Our first code

In [1]:
```
print ("hello world!")
```

```
hello world!
```

THIS IS OUR FIST PYTHON CODE

WE WILL USE THIS METHODE TO RUN MORE OUTPUTS

In [2]:
```
print('hello cynthia, how are you?')
```

```
hello cynthia, how are you?
```

If you notice, our code is on a straight line, lets add another line or paragraph

In [3]:
```
print("i feel alright, \ni feel good") #A new line has been added
```

```
i feel alright,
i feel good
```

VARIABLES(variables are used to store data in a compt. memory

In [4]:
```
a = 20
print(a) #we have given 'a' a value
```

```
20
```

In [5]:
```
milk = 200.99
meat = 405
b = milk + meat
print (b)  #now we have allocated values to meat and milk and have tied it to 'b' and ha
```

```
605.99
```

INPUT (we use input function to collect info from a user)

In [6]:
```
name = input("what is your name")
print("Hello " + name) #we give a single space after the first apostroph "hello "
```

```
what is your namemike
Hello mike
```

In [7]:
```
#numbers, string and boolens (10(int), "egg"(string), True(boolen))
#we will use this data types to colect and perform a type conversion
```

In [8]:
```
birth_year = input("input your birth year: ")
age = 2023 - birth_year"
print (age)
```

```
  File "C:\Users\Christopher\AppData\Local\Temp\ipykernel_1484\1904452474.py", line 2
    age = 2023 - birth_year"
                            ^
SyntaxError: EOL while scanning string literal
```

In [ ]:
```
#our program returned an error
#python does not understand how to deduct "int"(2023) from a string "birth_year"
```

In [9]:
```
birth_year = input("input your birth year: ")
int(birth_year)
age = 2023 -int(birth_year)
print (age)
#we used the int_function to convert the string to an int so python undrstands it
```

```
input your birth year: 1994
29
```

Int and float

In [10]:
```
#we use the type function to display the built in function (10, 10.1 ,str and bool)
a = 100
type(a)
```

Out[10]:
```
int
```

In [11]:
```
b= 2.5
type(b)
```

Out[11]:
```
float
```

In [12]:
```
street_name = "marcually hill"
type(street_name)
```

Out[12]:
```
str
```

In [ ]:
```
#we can convert a float to an int or a string boolen in python
```

In [13]:
```
summ = 0
first = input("what is your first input: ")
second = input("what is your second input: ")
sum = float(first) + float(second)
print(sum)
```

```
what is your first input: 10.1
what is your second input: 20
30.1
```

In [ ]:
```
#we should also know that the float function should follow since one of the inputs conta
```

Strings and Objects

In [14]:
```
course = 'Python For Beginners'
course.upper()
#converting to uppercase
```

Out[14]:
```
'PYTHON FOR BEGINNERS'
```

In [15]:
```
print ("course")
```

```
course
```

In [16]:
```
course.lower()
```

Out[16]:
```
'python for beginners'
```

In [17]:
```
print(course.find('y'))
```

```
1
```

In [ ]:
```
#tracing or counting in pythons starts with 0
#so 'pyhton' is counted as
#python
#012345
```

In [18]:
```
#replace function
course = 'python for beginners'
print(course.replace('for', '4'))
#here, we replaced 'for' with '4'
```

```
python 4 beginners
```

Arithmetic Operations

In [19]:
```
#we will be deaLing with operations like add, multiply , divisions and minus
```

In [20]:
```
print(10+4)
```

```
14
```

In [21]:
```
print(3*5)
```

```
15
```

In [22]:
```
print(146 - 30)
```

```
116
```

In [23]:
```
print(10//3)
```

```
3
```

In [24]:
```
print (10/3)
```

```
3.3333333333333335
```

In [25]:
```
print(10**2)
#power operation
```

```
100
```

In [26]:
```
#we can also use a like variable alternative
x = 10
x = x + 3
print(x)
```

```
13
```

In [30]:
```
#we can also use Augmented Assignmet Operator
```

```
x = 10
x += 3
print(x)
```

```
13
```

In [31]:
```
x = 10 + 4 * 5
print(x)
```

```
30
```

In [33]:
```
#lets see how python handles this...
#our output shows that py solved the bracket first(10+4) and then multiplied with 5
x = (10 + 4) * 5
print(x)
```

```
70
```

Boolens or comparison operator

In [37]:
```
x = 3 > 2
print(x)
type(x)
```

```
True
bool
```

Out[37]:

In [39]:
```
x = 3 ==2
print(x)
#we got a false because the agurement or comparison isnt true
```

```
False
```

In [40]:
```
x = 3 != 2
print(x)
```

```
True
```

logical operators or operation

In [43]:
```
meat = 24
print (meat > 10 and meat< 12)
```

```
False
```

In [44]:
```
#we can also use the 'or', 'and' and 'not'
meat = 24
print (meat > 10 and meat< 12)
```

```
False
```

In [45]:
```
meat = 24
print (meat > 10 or meat< 12)
```

```
True
```

In [49]:
```
meat = 24
print (not meat > 10 )
```

```
False
```

In [ ]:
```
meat = 24
print (meat > 10 and meat< 12)
```

If Statement

In [51]:
```
price = 45
```

In [53]:
```
if  price > 23:
    print("price is cheap")
#satemens are acheived if the conditions are met
```

```
price is cheap
```

In [56]:
```
if  price < 23:
    print("price is high")
elif price > 40:
    print("this is too costly")
    #if our conditions are met, we will get true
```

```
this is too costly
```

In [63]:
```
if  price < 23:
    print("price is cheap")
elif price < 40:
    print("this is costly")
elif price > 45:
    print ("cant afford it")
else:
    print ("No way")
```

```
No way
```

In [75]:
```
weight = input('what is your weight: ')
weight_measurement = input('kg or lb?: ')
weight_in_kg = (weight) + (weight_measurement)
print("Your weight is", weight_in_kg)
#Do this exercise in different ways and try to be creative
```

```
what is your weight: 50
kg or lb?: kg
Your weight is 50kg
```

In [80]:
```
weight = int(input("weight: "))
unit = input("(k)g or (l)bs: ")
if unit.upper() == "k":
    converted = weight//0.45
    print("weight in lbs: " + str(converted))
else:
    converted = weight * 0.45
    print("weight in kg: " + str(converted))
```

```
weight: 50
(k)g or (l)bs: k
weight in kg: 22.5
```

While Loops

In [88]:
```
i = 1
while i <= 5:
    print(i)
    i = i + 1
```

```
1
2
```

```
3
4
5
```

In [89]:
```
i = 1
while i <= 12:
    print(i)
    i = i + 1
```

```
1
2
3
4
5
6
7
8
9
10
11
12
```

In [90]:
```
i = 1
while i <= 5:
    print(i * '*')
    i = i + 1
#we can use this on strings too
```

```
*
**
***
****
*****
```

Lists

In [3]:
```
names = ["john", "kelly", "eva", "mary"]
print(names)
```

```
['john', 'kelly', 'eva', 'mary']
```

In [4]:
```
#getting elements from this list
print(names[0])
```

```
john
```

In [6]:
```
#we can also select a range of vlues from this list
#we will run a code thst collects the first three names
print(names[0:3])
```

```
['john', 'kelly', 'eva']
```

In [7]:
```
print(names[0:-2])
```

```
['john', 'kelly']
```

In [8]:
```
print(names[-4])
```

```
john
```

In [ ]:
```
#there are alot of actions we can  perform on a list
#we will see more in the intermediate level
```

List Methods

In [11]:
```
nums = [1,2,3,4,5]
nums.append(6)
print(nums)
```

```
[1, 2, 3, 4, 5, 6]
```

In [12]:
```
#let us remove an item from list
nums.remove(4)
print(nums)
```

```
[1, 2, 3, 5, 6]
```

In [13]:
```
#we can also find out how many items we have in a given list
print(len(nums))
```

```
5
```

In [14]:
```
#we have generated the number of elements in our list
```

For loops

In [ ]:
```
#we can use the for loops to iterate over a list and acess elements
```

In [16]:
```
nums1 = [1,2,3,4,5,6,7,8,9]
print(nums1)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

In [ ]:
```
#now we want  to print outc each elements individually without the sq bracket
```

In [21]:
```
for item in nums1:
    print(item)
```

```
1
2
3
4
5
6
7
8
9
```

In [22]:
```
#let us find out how items we have using the 'while' loop
i= 0
while i < len(nums1):
    print(nums1 [i])
    i = i + 1
```

```
1
2
3
4
5
6
7
8
9
```

range() function

In [23]:
```
#we can use the range() function to get a list of numbrs
obj = range(10)
print(obj)
```

```
range(0, 10)
```

In [25]:
```
#let us use a for loop
for obj in obj:
    print(obj)
```

```
0
1
2
3
4
5
6
7
8
9
```

In [29]:
```
#let us use two values
obj2 = range(5, 15)
for obj2 in obj2:
    print(obj2)
```

```
5
6
7
8
9
10
11
12
13
14
```

immutable list 'tuple

In [33]:
```
#a tuple is imutable because elements in it can neither be removed or added
cakes = (1,2,3,4,5,6)
cakes[3] = 6
#the error message is a clear indication of tuple immutability
#however, there are actions we can perform on a tuple in our next topic
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_6024\4024452111.py in <module>
      1 #a tuple is imutable because elements in it can neither be removed or added
      2 cakes = (1,2,3,4,5,6)
----> 3 cakes[3] = 6

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

In [ ]:

In [ ]:

In [ ]: