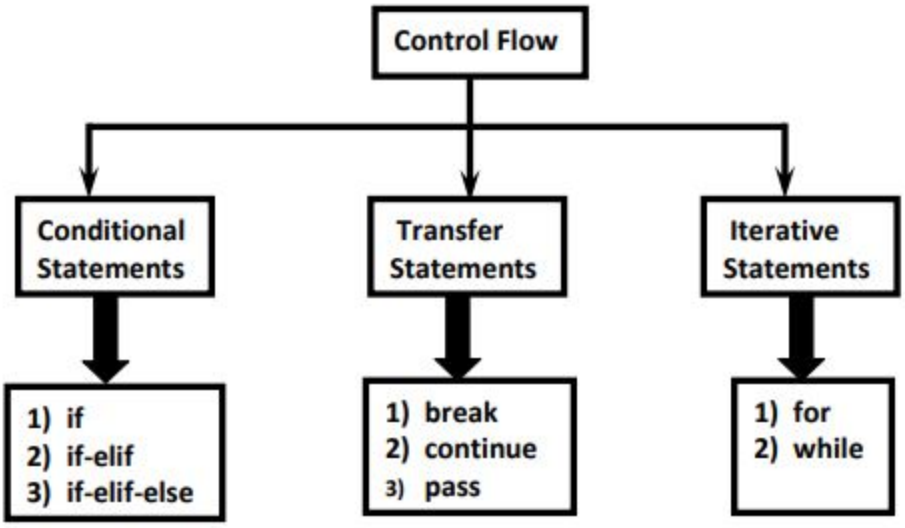
5. Flow Control

Flow control describes the order in which statements will be executed at runtime.

In [1]:
```
from IPython.display import Image
Image(filename="flow_control.jpg")
```

Out[1]:



1. Conditional Statements

A. if

```
if condition : statement

    or

if condition :
    statement - 1
    statement - 2
    statement - 3
if condition is true then statements will be executed
```

In [3]:
```
name = input("Enter Name:")
if name=="Pankaj":
    print("Hello Pankaj Good Morning")
print("How are you")
```

```
Enter Name:Pankaj
Hello Pankaj Good Morning
How are you
```

B. if-else

```
if condition:
    Action-1
else:
```

```
    Action-2

if condition is True then Action-1 will be executed otherwise Action-2 will be
executed
```

In [4]:

```
name = input("Enter Name : ")
if name == "Pankaj":
    print("Hello Pankaj Good Morning")
else:
    print("Hello Guest GoodMorning")
print("How are you")
```

```
Enter Name : pankaj
Hello Guest GoodMorning
How are you
```

C. if-elif-else

```
if condition1:
    Action-1
elif condition2:
    Action-2
elif condition3:
    Action-3
elif condition4:
    Action-4
 ...
else:
    Default Action

Based condition the corresponding action will be executed.
```

In [5]:

```
brand = input("ENter Your Favourite Brand:")
if brand=="RC":
    print("It is childrens brand")
elif brand=="KF":
    print("It is not that much kick")
elif brand=="FO":
    print("Buy one get one Free")
else:
    print("Other Brands are not recommended")
```

```
ENter Your Favourite Brand:KF
It is not that much kick
```

Note:

```
1. else part is always optional

 Hence the following are various possible syntaxes.
 1. if
 2. if - else
 3. if-elif-else
 4.if-elif

2. There is no switch statement in Python
```

Q. Write a program to find biggest of given 3 numbers from the input?

In [6]:
```
n1 = int(input("ENter First Number : "))
n2 = int(input("ENter Second Number : "))
n3 = int(input("ENter Third Number : "))

if n1>n2 and n1>n3:
    print("Biggest Number is : ",n1)
elif n2>n3:
    print("Biggest Number is : ",n2)
else:
    print("Biggest Number is : ",n3)
```

```
ENter First Number : 13
ENter Second Number : 34
ENter Third Number : 54
Biggest Number is :  54
```

2. Iterative Statements

If we want to execute a group of statements multiple times then we should go for Iterative statements.
Python supports 2 types of iterative statements.

1. for loop
2. while loop

A. for Loop

If we want to execute some action for every element present in some sequence(it may be string or collection)then we should go for for loop.

Syntax:

```
for x in sequence :
    body
```

where sequence can be string or any collection.
Body will be executed for every element present in the sequence.

To print characters present in the given string

In [8]:
```
s = "pankaj"
for x in s:
    print(x)
```

```
p
a
n
k
a
j
```

To print characters present in string index wise

In [1]:
```
s = "pankaj"
i = 0
for x in s:
```

```
    print("at index ",i,":",x)
    i = i+1
```

```
at index  0 : p
at index  1 : a
at index  2 : n
at index  3 : k
at index  4 : a
at index  5 : j
```

To print Hello 5 times with count

In [6]:
```
for x in range(5):
    print("Hello : ",x)
```

```
Hello :  0
Hello :  1
Hello :  2
Hello :  3
Hello :  4
```

To print Hello 5 times (1 - 5)

In [8]:
```
for x in range(1 , 6):
    print("Hello : ",x)
```

```
Hello :  1
Hello :  2
Hello :  3
Hello :  4
Hello :  5
```

To print 10 to 1 in decreasing order

In [11]:
```
for x in range(10,0,-1):
    print(x,end='\t')
```

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

print sum of number in given in a single line

In [14]:
```
l = list(map(int,input().split()))
sum = 0
for x in l:
    sum += x
print("Sum is :",sum)
```

```
5 6 2 4 1
Sum is : 18
```

B. while loop

If we want to execute a group of statements iteratively until some condition false,then we should go for while loop.

Syntax:

```
while condition:
    body
```

To print numbers from 1 to 10 even number by using while loop

In [15]:
```
x = 1
while x <= 10:
    if x%2 == 0:
        print(x)
    x = x+1
```

```
2
4
6
8
10
```

Write a program to prompt user to enter some name until entering Pankaj

In [16]:
```
name = ""
while name!="Pankaj":
    name = input("Enter Name :")
print("Thanks for confirmation")
```

```
Enter Name :pankaj
Enter Name :rahul
Enter Name :shyam
Enter Name :ram
Enter Name :Pankaj
Thanks for confirmation
```

Print a string in reverse order

In [40]:
```
s = "hello"

while s:
    print(s[-1], end=' ')
    s = s[:-1]
```

```
o l l e h
```

Q. count no of vowels in given string

In [41]:
```
s = input().lower()
i = 0
count = 0
while i < len(s):
    c = s[i]
    if (c == 'a') or (c == 'e') \
        or (c == 'i') or (c == 'o') \
        or (c == 'u'):
            count += 1
    i += 1

print(f"{s} contains {count} vowels")
```

```
pankaj kumar
pankaj kumar contains 4 vowels
```

In [42]:
```
s = input().lower()
i = 0
count = 0
while i < len(s):
    c = s[i]
    conds = [
        (c == 'a'),(c == 'e'),
        (c == 'i'),(c == 'o'),
```

```
        (c == 'u') ]
    if any(conds):
        count += 1
    i += 1

print(f"{s} contains {count} vowels")
```

```
pankaj kumar
pankaj kumar contains 4 vowels
```

using in Operator

In [43]:

```
s = input().lower()
i = 0
count = 0
while i < len(s):
    if s[i] in 'aeiou':
        count += 1

    i += 1
print(f"'{s}' contains {count} vowels.")
```

```
pankaj kumar
'pankaj kumar' contains 4 vowels.
```

Q. Prime Number

In [48]:

```
#  1.

num = int(input("Number: "))
check = 2

# num = 13

if num % 2 == 0:
    print("not prime")
    exit()

if num % 3 == 0:
    print("not prime")
    exit()
if num % 4 == 0:
    print('not prime')
    exit()
...

if num % 12 == 0:
    print('not prime')
    exit()

print('prime')
```

```
Number: 127
prime
```

In [47]:

```
#  2.

num = int(input("Enter a number: "))
if num <= 1:
    print("Not a Prime")
else:
    check = 2
    while check < num:
```

```
        if num % check == 0:
            print("Not a Prime")
            break
        check += 1
    else:
        print("Prime")
```

```
Enter a number: 127
Prime
```

In [49]:

```
#  3.

num = int(input("Enter a number: "))
if num <= 1:
    print("Not a Prime")
else:
    check = 2
    while check < num // 2:
        if num % check == 0:
            print("Not a Prime")
            break
        check += 1
    else:
        print("Prime")
```

```
Enter a number: 127
Prime
```

In [50]:

```
#  4.

num = int(input("Enter a number: "))
if num <= 1:
    print("Not a Prime")
else:
    if num <= 3:
        print("prime")
    elif num % 2 == 0 or num % 3 == 0:
        print("Not Prime")
    else:
        check = 5
    while check < (num**(1/2))+1:
        if num % check == 0:
            print("Not a Prime")
            break
        check += 2
    else:
        print("Prime")
```

```
Enter a number: 127
Prime
```

