# Python Introduction
# Let's Dive In!

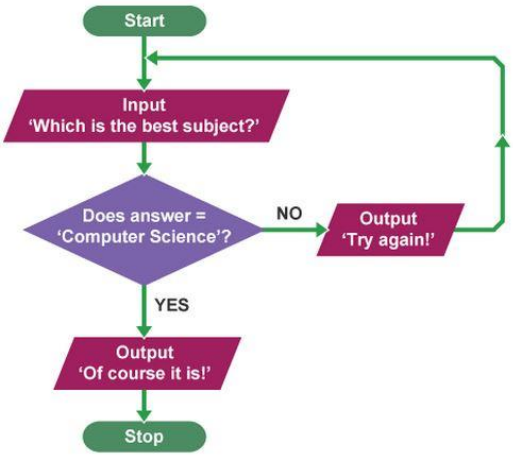
Remember computer programming is about **solving problems** and **creating useful applications**. Python is an amazing language that will quickly allow you to build amazing, interactive software and solve all sorts of interesting problems.

But first we must learn some basics. We need to get familiar with some key principles before we can use Python effectively. The next few lessons are designed to get you familiar with the basics of Python.

All computer programming languages, really, have a few key aspects:

1. **Input**
2. **Output**
3. **Decisions**
4. **Calculations**
5. **Repetition**

Python will give you tools in **all** of these areas. Let's take a look!




## Input, Output and Variables:

## Output:

Generally, Python can **output** (produce): lists, numbers, words, and images to the screen. The most interesting and easiest are outputs are **words** and **images** so let's start by practicing creating these *simple* outputs.

# Exercise#1

**Type the following into Pickcode (don't cut and paste! – Type it in please)**

```
print("I am a cool student.")
print("I am learning Python.")
print("Python is fun!")
```

**Type the following into Pickcode (don't cut and paste! – Type it in please)**

```
print("Im the best student in the world! ")
print("\n")
print("I am learning Python.")
```

make sure you know what **\n** does.

**Type the following into Pickcode**

```
print("I'm the \"best\" coding student in Whistler! ")
print("I am learning Python. ")
```

Please **save** your exercises **in Pickcode** so you can submit all exercises as a single assignment later.

What does \" do?
\" and \n are called:

# Escape Characters \

**Escape Characters** are special characters that **help us organize written output**. There are many of these. Here is a short list of some you should know.

| **Escape character:** | **What it does:** |
|---|---|
| \r | Return (goes back to start of **same** line. Writes over text. |
| \" | Double-quote |
| \t | Tab |
| \n | Newline (line break) |

You can use the following link to see more cool **escape characters** that can help you **organise your written output**:
https://www.w3schools.com/python/gloss_python_escape_characters.asp

# Spicing up your output using emojis

Python allows you to get creative with written output by adding emojis! This is a nice way to make a program more personal and **fun to for the user**.

Using emojis in Python is easy. You simply need to **add in a bit of code** or **cut and paste the emojis into a print statement**.

Try typing the following code into Pickcode to see how emojis work. **Add them to the exercise#1** you just completed above.

```
print("Are you happy today?", "\U0001f600")
print("No I am sick!!! ", "\U0001F922")
```

```
print("Hi there! Pick your favorite animal below:", "\U0001f600")
print("1.", "\U0001F40E")
print("2.", "\U0001F411")
print("3.", "\U0001F412")
print("4.", "\U0001F436")
```

The codes for the emojis used above can be found at:

https://apps.timwhitlock.info/emoji/tables/unicode#block-1-emoticons
https://unicode.org/emoji/charts/full-emoji-list.html#1f922

**Note:** that you must modify the codes on the pages above from U+1F411 to \U0001F411 (You need to **add the backslash** before the **U** and **three 0"s** after the **U** instead of the **+** sign)

You can also **attempt to just cut and paste emojis into a print statement** like the example below. Try it out! You can **highlight**, **then cut and paste** the emoji from the one of the links above. Try the examples below:

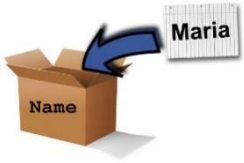
```
print("My name is Robotron!", "\U0001F916")
print("I came here in a spaceship", "🛸")
```

*Too Easy*

Now you can get words and emojis to appear on a screen using Python. But that's not *too* exciting... you can do that by just typing into a word doc! Let's keep going:

# Input and Variables:

One way Python can gather information is by getting it from **humans**! **Humans** can type words, numbers, lists, and commands into python. We can then store these items in **variables.**




A **variable** (in **all** computer programming languages) is**: a way to label and store information...** like putting something into a labelled box. Let's have a look to see how this works:

## Exercise#2

**Type** the following into Pickcode:

```
age=17
name="Jeff"
fav_food="apples"

print(age)
print(name)
print(fav_food)
print("\n")
print("Hi my name is",(name))
print("I'm",(age))
print("My favorite food is",(fav_food))
```

See how we are **creating storage spaces** and then **stuffing information into them**? In programming, we do this with the **=** (equals sign)...which in programming does **NOT** mean "equal to"...it means ***"assign value of"***

Type the following into Pickcode below exercise#2:

```
fav_food="banana"   #change my favorite food :)

print(fav_food)

print("\n")         # leave a space
fav_food="toast"    # changed my favorite food again!
print("my name is")
print(name)
print("my favorite food is")
print(fav_food)
```

When you see a **hashtag #** in python, this means you have created a "comment" in your code. Comments are ignored by the computer, but are handy for humans who like to put comments and reminders in their code

Variables are a way of **creating storage spaces** and then **stuffing things into them**.

= means ***"assign value of"***

**Type** the following in to Pickcode! Carefully look at the output after you run the program to **see if it makes sense**.

```
team_players=23
extras=5
total= team_players + extras
print("The number of regular players on our team is")
print(team_players)
print("We have this may extra players")
print(extras)
print ("Total amount of players is")
print(total)
print("\r")
extras=10
total= team_players + extras
print ("Now we have this many extras!")
print(extras)
print("So our total number of players is")
print(total)
```

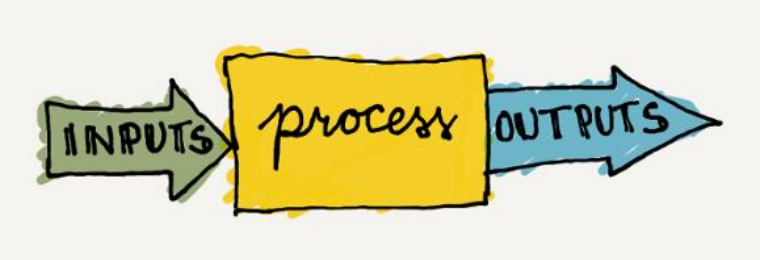
## You can now use Python to:

1. print stuff to the screen and
2. use **variables**!

## Good for you!

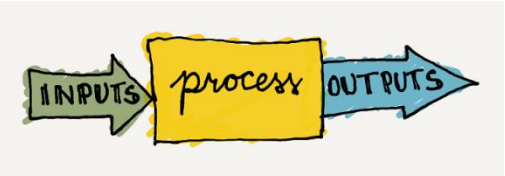
Later we will look at how to **draw images** with python,

but for now let"s look at **INPUT (collection information to use in a program)**:




Continued on next page....

# Input (Collecting information)




**Collecting information** is a key part of computer programming.
One way Python does this is with the **`input()`**command.
The **`input()`** command allows python to **accept and store typed information from a human.**

## Exercise#3

**Enter the following code into Pickcode**. When you run the program, the **`input()`** command will prompt you to enter information. Try it out!

```
print ("Hi, My name is Calcutron, I'm from planet Zerp.")
name=input("What is your name?")
print ("Hi",name)
planet=input("What is the name of your planet?")
print(planet,"?????")
print("Doesn't sound too cool. I'm renaming your planet Salad Ball!")
print("You live on Salad Ball!")
print("Do you like the new name of your planet",name,"?")
answer=input("yes or no?")
if answer=="yes":
  print("Good!",name,"Who is your king on planet Salad Ball?")
if answer=="no":
  print("that's to bad",name,"I like the new name.")
  print("Salad Ball!"*3)
```

Notice the **"if" statements** above...we will talk about later but it's not to hard to see what an **"if" statement** does from the example. **INDENTING!** – notice the **indenting** – very, very, very important!

## **Exercise#4** (please save your work)


- Create a new program like the one in exercise#3 (you can use the code above as a template). Your goal will be to **create a character that someone can have a conversation with**.
- Use **If** statements (don't forget about **indenting**).
- Make it as fun and as interesting as possible for the user!
- **Use emojis**!
- Show Mr. Walzl when you are done. You will be marked on this!
- Need a creative boost? Maybe use the scenario of a person going to buy something from the corner store.

# Fun with Formatting:

Making stuff look good and easy to read on the screen is an important part of programming. Creating an environment where humans can easily interact with your software is crucial. We also want to be able to engage users and **clearly** communicate solutions to the problems our programs solve.

Below are a few ways to **help make your** OUTPUT **more readable and organized** from a user's point of view:

## *Pausing* Your Program:


When creating cool stuff using python, you will sometimes want your program to **pause**. This can be useful when you want the program to wait a bit between outputs.

**Exercise#5** (type the following into Pickcode. Then answer the questions **on the next page** using a comments section below you're the original code.

```
import time
print("hello!")
time.sleep(1)
print("What's up?")
time.sleep(1)
answer=input("what's your favorite color? ")
time.sleep(1)
print(f"your favorite color is {answer}?")
print(f"{answer} is a dope color!")

#hashtag-(#) indicates comments that won't be run as part of your program
#anything after the hashtag should only notes for the programmer or his/her pals.
#Read the questions on the next page
#Answer the questions using #hashtage comments below this code before you submit the
code
```


What is happening in this example? Lots of stuff:

1. We **imported** the **time** "*library*". This is necessary to make the **sleep** function work.
2. We used the **`time.sleep()`** function to pause the program. The number (1) in the brackets is used to indicate the number of seconds the program will pause for.
3. We used the **`f {}`** function to insert a variable into a print statement….cool!
4. We used **`# (hashtags)`** to place comments in our code.

## Exercise#5 QUESTIONS:

Write the answers to the following questions **as comments at the end of your exercise#5** in your python code using the **# hashtag symbol**
Make sure you **submit your code**.

1. What is a **library** in python (look it up)? Why do we have to import the time library for this code?
2. Change the `time.sleep(1)` in the code to `time.sleep(3)` and describe what changes this makes to how the program runs.
3. Describe what the `f {}` command does.
4. What **color** are the comments in Pickcode when you use the # symbol?

# Clearing the screen:

This function is particularly useful when you need to display *several pieces of information on the screen at different times*:

## Exercise#6

Type the following into Pickcode (don"t cut and paste! – Type it in please and SAVE)

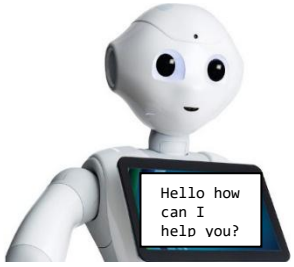
```
import os
import time
print("Hello there!")
time.sleep(2)
os.system("clear")
name=input("What is your name? ")
time.sleep(1)
os.system("clear")
print(f"Hi {name}! Nice to see you today!")
time.sleep(4)
os.system("clear")
print("Pleasant weather we're having isn't it?")
time.sleep(3)
os.system("clear")
print("Good Bye!")
time.sleep(3)
os.system("clear")
```

*What's happening in the example is above?*

- We have to `import os` (operating system library) and `time` library
- We used the `os.system("clear")` command to clear the screen
- We also used `time.sleep()` to pause the program at appropriate spot.

# Stand out!

Another cool formatting feature.
Let's really **jazz up your output** try the following:




## Exercise#7

Type the following into Pickcode to see what it does. Save your work. Watch your *indenting*!

```
import time
import sys
sentence=("Check this out!!! What a cool way to format output in Python!")

for letter in sentence:
        sys.stdout.write(letter)
        sys.stdout.flush()
        time.sleep(0.1)
```

*What happening in the example is above?*

- We have to `import sys` (system library) and the `time` library
- We use a "`for` loop" to repeat some instructions…more about this later.
- We used the `sys.stdout.write(letter) and flush()` so we can keep write each character out on the same line.
- We also used `time.sleep(0.1)` to pause the program for 0.1 seconds after each character is written to the screen at appropriate spots

## Exercise#8

Create a cool dialog between a robot and a human **like what you did in exercise#4,** but this time make sure you include the following:

- Use the **clear screen** and **sleep functions** in your program (appropriately).
- Use `if` statements
- Use variables
- Use the `f{}` command to insert variables into text.
- Use the `stdout` function and code above to create a "typing text" effect.
- Make sure the dialog is as **engaging** and **fun** as possible.
- Use Emojis.