Topics -

1. Operators.
2. If-Else statements.
3. Loops with Else.
4. Iterating by index.

Topic 1 - Operators.

An operator is used to perform some changes between two or more operands.
For example - If we want to add two numbers, we use the "+" operator.

Arithmetic Operators - They are used for arithmetic operations.

In [4]:
```
# They are the basic arithmetic operators (+, -, *, /)
# Note - This "/" is division, and returns the quotient.
# Note - This "//" is integer/floor division, and returns the quotient
# Note - This "%" is modulus operator, and returns the remainder after
# Note - This "**" is the power operator.

print(40+20)

print(40-20)

print(40*20)

print(40/20)

print(5%2)
# The remainder will be 1.

print(5**2)
# This returns 5 to the power 2 i.e 25
```

```
60
20
800
2.0
1
25
```

In [2]:
```
# Difference between division and floor division.

print(10/4)  # This will give exact answer i.e 2.5
print(10//4) # This will round down i.e from 2.5 to 2.
```

```
2.5
2
```

Relational Operators - They are used for comparison.

In [5]:
```
print(4 > 5)
# Since, 4 is not greater than 5, this will return False.
```

```
False
```

In [6]:
```
print (10 < 20)
# Since, 10 is less than 20, this will return True.
```

```
True
```

In [9]:
```
# Similarly, we have greater or than equal to operator.

print(4 >= 4)
```

```
True
```

In [10]:
```
# Less than or equal to operator.

print(5 <= 5)
```

```
True
```

In [12]:
```
# Double equal to (==) operator checks whether both sides are equal o

print(4 == 4)
# This will be True, since 4 is equal to 4.


print(4 == 5)
# This will be False, since 4 is not equal to 5.
```

```
True
False
```

In [13]:
```
# Not equal to (!=) operator checks whether both sides are not equal 

print(4 != 4)
# This will be False, since 4 is equal to 4.

print(4 != 5)
# This will be True, since 4 is not equal to 5.
```

```
False
True
```

Logical Operators - They are used for logic building.

In [15]:
```
# AND operator - This gives 0 if either is 0 (and 1 if both are 1).

print(0 and 0)
print(1 and 0)
print(0 and 1)
print(1 and 1) # Only this will give 1.
```

```
0
0
0
1
```

In [16]:
```
# OR operator - This gives 1 if either is 1 (and 0 if both are 0).

print(0 or 0) # Only this will give 0.
print(1 or 0)
print(0 or 1)
print(1 or 1)
```

```
0
1
1
1
```

In [18]:
```
# NOT operator - This gives the opposite of the input.

print(not 0) # This will give 1/True, since opposite of 0 is 1.
print(not 1) # This will give 0/False, since opposite of 1 is 0.
```

```
True
False
```

Bitwise Operators - They are used on binary values.

In [19]:
```
# Bitwise AND.
print(2 & 3)

# Note - Internally 2 and 3 have first been converted to binary values
```

```
2
```

In [23]:
```
# Bitwise OR.
print(5 | 6)

# Note - Internally 5 and 6 have first been converted to binary values
```

```
7
```

In [27]:
```
# Bitwise XOR - This gives 0 if both are same (and 1 if both are diffe
print(2 ^ 3)

# Note - Internally 2 and 3 have first been converted to binary values
```

```
1
```

In [28]:
```
# Bitwise NOT.
print(~3)

# Note - Internally 3 has first been converted to binary values, then
```

```
-4
```

In [31]:
```
# Bitwise LEFT.
print(4 << 2)

# Internally 4 first been converted to binary values, then all values
```

```
16
```

In [33]:
```
# Bitwise RIGHT.
print(4 >> 2)

# Internally 4 first been converted to binary values, then all values
```

```
1
```

Assignment Operators - They are used to assign values.

In [34]:
```
a = 2

# Here, literal/value 2 is assigned to variable "a" using the assignme

print(a)
```

```
2
```

In [37]:
```
# Assignment operator can be used in conjunction with other operators.


a = 2
a += 5  # This means a = a + 5
print(a)
```

```
7
```

In [39]:
```
a = 10
a -= 2  # This means a = a - 2
print(a)
```

```
8
```

In [40]:
```
a = 4
a *= 8  # This means a = a*8
print(a)
```

```
32
```

In [42]:
```
a = 100
a /= 25  # This means a = a/25
print(a)
```

```
4.0
```

Membership Operators - They are used to identify whether something is present in another thing or not.

In [44]:
```
# in operator - This returns True if something is present.

# We want to check whether "e" is present in "Delhi" or not.
print("e" in "Delhi")

# This returns True because "e" is present in "Delhi".
```

```
True
```

In [46]:
```
# not in operator - This returns True is something is not present.

# We want to check whether "x" is present in "Ayush" or not.
print("x" not in "Ayush")

# This returns True because "x" is not present in "Ayush".
```

```
True
```

In [47]:
```
# These operators work in all cases where we can traverse through some
# For example - strings, lists, tuples etc.

List = [1,2,3,4]
print(5 in List)
```

```
False
```

In [49]:
```
#### Practice - Find the sum of a 3-digit number entered by the user.

number = int(input("Please enter a 3 digit number"))

a = number%10
# This will give us the last digit.
# Suppose number is 345, then 345 % 10 = 5

b = number//10      # This will give us 345//10 = 34
b %= 10             # This will give us 34 % 10 = 4

c = number//100.    # This will give us 345//100 = 3

print("The sum of your number is = ", a+b+c)
```

```
Please enter a 3 digit number345
The sum of your number is =  12.0
```

Topic 2 - If-Else statements.

They are used to create a branching condition i.e we want to give a choice to our program to do either of the given choices based on some logic.

For example-
Let's say we want to login to our Instagram account.
Here, there are only two options - If the login credentials are correct, we can view our account and if they are incorrect, then we get an error message.
This is done through if-else statements.

Suppose we are creating a program which takes an input from the user and tells us whether it is positive or negative.

In [51]:
```
number = int(input("Please enter a number = "))

if number >= 0:
    print("This is a positive number")

else:
    print("This is a negative number")
```

```
Please enter a number = -34
This is a negative number
```

We can also create a program which checks multiple conditions simultaneously.

In [53]:
```
# Let's say "Name" is "Ayush" and "Password" is 1234.
# If this is entered, then we can login, otherwise not.

Name = input("Please enter your name = ")
Password = input("Please enter your password = ")

if Name == "Ayush" and Password == 1234:
    print("Welcome")

else:
    print("Please try again")
```

```
Please enter your name = Anshu
Please enter your password = 6969
Please try again
```

If there are two possibilities, we use if-else, but if there are more than two possibilities, then we use if-elif-else.

If we use if-else inside if-else, then it is called as nested if-else statements.

In [60]:
```
Name = input("Please enter your name = ")
Password = input("Please enter your password = ")

if Name == "Ayush" and Password == 1234:
    print("Welcome")

elif Name == "Ayush" and Password != 1234:
    print("Wrong Password, Please try again")
    Password = input("Enter password again = ")
    if Password == 1234:
        print("Welcome")
    else:
        print("Wrong password again")

else:
    print("Please try again")
```

```
Please enter your name = Ayush
Please enter your password = 12345
Wrong Password, Please try again
Enter password again = 1234567
Wrong password again
```

Write a program to enter the largest of three numbers entered by the user.

In [62]:
```
a = int(input("Please enter the first number = "))
b = int(input("Please enter the second number = "))
c = int(input("Please enter the third number = "))

if a > b and a > c:
    print("First number is the largest")

elif b > a and b > c:
    print("Second number is the largest")

elif a==b==c:
    print("All numbers are equal")

else:
    print("Third number is the largest")
```

```
Please enter the first number = 12
Please enter the second number = 15
Please enter the third number = 7
Second number is the largest
```

Write a menu driven program for a calculator.

In [64]:
```
a = int(input("Please enter the first number = "))
b = int(input("Please enter the second number = "))

operation = input("Please select the operation to perform = ")

if operation == "+":
    print("The sum is = ", a+b)

elif operation == "-":
    print("The difference is = ", a-b)

elif operation == "*":
    print("The product is = ", a*b)

else:
    print(a/b)
```

```
Please enter the first number = 5
Please enter the second number = 6
Please select the operation to perform = *
The product is =  30
```

Modules - They are basically files which have some in-built methods/functions written inside them, which can be used directly once we import them.

These modules are written by someone else and can be used by anyone.
This is done to ensure code reusability (which is one of the key principles of python).

Math module - It contains all the functions related to mathematics.

In [65]:
```
# To import a module, we use import keyword followed by the module nam

import math
```

In [66]:
```
# Now, we are going to use the methods written inside this module.
# For example - we want to find the factorial, we can find it as follo

# To find the factorial of 5.
math.factorial(5)
```

Out[66]: 120

In [67]:
```
# Floor - It rounds down the value to the nearest integer.
math.floor(6.9)
```

Out[67]: 6

In [68]:
```
# Ceil - It rounds up the value to the nearest integer.
math.ceil(6.9)
```

Out[68]: 7

In [69]:
```
# Sqrt - It is used to find the square root of any number.
math.sqrt(225)
```

Out[69]:
```
15.0
```

Keywords module - It contains the list of keywords (reserved words by interpreter).

In [70]:
```
import keyword
```

In [71]:
```
# To print the list of keywords, we use kwlist.
print(keyword.kwlist)
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await',
'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'excep
t', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is',
'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try',
'while', 'with', 'yield']
```

Random module - It is used to generate random numbers.

In [72]:
```
import random
```

In [74]:
```
# To generate a random number between 0 to 100, we use a method calle
print(random.randint(0,100))
```

```
27
```

In [77]:
```
# To return a sub-sample from a population sample, we use a method ca
# Here, from the given list, we have selected a sub-list of 3 random e
random.sample([1,2,3,4,5,6,7,8,9,10], 3)
```

Out[77]:
```
[6, 5, 2]
```

DateTime module - It is used for methods related to date and time.

In [78]:
```
import datetime
```

In [80]:
```
# To show the current datetime, we use the datetime.now() method

print(datetime.datetime.now())
```

```
2023-12-12 11:48:12.017605
```

Topic 3 - Loops

They are used to iterate/move/pass over any iterable object such as string, list, tuple etc. again and again.

For example-
If we are on any website such as Flipkart, and there we have containers which have the products listed.
Then, suppose we want to fill these products into these containers, we can do this using loops.
We are going to fetch the data from the database using loops and load them into these containers.

In python, we have two types of loops -

1. while loops - They are used when we know the condition of stopping.
2. for loops - They are used when we know how many times the loop needs to be run.

While Loops

In [2]:
```
# Suppose we are trying to print a table.

number = int(input("Please enter a number"))

i = 1
while i <= 10:
    print(number, "*", i, "=", number*i)
    i += 1
```

```
Please enter a number5
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
5 * 10 = 50
```

While loops with else. - Generally else is used with if statements, but in python else can be used with while loop also.

In [3]:
```
x = 1

while x < 3:
    print(x)
    x += 1

else:
    print("Limit crossed")
```

```
1
2
Limit crossed
```

To understand this concept better, we are going to write a program called as guessing game.
Here, we will be having a jackpot number, and our aim is to guess that number.

After each attempt, we will get a hint which will tell us whether we are far or near to the actual jackpot number.

In [6]:
```
# Step 1 - We need to generate a random number which will serve as our
import random
jackpot = random.randint(1, 100)

# Step 2 - We ask the user to guess the number by taking an input.
guess = int(input("Guess the number = "))

# Step 3 - First we check whether the guess is greater or smaller than
# And, since the program must run again and again until we hit jackpot

# We will also maintain a counter to check how many attempts it took u
# Initially, counter = 1, since we have already made one attempt at gu
counter = 1
while guess != jackpot:
    if guess < jackpot:
        print("Guess higher")

    elif guess > jackpot:
        print("Guess lower")

    guess = int(input("Guess the number again = ")) # This is done to
    counter += 1
    # This is done to increment the counter, so that we can look at ho

else:
    print("Congratulations, You have hit the jackpot")
    print("You took the following attempts = ", counter)
```

```
Guess the number = 75
Guess lower
Guess the number again = 45
Guess lower
Guess the number again = 33
Congratulations, You have hit the jackpot
You took the following attempts =  3
```

For Loops

In [8]:
```
# Here, i is the iterator (we can take anything else also).
# range function gives us the range of the function between two numbe
# range(1,11) means numbers will be generated from 1 to 10 (11 is exc

for i in range(1,11):
    print(i)

# We can imagine, it is written as - for i in 1,2,3,4,5,6,7,8,9,10
# Here, first i is 1, then 2, then 3 and so on till 10.
# We are printing i after each time until loop is ongoing.
```

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

In [10]:
```
# Range also has a third parameter called as "step", which is used to

for i in range(1,11,2):
    print(i)

# This time, we are actually taking every second element, and then pri
# We can imagine, it is written as - for i in 1,3,5,7,9
# First is 1, then 3 (we skipped 2), then 5 (we skipped 4) and so on.
# And we are printing each element one by one.
```

```
1
3
5
7
9
```

In [11]:
```
# We can also do this in the reverse order as follows.
# This time we start from 10 and go till 0 (0 is excluded), and step s
# moving in the negative direction.

for i in range(10,0,-1):
    print(i)
```

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

In [12]:
```
# For loop can also iterate over other iterables such as strings, list

# Iteration over string.
for i in "Ayush":
    print(i)
```

```
A
y
u
s
h
```

In [13]:
```
# Iteration over list.
for i in [1,2,3,4,5]:
    print(i)
```

```
1
2
3
4
5
```

In [14]:
```
# Iteration over tuple.
for i in (1,2,3,4,5):
    print(i)
```

```
1
2
3
4
5
```

In [15]:
```
# Iteration over sets.
for i in {1,2,3,4,5}:
    print(i)
```

```
1
2
3
4
5
```

Problem - The current population of a town is 10,000. This increases at the rate of 10% per year.
Write a program to find the population of the program after 10 years.

In [17]:
```
population = 10000

for i in range(1,11):
    print("Year = ", i, "Population = ", population)
    population = population + (0.10*population)
```

```
Year =  1 Population =  10000
Year =  2 Population =  11000.0
Year =  3 Population =  12100.0
Year =  4 Population =  13310.0
Year =  5 Population =  14641.0
Year =  6 Population =  16105.1
Year =  7 Population =  17715.61
Year =  8 Population =  19487.171000000002
Year =  9 Population =  21435.888100000004
Year =  10 Population =  23579.476910000005
```

Problem - Find the sum of this series where the number of terms is given as input by the user.

1/1! + 2/2! + 3/3! .....upto n terms.

In [6]:
```
import math
number = int(input("Please enter a number = "))

result = 0
for i in range(1, number+1):
    result = result + (i/math.factorial(i))

print("The final result is = ", result)
```

```
Please enter a number = 3
The final result is =  2.5
```

END OF SESSION 2.