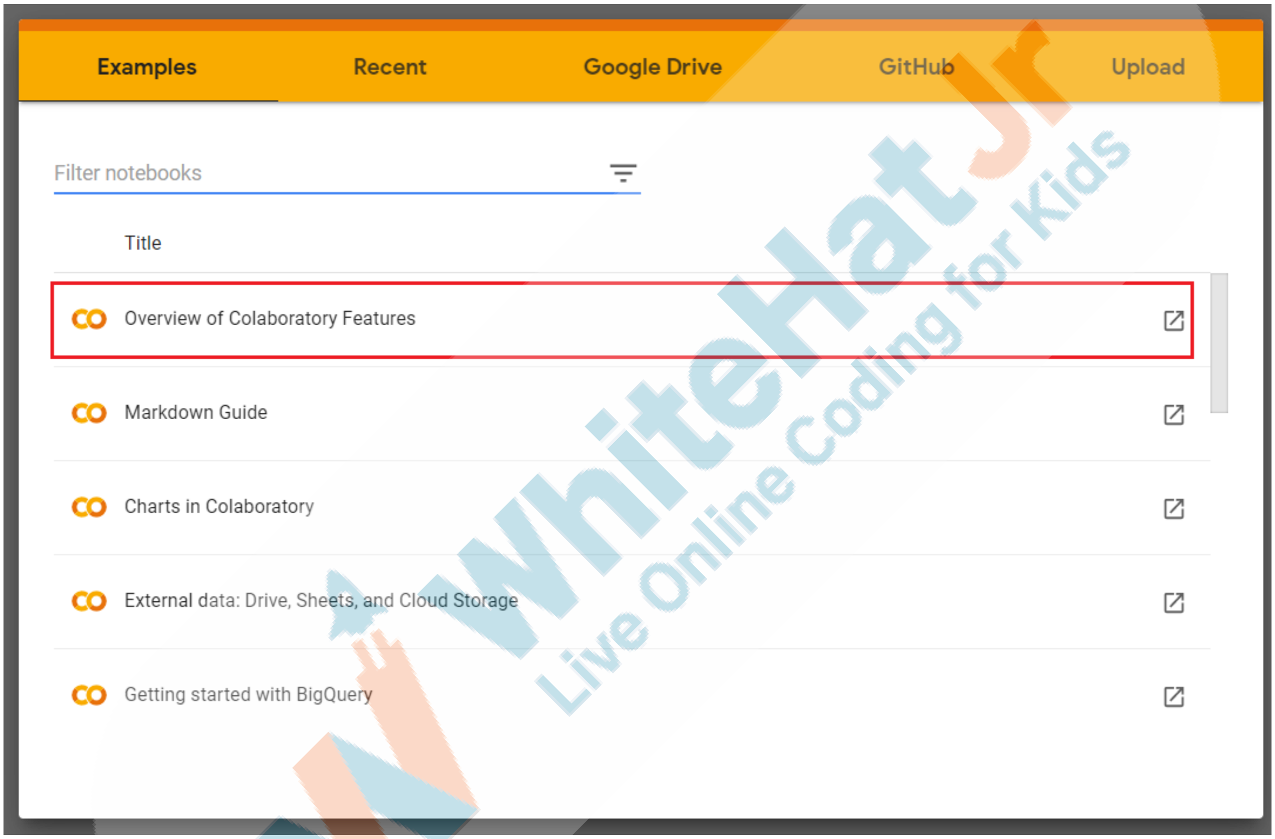
Getting started with BigQuery

Once you open the link, you will see a pop-up with a list of **colab notebooks**.	
What do you think a notebook is?	**ESR**: Varied.
In general, the notebooks we have been using in schools are to take notes, right?	**ESR**: Yes.
But, colab notebooks are **interactive digital documents** which can have **code**, **text**, **media** files, etc all together at one place, that can be shared easily with anyone.	
Now let's see how a notebook looks.	

The teacher opens the first link in the pop-up.

***Note 1**: This pop up comes with a list of these reference notebooks for users.*

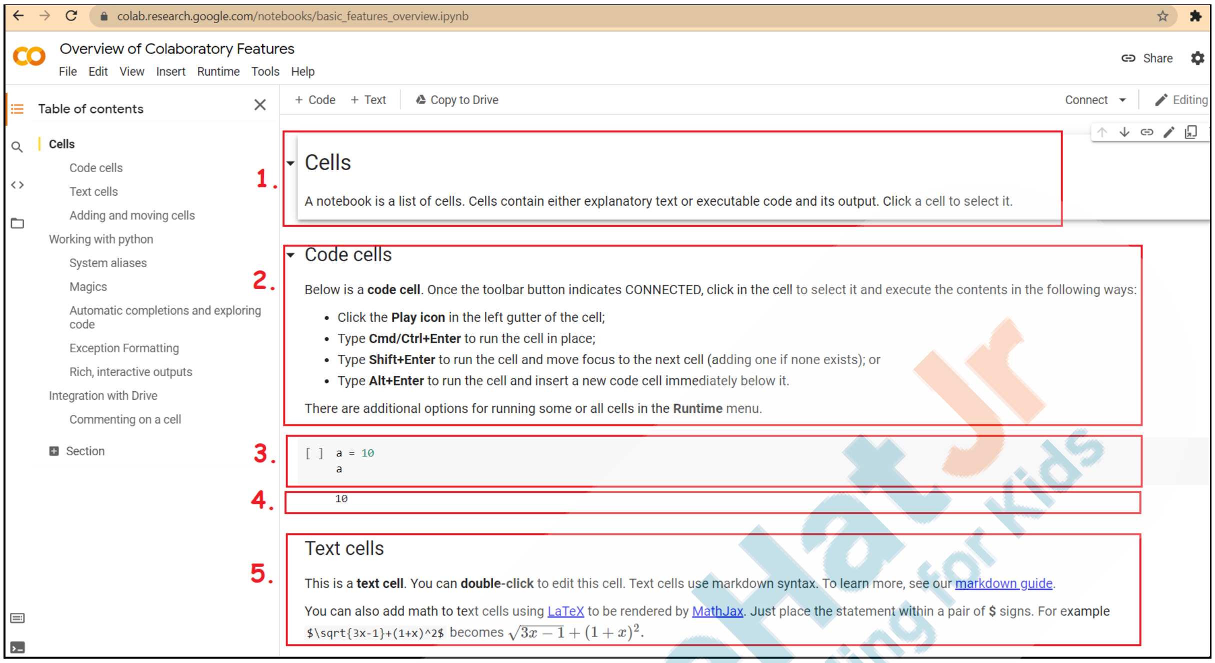
***Note 2**: If the same list is not visible, open the link in the incognito, to see the same list.*



Title

- Overview of Colaboratory Features
- Markdown Guide
- Charts in Colaboratory
- External data: Drive, Sheets, and Cloud Storage
- Getting started with BigQuery

Once you open the notebook link, “**Overview of Colaboratory Features**” you will see some **code cells** and **text cells**.



Cells

A notebook is a list of cells. Cells contain either explanatory text or executable code and its output. Click a cell to select it.

Code cells

Below is a **code cell**. Once the toolbar button indicates CONNECTED, click in the cell to select it and execute the contents in the following ways:

- Click the **Play icon** in the left gutter of the cell;
- Type **Cmd/Ctrl+Enter** to run the cell in place;
- Type **Shift+Enter** to run the cell and move focus to the next cell (adding one if none exists); or
- Type **Alt+Enter** to run the cell and insert a new code cell immediately below it.

There are additional options for running some or all cells in the **Runtime** menu.

```
[ ]  a = 10
     a
```

```
     10
```

Text cells

This is a **text cell**. You can **double-click** to edit this cell. Text cells use markdown syntax. To learn more, see our markdown guide.

You can also add math to text cells using LaTeX to be rendered by MathJax. Just place the statement within a pair of $ signs. For example `$\sqrt{3x-1}+(1+x)^2$` becomes $\sqrt{3x-1}+(1+x)^2$.

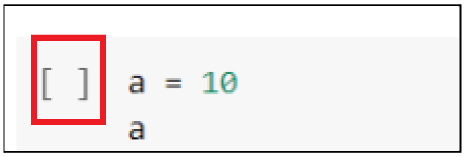
***Note**: Let the student read the instructions in the notebook shared through your screen and help them understand what it means.*

- A **Cell** in a notebook is a container where you can write your **code** or **text**. One cell can be of either code or text, not for both at the same time. (In image above number 1, 2, 3 and 5 are all cells in the notebook.)

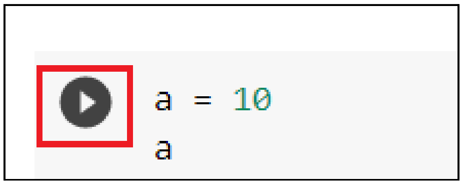
- **Code cells** are where we write the Python code.(In the image above number 3 is the code cell, 4 shows the output once we run the cell 3 in the image above.)

    - **Running a code cell**:

        - Use the **play button** which shows up on the top left of the code cell.



```
[ ]  a = 10
     a
```



```
a = 10
a
```

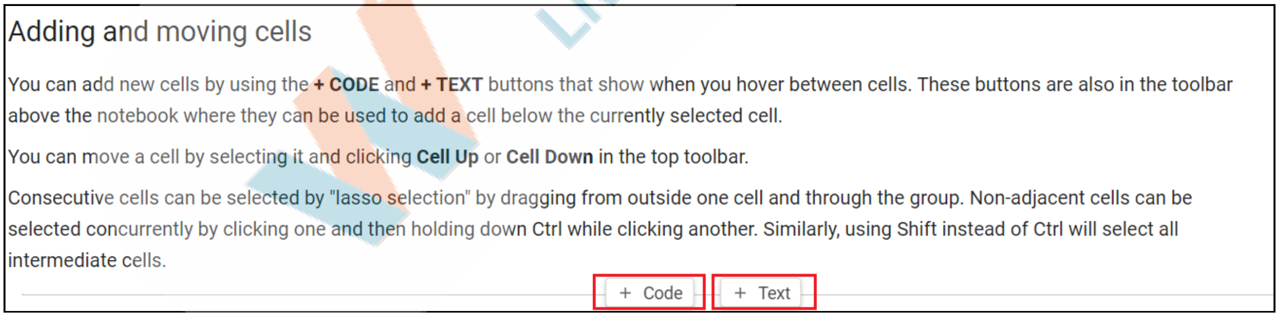
  - Click on the cell and use **Cmd/Ctrl+Enter.**

- A **text cell** is where you can add simple text using markdown syntax.(In the image above, 2 and 5 are text cells)

***Note**: Help the student to recollect the markdown editors they have used before and how they wrote reflection notes.*

Adding a NEW cell:

- To add new cells we can use the **+CODE** and **+TEXT** buttons that appear when we hover in between any two cells.



Adding and moving cells

You can add new cells by using the **+ CODE** and **+ TEXT** buttons that show when you hover between cells. These buttons are also in the toolbar above the notebook where they can be used to add a cell below the currently selected cell.

You can move a cell by selecting it and clicking **Cell Up** or **Cell Down** in the top toolbar.

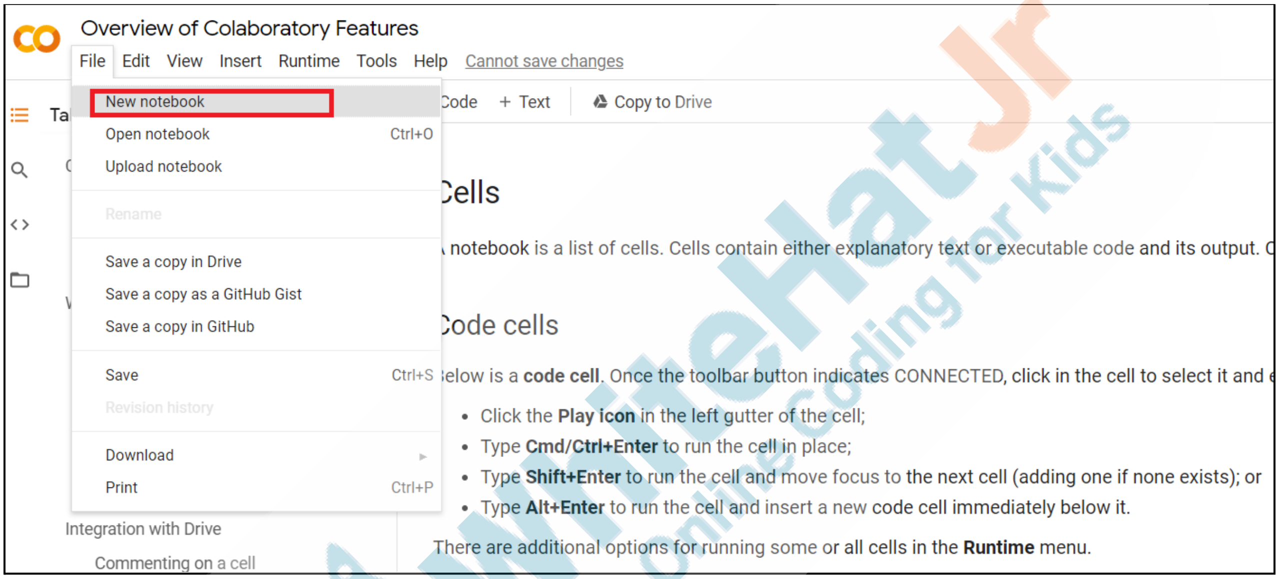
Consecutive cells can be selected by "lasso selection" by dragging from outside one cell and through the group. Non-adjacent cells can be selected concurrently by clicking one and then holding down Ctrl while clicking another. Similarly, using Shift instead of Ctrl will select all intermediate cells.

Those were pretty simple steps to use in the notebook, right? Now that we are little familiar with notebooks, let's create	**ESR:** Yes.

our own colab notebooks and get started with Python programming!

Creating a NEW Collab notebook:

- Go to “**File**” in the top left corner of the notebook.
- Click on “**New notebook**”.



Cells

notebook is a list of cells. Cells contain either explanatory text or executable code and its output.

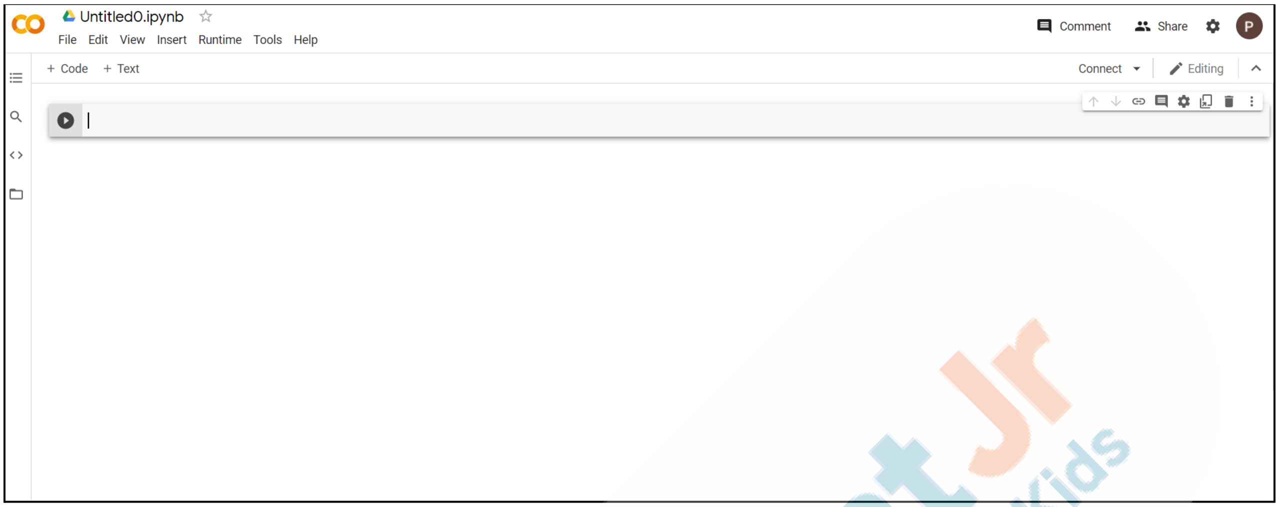
Code cells

elow is a **code cell**. Once the toolbar button indicates CONNECTED, click in the cell to select it and

- Click the **Play icon** in the left gutter of the cell;
- Type **Cmd/Ctrl+Enter** to run the cell in place;
- Type **Shift+Enter** to run the cell and move focus to the next cell (adding one if none exists); or
- Type **Alt+Enter** to run the cell and insert a new code cell immediately below it.

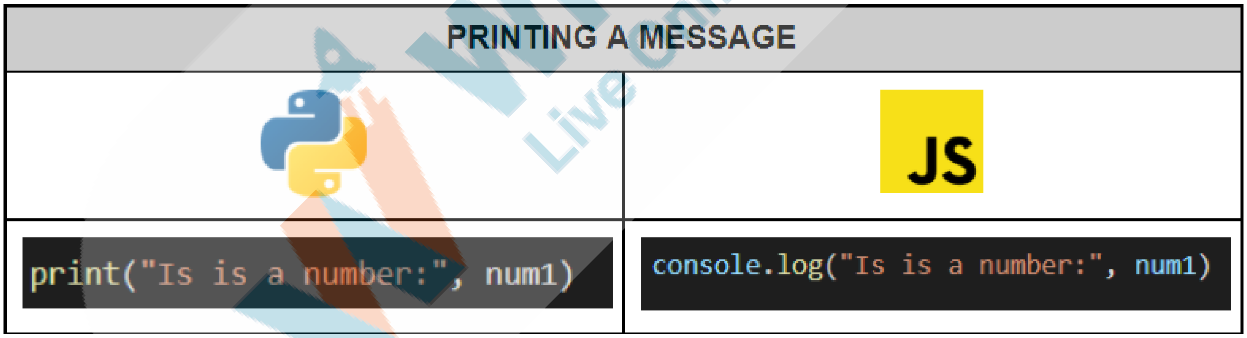
There are additional options for running some or all cells in the **Runtime** menu.

An Empty Colab Notebook with one code cell:



Since we are already familiar with basic concepts of programming using JavaScript, let's understand the Python syntax(the way we write the code in any programming language) using those concepts first.

Python vs JavaScript Syntax reference document: *Teacher Activity 2*.



PRINTING A MESSAGE

```
print("Is is a number:", num1)
```

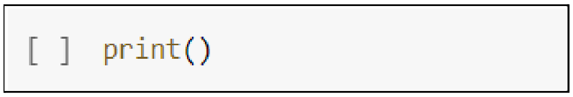
```
console.log("Is is a number:", num1)
```

Remember how we can print something in JavaScript?

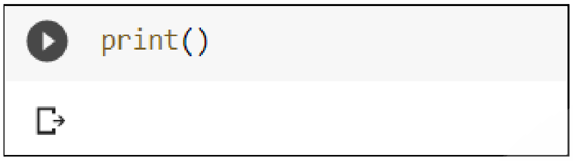
Great!

In Python, we use the **print()** method. If we do not pass anything in the method, there will simply be no output without any error.

ESR: We used **console.log()**.



```
[ ]  print()
```



```
print()
```

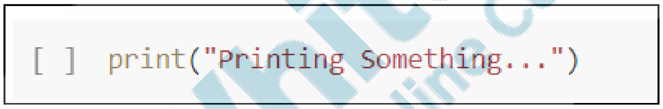
Now, let’s try to print strings and number variables using the **print()** method.

Do you remember how we print string variables and number variables in JavaScript?

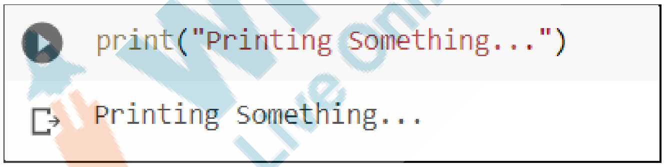
ESR: The string variables are used within quotation marks, **" "** and numbers variables are to be printed without the quotation marks, **" "**.

Great!

In Python too, we do the same, to print string and numbers.



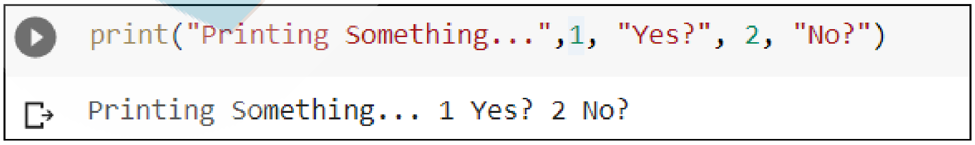
```
[ ]  print("Printing Something...")
```



```
print("Printing Something...")
```

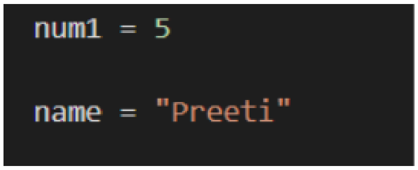
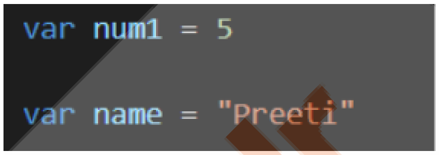
```
Printing Something...
```

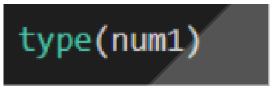
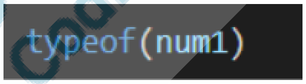
We can also combine multiple values, strings and numbers, together in the **print()** method separated by comma.



```
print("Printing Something...",1, "Yes?", 2, "No?")
```

```
Printing Something... 1 Yes? 2 No?
```

DEFINING VARIABLES	
	JS
 `num1 = 5` `name = "Preeti"`	 `var num1 = 5` `var name = "Preeti"`

CHECKING TYPE OF A VARIABLE	
	JS
 `type(num1)`	 `typeof(num1)`

DATA TYPES			
		JS	
• str	“Preeti”	• string	“Preeti”
• int • float	5 5.5	• number	5 5.5

Variables: Can you tell me, how do we use variables in JavaScript? Amazing!	**ESR:** We use the **var** keyword and then the **name** of the variable.

With Python, it is pretty simple, we just have to use the **name** of the variable.

Do you remember the types of the variables we have learned about in JavaScript?

Perfect!
Today, we will understand the “string” and “number” data type variables in Python.

<u>In JavaScript, we have a single data type for all numeric values, but in Python we have different data types for different numeric values</u>.

A whole/integer(both positive and negative) number in Python is of type **int** and a decimal number is of type **float**.

We can take two variables, **num1** and **num2**.

Let’s see what's the type of these two numbers.

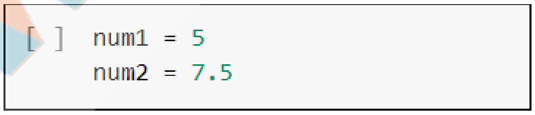
Remember how we find out the type of any variable in JavaScript?

Correct! In Python, we use the **type()** method.

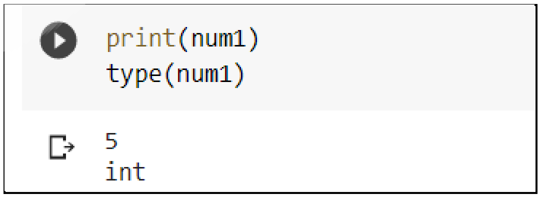
ESR: We learned about:

- string
- number
- array
- JSON Object

ESR: We use the **typeof()** method.

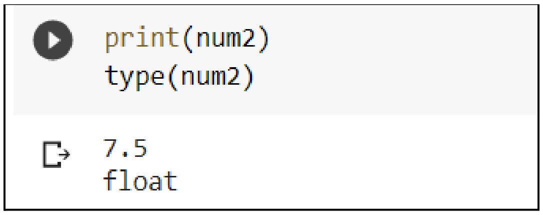


```
[ ]  num1 = 5
     num2 = 7.5
```



```
print(num1)
type(num1)
```

```
5
int
```



```
print(num2)
type(num2)
```

```
7.5
float
```

MATHEMATICAL OPERATIONS			
		JS	
• Addition (+)	5 + 5	• Addition (+)	5 + 5
• Subtraction(-)	5 - 5	• Subtraction(-)	5 - 5
• Multiplication(*)	5 * 5	• Multiplication(*)	5 * 5
• Division(/)	5 / 5	• Division(/)	5 / 5
• Modulus(%)	5 % 5	• Modulus(%)	5 % 5

Next, we move on to Mathematical operators.

Do you remember the Mathematical operations in JavaScript?

In Python, it's exactly the same.

Mathematical Operations:

- Addition
    - Symbol : +

- Subtraction
    - Symbol : -

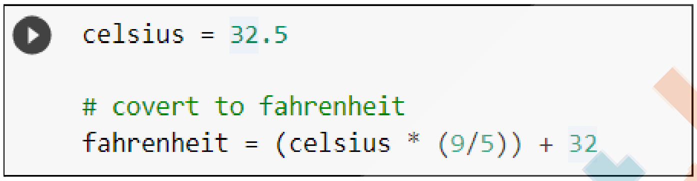
ESR: Yes. Addition, subtraction, division and multiplication.

● Multiplication ○ Symbol : * ● Division ○ Symbol : / ● Modulus ○ Symbol : % Let's quickly try a few operations in Python. We will try to convert the temperature value in Celsius to Fahrenheit. Before that, can you quickly tell me what's a normal human body temperature? Correct! Let's try to make a simple temperature converter using Python. Do you know how to convert degree Fahrenheit to degree Celsius? We can use the formula:	**ESR**: It is 37° Celsius or 98.6° Fahrenheit **ESR**: Varied.
$\boxed{\text{Fahrenheit} = \frac{9}{5} \times (\text{Celsius}) + 32}$	
Can you tell what all operations we need to do here? Yes. Correct! We need to: ● Divide 9 by 5. ● Multiply it by the value of the temperature in Celsius.	**ESR:** We need to: ● Divide ● Multiply ● Add

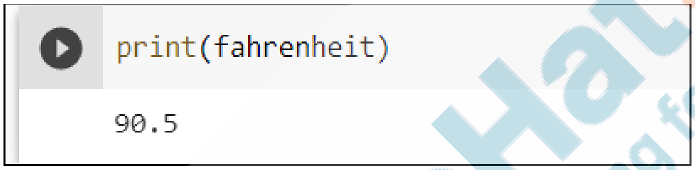
- Add 32 to the result.

Let’s take a variable **celsius** with some value.

We can take another variable, **fahrenheit** to store the result after converting the **celsius** value.

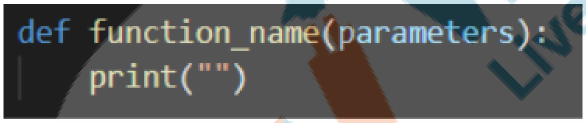


```
celsius = 32.5

# covert to fahrenheit
fahrenheit = (celsius * (9/5)) + 32
```

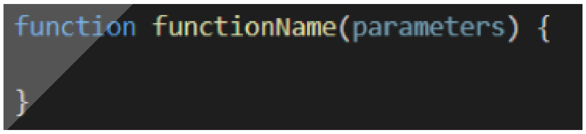


```
print(fahrenheit)

90.5
```

FUNCTIONS



```
def function_name(parameters):
    print("")
```



```
function functionName(parameters) {

}
```

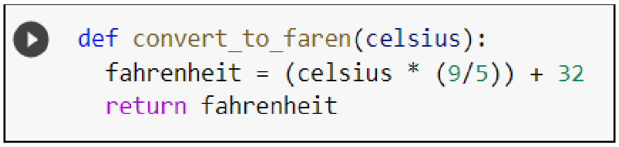
That was pretty simple again, right?

ESR: Yes.

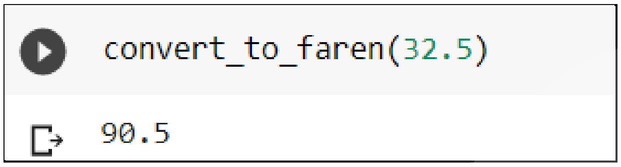
Remember how we used to write **functions** in JavaScript?

ESR: We use **function** keyword and then **name** of the function with parenthesis(curved brackets).

Superb! In Python, this syntax changes when we compare it with JavaScript. ● First, we use **def** keyword instead of function. ● Second, we use colon(**:**) after the name of the function ● Third, there are **no curly brackets**, { }, to represent the starting and ending of the code block. Can you think, how would we identify where the function starts and where it ends? For this we use indentation. Python uses **indentation** to define a block of code. Do you remember what indentation is? It's the margin or white spaces we leave from the left side at the beginning of the line of code. In Python, everything that has equal white spaces from the left side is a part of the same code block. We have to strictly follow this in Python, otherwise we will get errors! Let's quickly define a function to do the same temperature conversion with **Celsius value as input parameter** value to **return** the **Fahrenheit** value after conversion and call the function. ***Note**: We should define the functions before calling.* We can use the **return** keyword as we did in JavaScript to return the values. Calling a function is similar to JavaScript.	**ESR**: Varied **ESR**: Varied



```
def convert_to_faren(celsius):
  fahrenheit = (celsius * (9/5)) + 32
  return fahrenheit
```



```
convert_to_faren(32.5)
```

```
90.5
```

What if I want someone to enter the value they want to convert, then we will need to ask the person to enter the value.

This can be using the **input()** method in Python.

Let's take variable **cels** to take input.

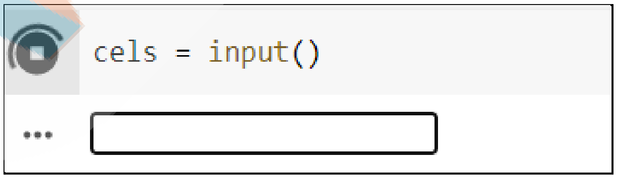
Once we run the cell, an input box will appear to enter the value.

Type the value in the box and press **Enter**.

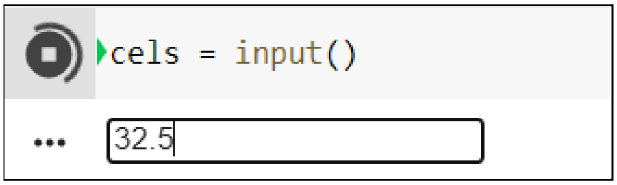
We can print the value of a variable using the **print()** method.

Also, when we use the **input()** method, everything is considered as **str** type in python.

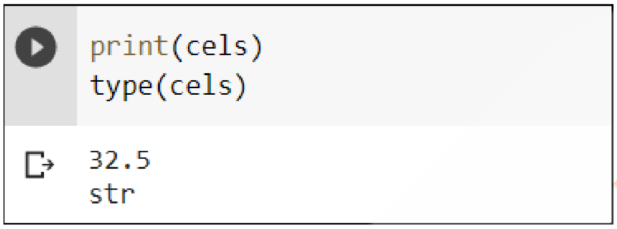
Since the variable value enters the user, we will need to convert it to a float number while calling the function.



```
cels = input()
```

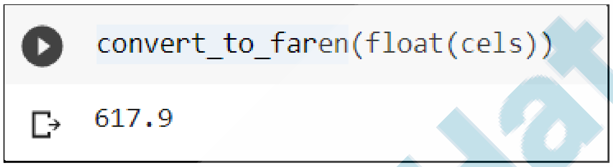


```
cels = input()
... 32.5
```



```
print(cels)
type(cels)
```

```
32.5
str
```

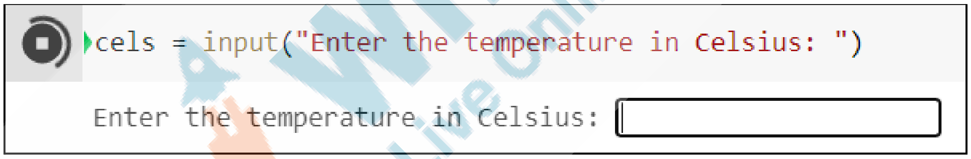


```
convert_to_faren(float(cels))
```

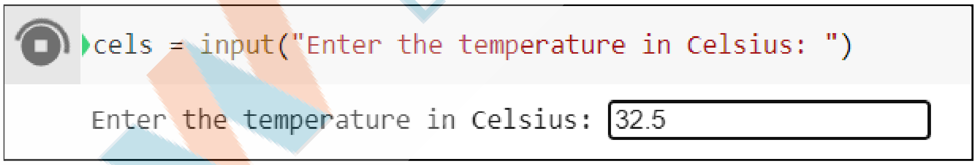
```
617.9
```

We can also give some instructions while taking input, so the user knows what to enter.

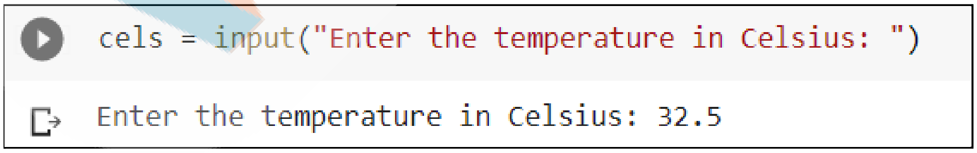
Note: Press the “Enter” key after entering the value in the input box to run the further program.



```
cels = input("Enter the temperature in Celsius: ")
Enter the temperature in Celsius:
```



```
cels = input("Enter the temperature in Celsius: ")
Enter the temperature in Celsius: 32.5
```



```
cels = input("Enter the temperature in Celsius: ")
```

```
Enter the temperature in Celsius: 32.5
```

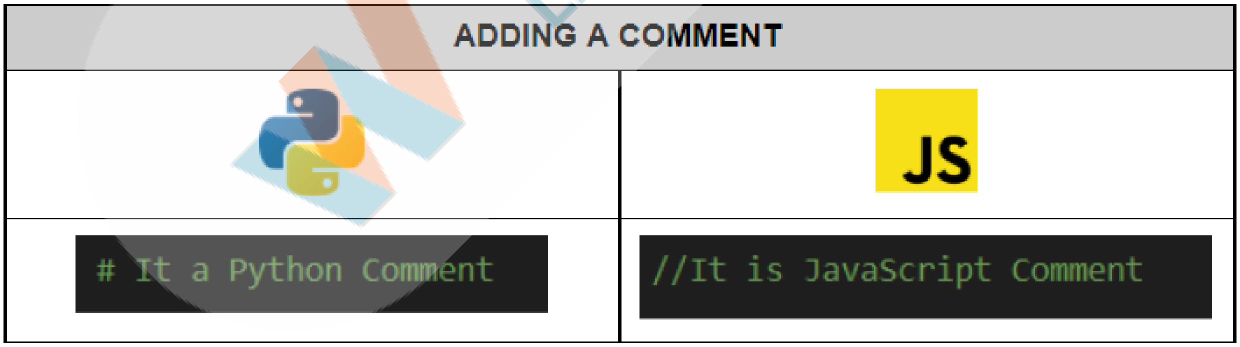
That was interesting! We now know a few things to write in Python programming as well. Now you write functions for addition and subtraction mathematical operations. Also, you will write one more function to take input from the user and perform the mathematical operations based on the option chosen by the user. Are you excited?	**ESR:** Yes.
Teacher Stops Screen Share	
So now it's your turn. Please share your screen with me.	
Teacher Starts Slideshow **Slide 23 to 24** Refer to speaker notes and follow the instructions on each slide.	
We have one more class challenge for you. Can you solve it? Let's try. I will guide you through it.	
Teacher Ends Slideshow	
STUDENT-LED ACTIVITY - 20 mins	
- **Ask the student to press the ESC key to come back to the panel.** - **Guide the student to start Screen Share.** - **The teacher gets into Fullscreen.**	

Student Initiates Screen Share

ACTIVITY

- **Conditionals**
- **Write functions to perform each mathematical operations**
- **Write a function to get the operation result based on the user input.**

Teacher Action	Student Action
Guide the student to start the Google Colab Notebook. ***[Student Activity 1]*.**	
Can you tell me how to add comments in JavaScript? In Python, we add comment using hash, **#,** symbol. Python vs JavaScript Syntax: ***[Student Activity 2]*.** Comments are not executable lines of codes. These are used to help the reader to understand the code much better, right? ***Note**: Help the student to understand the importance of comments and encourage them to use them in activities and projects.*	**ESR:** We use two forward slashes(//) **ESR**: Yes.

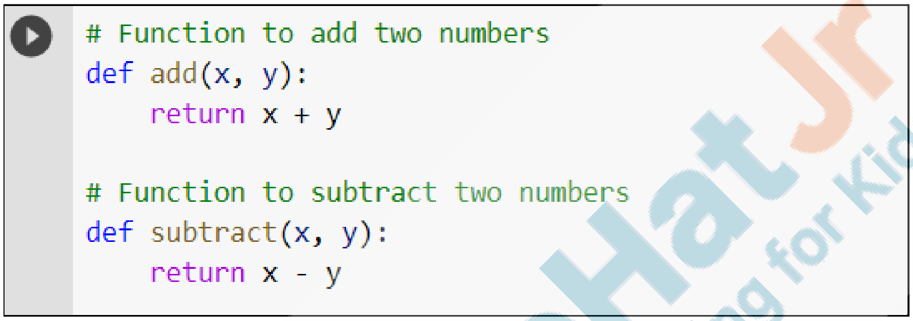


ADDING A COMMENT	
	JS
`# It a Python Comment`	`//It is JavaScript Comment`

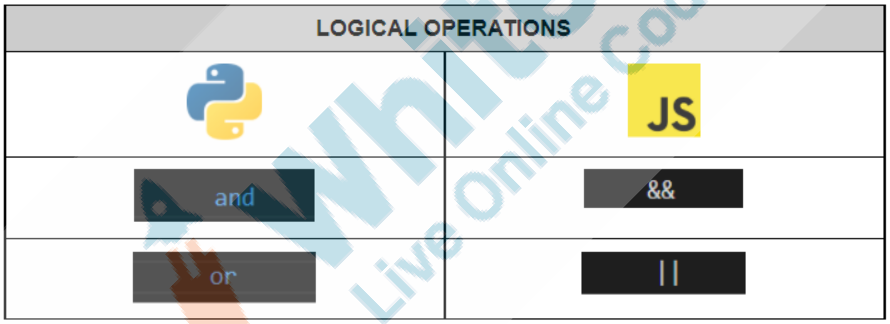
*Guide the student to write **2 simple functions** which the result of two numbers after:*

1. *Addition*
    - *Take a function with two parameter x and y*
    - *Return sum of x and y*
2. *Subtraction*
    - *Take a function with two parameter x and y*
    - *Return difference of x and y*

***Note**: We should define the functions before calling.*

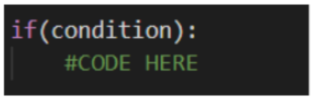
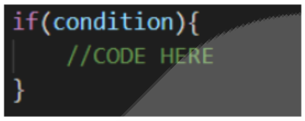


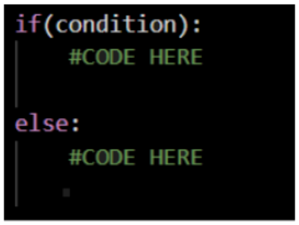
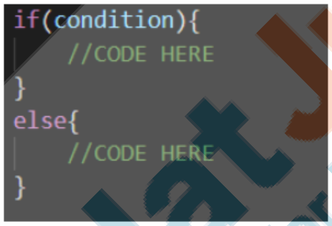
```
# Function to add two numbers
def add(x, y):
    return x + y

# Function to subtract two numbers
def subtract(x, y):
    return x - y
```

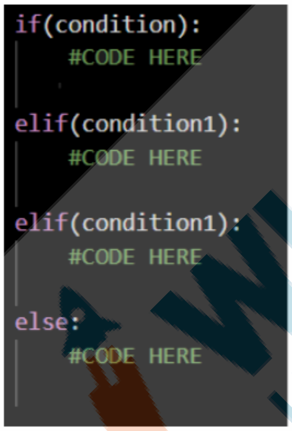
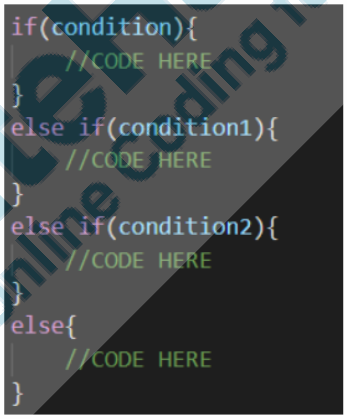


LOGICAL OPERATIONS	
	JS
and	&&
or	||

CONDITIONAL STATEMENT	
	JS


`if(condition):`
`    #CODE HERE`
	
	`if(condition){`
	`    //CODE HERE`
	`}`


`if(condition):`
`    #CODE HERE`
`else:`
`    #CODE HERE`
	
	`if(condition){`
	`    //CODE HERE`
	`}`
	`else{`
	`    //CODE HERE`
	`}`


`if(condition):`
`    #CODE HERE`
`elif(condition1):`
`    #CODE HERE`
`elif(condition1):`
`    #CODE HERE`
`else:`
`    #CODE HERE`
	
	`if(condition){`
	`    //CODE HERE`
	`}`
	`else if(condition1){`
	`    //CODE HERE`
	`}`
	`else if(condition2){`
	`    //CODE HERE`
	`}`
	`else{`
	`    //CODE HERE`
	`}`

Conditionals and Logical Operations:

Define the **calculate() function** based on **option** parameter:

- To check if the option is **1**(For Add) or **2**(For Subtract)

  Do you remember how we use conditionals in JavaScript?

ESR: Yes, we use **if**, **else** & **if-else** keywords.

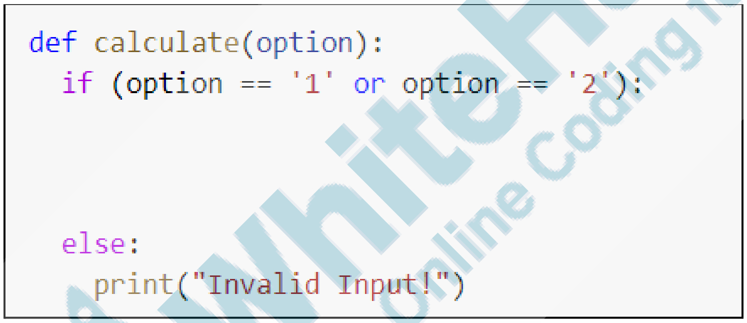
Yes. Great!!

In Python, we also check conditions using **if** and **else**.

But if we have more than two conditions with different outputs, we use **elif**.

Also, we can combine conditions together using **and**, **or** keywords in Python.
We do not use **&&** and **||** like JavaScript.

- Else print “**Invalid Input**”



```
def calculate(option):
  if (option == '1' or option == '2'):



  else:
    print("Invalid Input!")
```

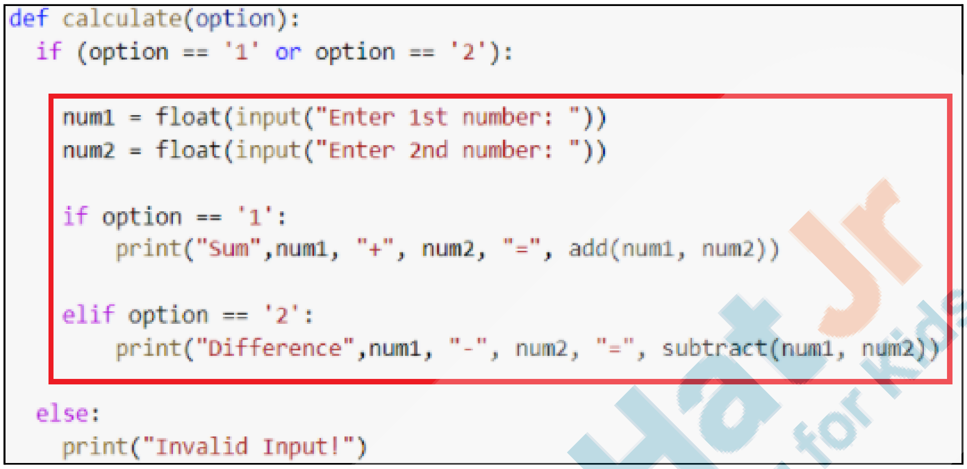
- Take 2 numbers variables, **num1** and **num2** as input from the user inside the first **if** condition. Taking input will have the default type as str for strings. Since we want to do the calculation we should convert the numbers to **float** type.

- Check if the **option** is **1**, then call the add function for the two input numbers, **add(num1, num2)**

- Check if the **option** is **2**, then call the subtract function for the two input numbers, **subtract(num1, num2)**

Indentation Matters a lot in Python!

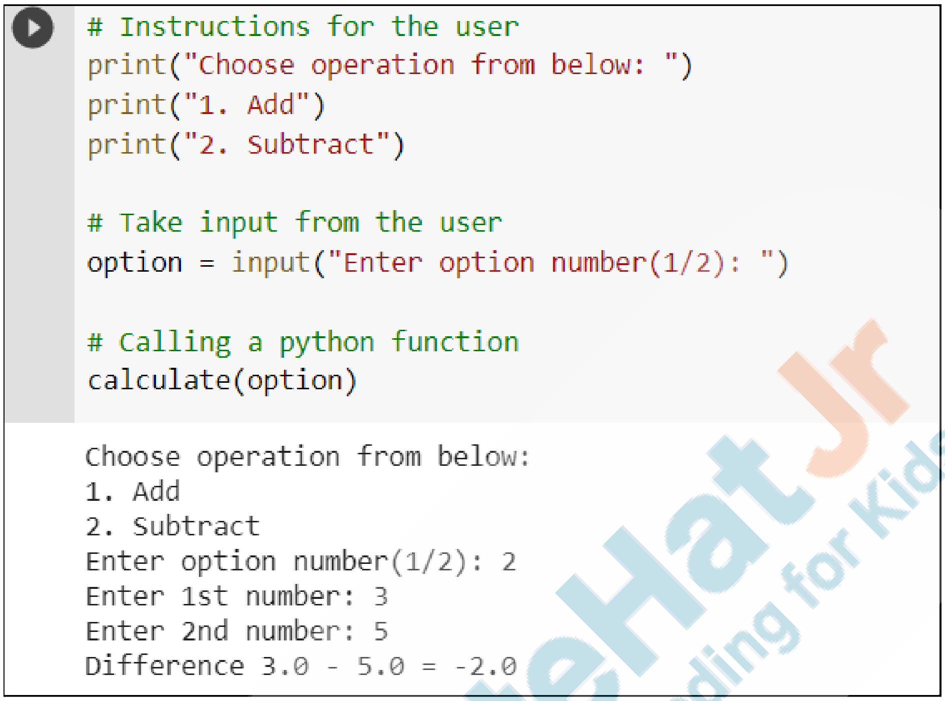
***Note**: Make sure the indentation is proper.*



```
def calculate(option):
  if (option == '1' or option == '2'):

    num1 = float(input("Enter 1st number: "))
    num2 = float(input("Enter 2nd number: "))

    if option == '1':
        print("Sum",num1, "+", num2, "=", add(num1, num2))

    elif option == '2':
        print("Difference",num1, "-", num2, "=", subtract(num1, num2))

  else:
    print("Invalid Input!")
```

Now let's give options to the user to choose from and call the function to test the result:

- Use the **print()** method to show the available options for operations.
- Take the **option** variable as input.
- Call the **calculate()** function with **option** value.



```
# Instructions for the user
print("Choose operation from below: ")
print("1. Add")
print("2. Subtract")

# Take input from the user
option = input("Enter option number(1/2): ")

# Calling a python function
calculate(option)
```

```
Choose operation from below:
1. Add
2. Subtract
Enter option number(1/2): 2
Enter 1st number: 3
Enter 2nd number: 5
Difference 3.0 - 5.0 = -2.0
```

You did really amazing work today!

We learned the basic syntax of Python and wrote functions to do addition and subtraction.

***Note**: To read more on Python Syntax, refer to the [Python Documentation](). *

Teacher Guides Student to Stop Screen Share

WRAP UP SESSION - 5 mins

Teacher Starts Slideshow
Slide 25 to 30

Activity details
Following are the WRAP-UP session deliverables:

- Appreciate the student.

- Revise the current class activities.
- Discuss the quizzes.

WRAP-UP QUIZ
Click on In-Class Quiz

Continue WRAP-UP Session
Slide 31 to 36

Activity Details

Following are the session deliverables:

- Explain the facts and trivia
- Next class challenge
- Project for the day
- Additional Activity (Optional)

FEEDBACK

- **Appreciate and compliment the student for trying to learn a difficult concept.**
- **Get to know how they are feeling after the session.**
- **Review and check their understanding.**

Teacher Action	Student Action
You get Hats off for your excellent work! In the next class, you will learn about the sequence data type in Python. You will learn about lists in Python. You will also understand for loops in Python and how to access lists using for loops.	*Make sure you have given at least 2 Hats Off during the class for:* Creatively Solved Activities +10 Great Question +10

PROJECT OVERVIEW DISCUSSION

Refer the document below in Activity Links Sections

Teacher Clicks

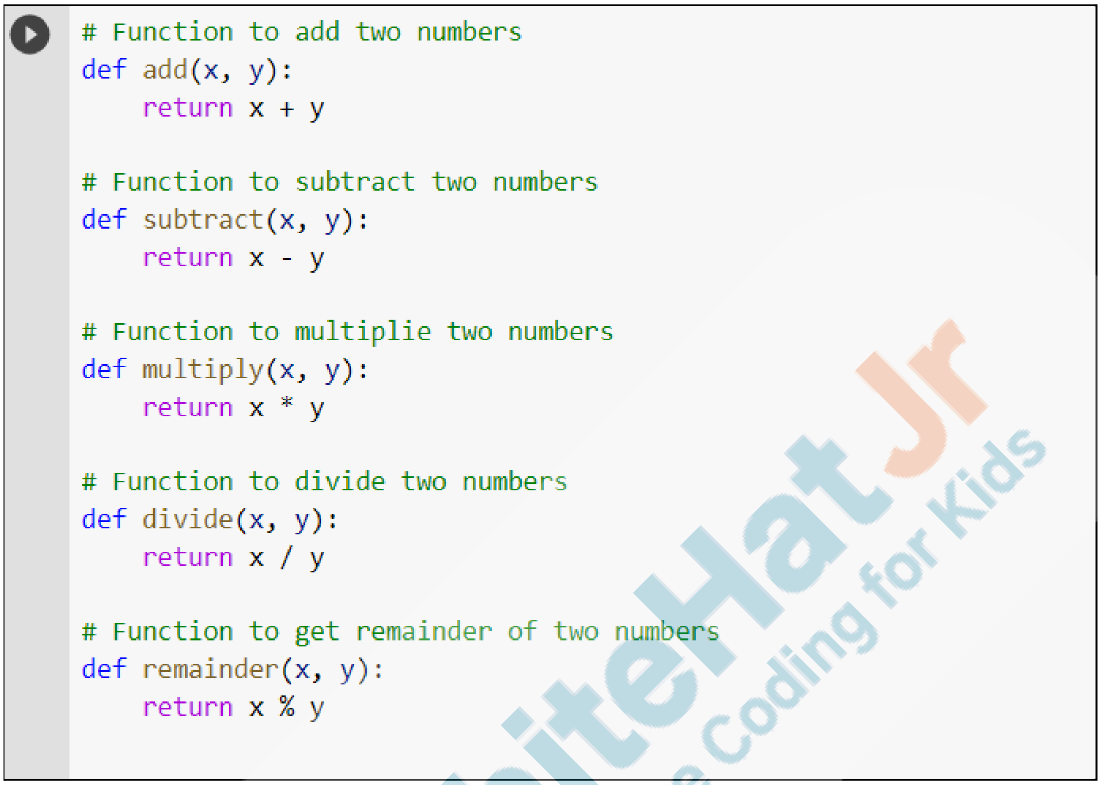
ADDITIONAL ACTIVITIES

(Optional)

Additional Activities

Encourage the student to extend the student activity to write 3 more functions to do the:

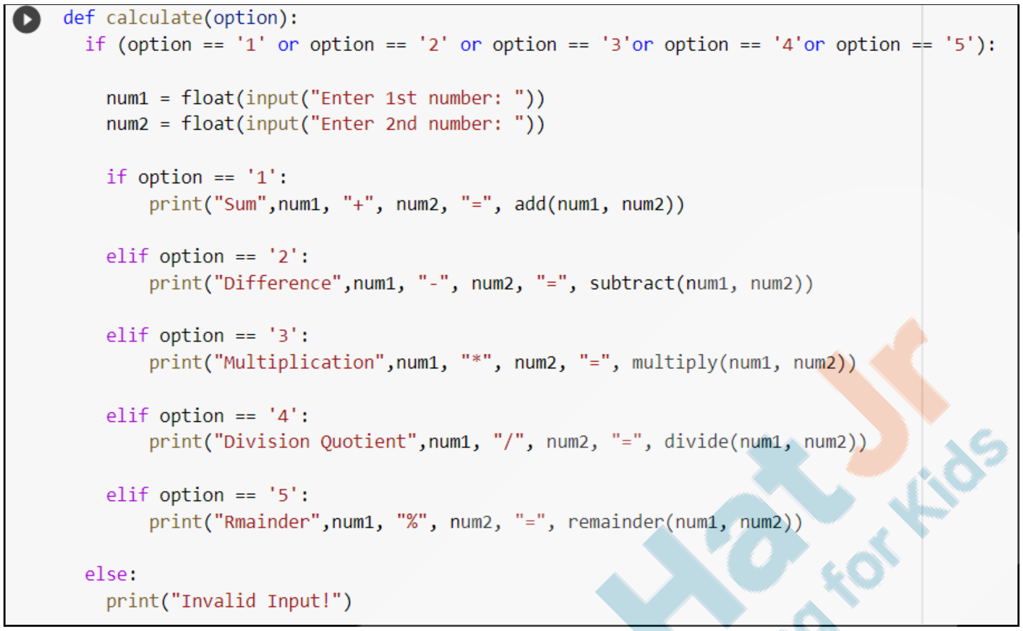
1. Multiplication
    - Take a function with two parameter x and y
    - Return multiplication of x and y
2. Division
    - Take a function with two parameter x and y
    - Return quotient of x and y
3. Remainder
    - Take a function with two parameter x and y
    - Return remainder of x and y



```python
# Function to add two numbers
def add(x, y):
    return x + y

# Function to subtract two numbers
def subtract(x, y):
    return x - y

# Function to multiplie two numbers
def multiply(x, y):
    return x * y

# Function to divide two numbers
def divide(x, y):
    return x / y

# Function to get remainder of two numbers
def remainder(x, y):
    return x % y
```

4. Add condition to check option and call the respective function:

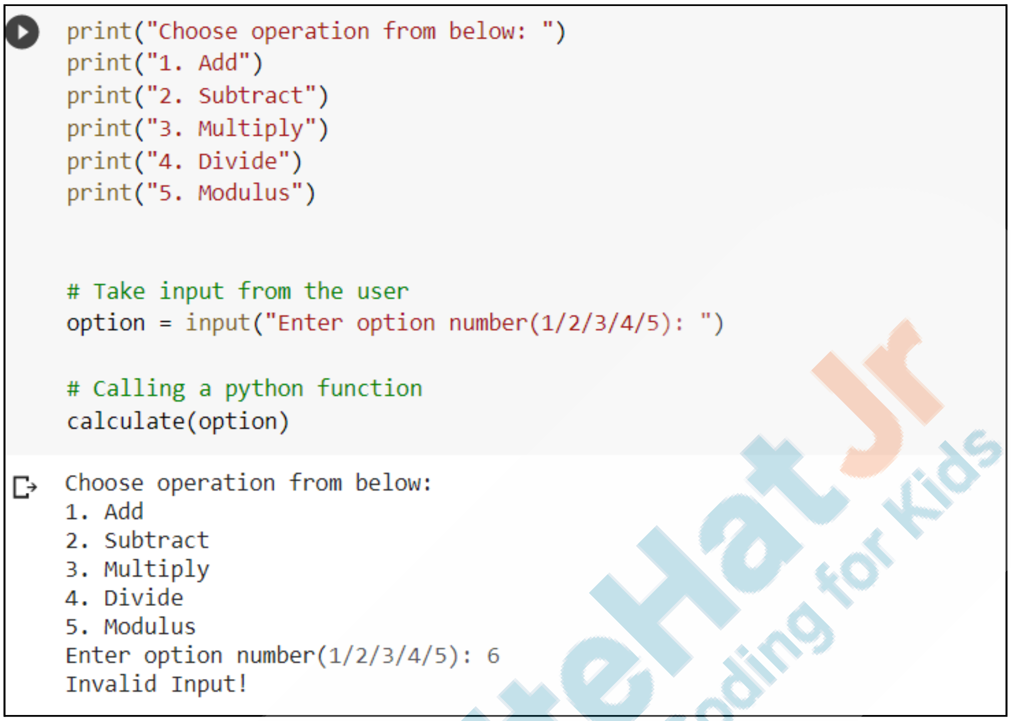
   Option 3 = multiply
   Option 4 = divide
   Option 5 = remainder



```
def calculate(option):
  if (option == '1' or option == '2' or option == '3'or option == '4'or option == '5'):

    num1 = float(input("Enter 1st number: "))
    num2 = float(input("Enter 2nd number: "))

    if option == '1':
        print("Sum",num1, "+", num2, "=", add(num1, num2))

    elif option == '2':
        print("Difference",num1, "-", num2, "=", subtract(num1, num2))

    elif option == '3':
        print("Multiplication",num1, "*", num2, "=", multiply(num1, num2))

    elif option == '4':
        print("Division Quotient",num1, "/", num2, "=", divide(num1, num2))

    elif option == '5':
        print("Rmainder",num1, "%", num2, "=", remainder(num1, num2))

  else:
    print("Invalid Input!")
```

5. Add option in the menu:

    Option 3 = multiply
    Option 4 = divide
    Option 5 = remainder



```
print("Choose operation from below: ")
print("1. Add")
print("2. Subtract")
print("3. Multiply")
print("4. Divide")
print("5. Modulus")


# Take input from the user
option = input("Enter option number(1/2/3/4/5): ")

# Calling a python function
calculate(option)
```

```
Choose operation from below:
1. Add
2. Subtract
3. Multiply
4. Divide
5. Modulus
Enter option number(1/2/3/4/5): 6
Invalid Input!
```

ACTIVITY LINKS		
Activity Name	**Description**	**Link**
Teacher Activity 1	Google Colab Notebook	Google Colaboratory
Teacher Activity 2	Python vs JavaScript Syntax	https://s3-whjr-curriculum-uploads.whjr.online/8036f431-e83d-46ab-a98e-1f6326be1b7d.pdf
Teacher Activity 3	Reference Code	https://colab.research.google.com/drive/1NxBe3DmI7wvflI-MA05ACsalSJrIELPN?usp=sharing
Student Activity 1	Google Colab Notebook	Google Colaboratory
Student Activity 2	Python vs JavaScript Syntax	https://s3-whjr-v2-prod-bucket.whjr.online/8efe7302-4787-462f-bfd1-94205d387034.pdf

Teacher Reference 1	Python Document Reference	The Python Tutorial — Python 3.9.5 documentation
Teacher Reference 2	Project Document	https://s3-whjr-curriculum-uploads.whjr.online/a6115a61-84f8-4b8e-9ac9-d380841515fa.pdf
Teacher Reference 3	Project Solution	https://colab.research.google.com/drive/16qbP7cR6rlVe-LRvluiR6hvla6c4QEnA?usp=sharing
Teacher Reference 4	Visual-Aid	https://s3-whjr-curriculum-uploads.whjr.online/2fb38059-239f-4c5f-b4e1-95f416d59267.html
Teacher Reference 5	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/92d11b34-f995-4def-9645-54f5cf87f635.html
Student Reference 1	Python Document Reference	The Python Tutorial — Python 3.9.5 documentation