C. Infinite Loop

In [13]:

```
# i = 0
# while True:
#     i = i+1
#     print("Hello",i)
```

D. Nested Loops:

Sometimes we can take a loop inside another loop, which are also known as nested loops

In [19]:
```
for i in range(4):
    for j in range(4):
        print("i=",i," j=",j)
```

```
i= 0  j= 0
i= 0  j= 1
i= 0  j= 2
i= 0  j= 3
i= 1  j= 0
i= 1  j= 1
i= 1  j= 2
i= 1  j= 3
i= 2  j= 0
i= 2  j= 1
i= 2  j= 2
i= 2  j= 3
i= 3  j= 0
i= 3  j= 1
i= 3  j= 2
i= 3  j= 3
```

Q. Write a program to dispaly *'s in Right angled triangled form

In [23]:
```
n = int(input("ENter number of row:"))
for i in range(1,n+1):
    for j in range(1,i+1):
        print("*",end=" ")
    print()
```

```
ENter number of row:5
*
* *
* * *
* * * *
* * * * *
```

Alternative way:

In [24]:
```
n = int(input("Enter number of rows:"))
for i in range(1,n+1):
    print("* " * i)
```

```
Enter number of rows:5
*
* *
* * *
* * * *
* * * * *
```

3. Transfer(Jump) Statements

A. break:

We can use break statement inside loops to break loop execution based on some condition.

In [26]:

```
for i in range(10):
    if i == 5:
        print("Processing is enough...plz break")
        break
    print(i)
```

```
0
1
2
3
4
Processing is enough...plz break
```

B. continue:

We can use continue statement to skip current iteration and continue next iteration.

Print 1 to 7 skip 5

In [27]:
```
for i in range(1,8):
    if i == 5:
        continue
    print(i)
```

```
1
2
3
4
6
7
```

4. loops with else block

Inside loop execution,if break statement not executed ,then after completion of loop else part will be executed.

else means loop without break

In [28]:
```
cart = [10,20,30,40,50]
for item in cart:
    if item>=500:
        print("We cannot process this order")
        break
    print(item)
else:
    print("Congrats ...all items processed successfully")
```

```
10
20
30
40
50
Congrats ...all items processed successfully
```

In [30]:
```
i = 1
while i<=5:
    print(i)
    i = i+1
```

```
else:
    print("all done !!")
```

```
1
2
3
4
5
all done !!
```

In [31]:
```
i = 1
while i<=5:
    print(i)
    i = i+1
    if i==3:
        break
else:
    print("all done !!")
```

```
1
2
```

5. Pass Statement

pass is a keyword in Python.

In our programming syntactically if block is required which won't do anything then we can define that empty block with pass keyword.

pass
 |- It is an empty statement
 |- It is null statement
 |- It won't do anything

In [33]:
```
if True:
    pass
```

In [34]:
```
def m1():
    pass
```

use case of pass:

Sometimes in the parent class we have to declare a function with empty body and child class responsible to provide proper implementation. Such type of empty body we can define by using pass keyword. (It is something like abstract method)

In [36]:
```
for i in range(100):
    if i%9 == 0:
        print(i)
    else:
        pass
```

```
0
9
18
27
36
45
```

54
63
72
81
90
99

6. del Statement

del is a keyword in Python.

After using a variable, it is highly recommended to delete that variable if it is no longer required,so that the corresponding object is eligible for Garbage Collection.

We can delete variable by using del keyword.

In [37]:
```
x = 10
del x
print(x)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_9476/3309143265.py in <module>
      1 x = 10
      2 del x
----> 3 print(x)

NameError: name 'x' is not defined
```

Note:We can delete variables which are pointing to immutable objects.But we cannot delete the elements present inside immutable object.

In [38]:
```
s = "pankaj"
print(s)
del s[0]
```

```
pankaj
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_9476/1648751915.py in <module>
      1 s = "pankaj"
      2 print(s)
----> 3 del s[0]

TypeError: 'str' object doesn't support item deletion
```

Difference between del and None:

In the case del, the variable will be removed and we cannot access that variable(unbind operation)

But in the case of None assignment the variable won't be removed but the corresponding object is eligible for Garbage Collection(re bind operation). Hence afterassigning with None value,we can access that variable.

In [39]:
```
a = "pankaj"
a = None
print(a)
```

```
None
```

In [ ]: