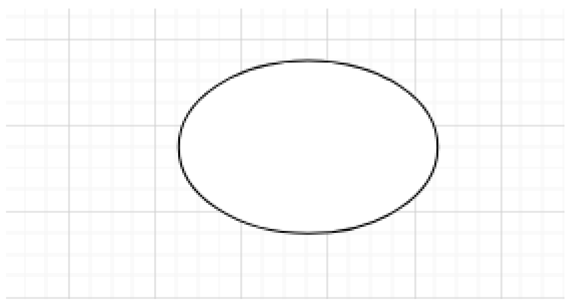
INTRODUCTION TO PROBLEM SOLVING AND PROGRAMING
A CRASH CORSE

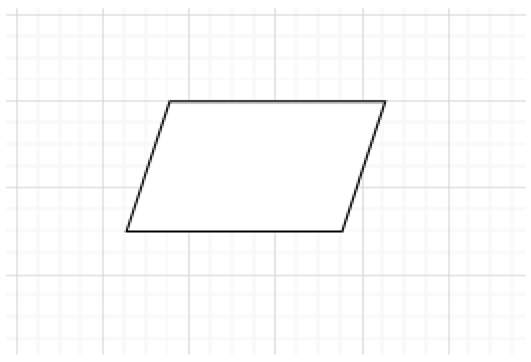
BY EEMAN MAJUMDER

Topic 1- Flowcharts

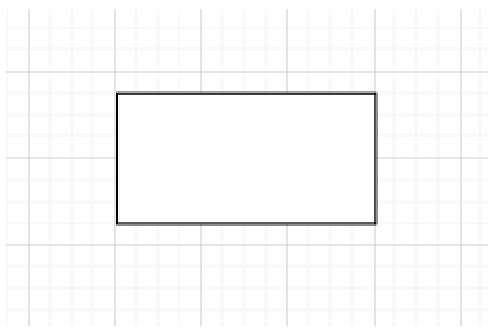
START/STOP



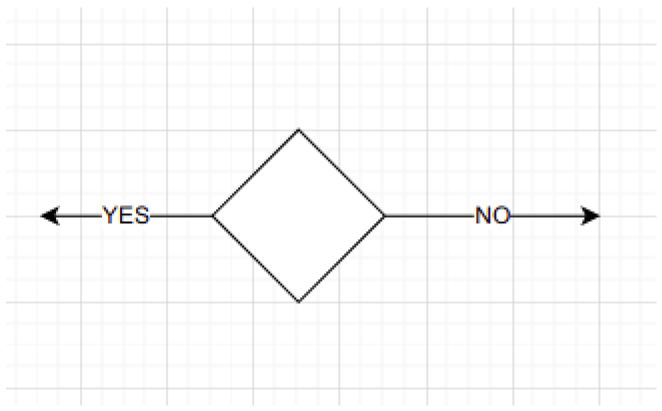
INPUT/OUTPUT



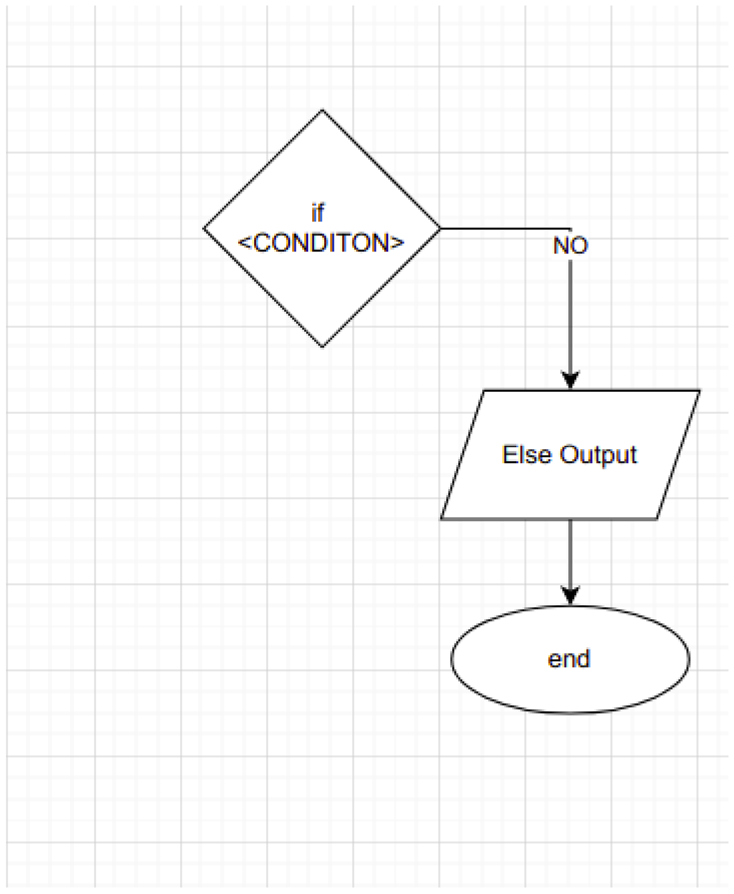
ASSIGNMENT/OPERATION



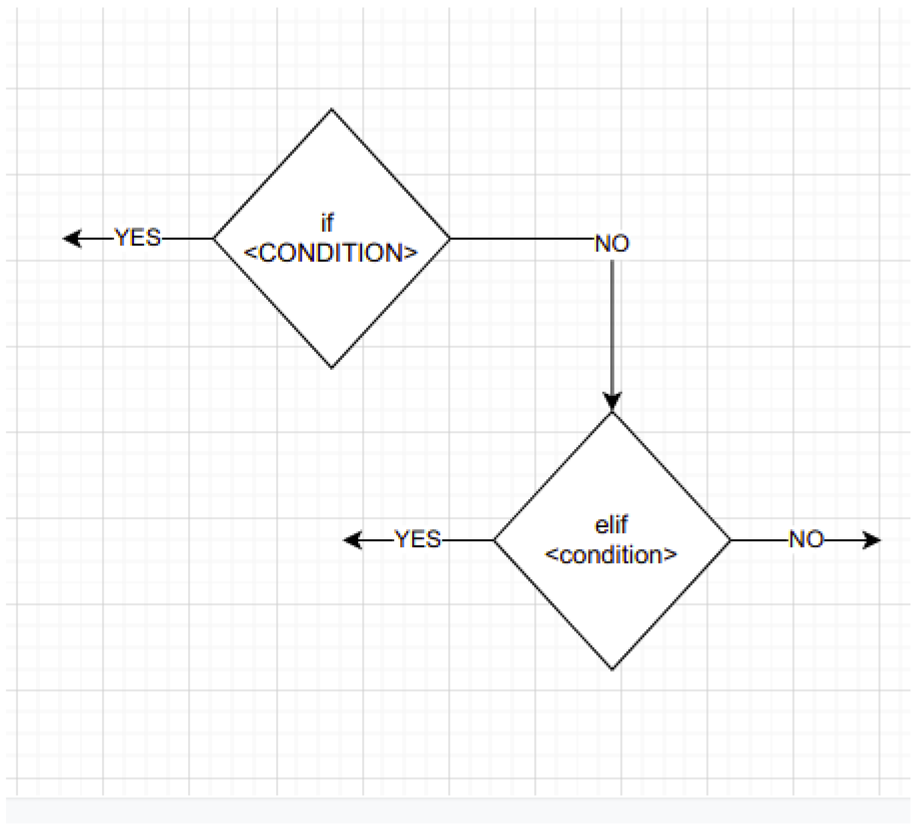
IF STATEMENT



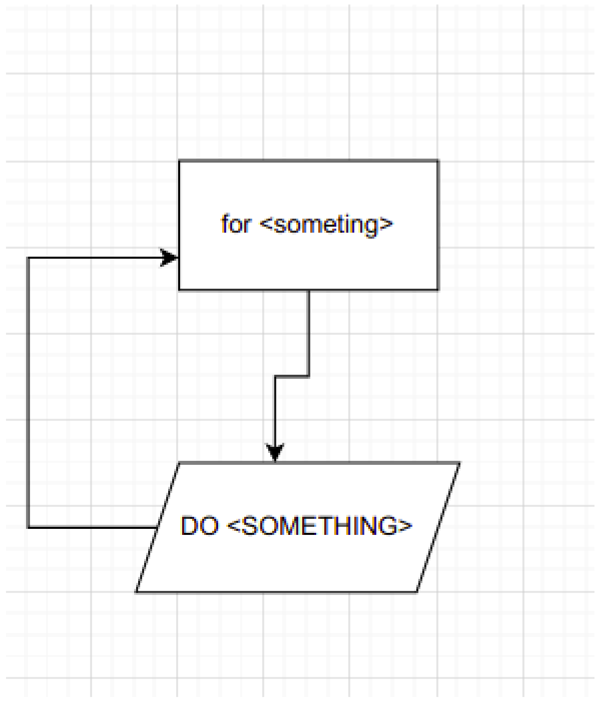
ELSE STATEMENT



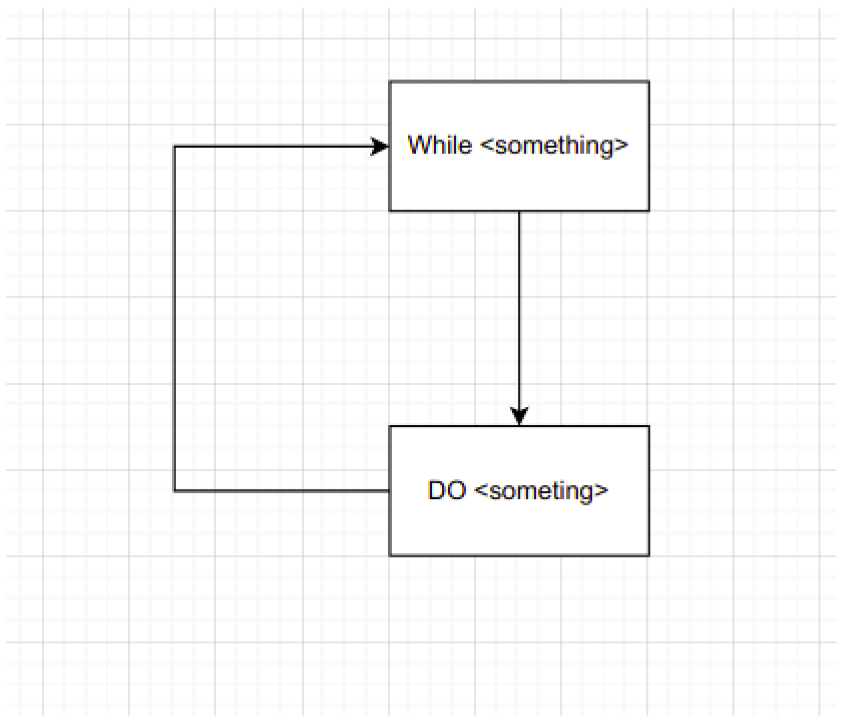
ELIF STATEMENT



FOR LOOP



WHILE LOOP



Topic 2- ALGORITHM

Instruction:

1. Always start with (1. Start)
2. Always end with (<last line number>. End/Stop)
3. Always give line number
4. Always type in simple sentences

Example:

```
1. Start
2. Ask user to give the value of side of square (s)
3. Area = S**2
4. Print (Area) as output as the area of square
5. End
```

Topic 3- PSEUDOCODE

Instruction:

1. Always start with {
2. Always end with }
3. Use technical terms in the pseudocode
4. Use code terms

5. Always number the lines

Example:

1. {
2. start/Initialize
3. s=Input the value of side of square
4. a=s**2
5. print(a)
6. End
7. }

Topic 4- BASICS OF PYTHON

DATA TYPES:

```
# types of data in python3
def types_of_data():
    # _____________Types of data in python3____________________

    # strings-anything between "________" or this '_____________' is known as string
    a = ""
    print(type(a))

    # Lists-anything between [_________] is called list
    l = []
    print(type(l))

    # Tuple-anything between (__________) is called tuple
    t = ()
    print(type(t))

    # Dictenory-Anything between {__________} is called dictenory
    d = {}
    print(type(d))
```

MORE ABOUT STRINGS:

```
def string_indexing():
    # so strings can edited in many ways
    # lets first learn about how to count strings
    #  012345
    # "Sample"
    # here index position of S is 0 , a is 1 and so on
    # For example
    s = "sample"
    print(s.index('s'))


def string_slicing():
    # lets learn how to manupulate strings
    # for example
    s = "sample"
    print(s[4])
    print(s[0:4])
    print(s[5])
    print(s[-1:-5])
    print(s[1:])
    print(s[::-1])  # to reverse a string

def string_substituion():
    # strings can be modified by index
    s = 'sample'
    s[3] = 'r'
    print(s)
def ways_to_make_anything_string():
    #method one
    #direct assignment
    a='sample'

    #method two
    #using str()
    a=1234
    a=str(a)
```

BASIC ARITHMETIC:

```
#basic calculations and logic
def arithmetics():
    #arithmetics
    print(2+2)#addition
    print(2-2)#subtraction
    print(2*2)#multiplication
    print(2/2)#division
    print(2**2)#exponent
    print(2%2)#modulus
    print(2//2)#Float
```

```python
def equalsto_aur_double_equals_ka_difference():
    #0=True
    #1=False
    # == is for comparing two values
    # = is for adding a value to a variable
    a=100
    b=200
    print(a==b)
    print('a is: ',a,'b is: ',b)
    b=a
    print(a==b)
    print('a is: ',a,'b is: ',b)
def variable_management():
    #variable management
    a=100
    b=200
    a=b
    print('a: ',a , 'b: ', b )
    c=a+b
    print('a: ',a , 'b: ', b ,'c; ',c)
    b=c+a
    print('a: ',a , 'b: ', b ,'c; ',c)
    b=a
    print('a: ',a , 'b: ', b ,'c; ',c)
    a=b+c
    print('a: ',a , 'b: ', b ,'c; ',c)
    print(a)
```

BASIC INPUT/OUTPUT:

```python
#basic Functions and user input and output
def basic_funtions():

    #print() funtion- used to output a value to the user
    print('hello world')
    a="bye world"
    print(a)

    #input() funtion- used to take input from a user
    #syntax a=input('enter a number: ')
    #       |           |
    #    a variable     |
    #               the line it will show to the user
```

```
   #output<<
   #>>>enter a number:
   b=input('a number:')
```

MORE ABOUT LISTS:

```
#more about lists
def list_indexing():
   #list can do everything string can and a little bit more
   # so lists can be edited in many ways
   # lets first learn about how to count lists
   #  0 1 2 3 4 5
   # [S,a,m,p,l,e]
   # here index position of S is 0 , a is 1 and so on
   # For example
   l = ['s','a','m','p','l','e']
   print(l.index('s'))

def nested_lists():
   l=['1','2','3','4',['1','2','3','4'],'5']
   print(l[4])
   print(l.index([1,2,3,4]))
def list_slicing():
   # lets learn how to manupulate lists
   # for example
   l = ['s','a','m','p','l','e']
   print(l[4])
   print(l[0:4])
   print(l[5])
   print(l[-1:-5])
   print(l[1:])
   print(l[::-1])  # to reverse a list
def nested_list_slicing():
   l=['1','2','3','4',['1','2','3','4'],'5']
   print(l[4][3])
   print(l[4][0])

def list_substituion():
   # lists can be modified by index
   l = ['s','a','m','p','l','e']
   l[3] = 'r'
   print(l)
def ways_to_make_anything_list():
   #method one
   #direct assignment
   l = ['s', 'a', 'm', 'p', 'l', 'e']

   #method two
   #using str()
   a='lists'
   a=list(a)
   print(a)

def list_funtions():
   #to add stuff to the list


   #the list.append() funtion- adds the given value to the end of list
   l=['1','2']
   l.append('dum')
   print(l)

   #the slicing way
   l[0]='13'
   print(l)

   #the list.extend() function- adds the given value to the last
   l.extend('hello world')
   print(l)

   #the list.insert()  function- inserts value to a given index
   l.insert(0,'hello')
   print(l)


   #to delete stuff from list

    #the list.pop(the thing you wanna delete) function- deletes the given digit from
        list and prints it out
   l.pop(1)
   print(l)

    #the list.remove(the thing you wanna delete) funtion- delets given digit from
        list
   l.remove('2')
   print(l)
```

MORE ABOUT TUPLES:

```
def every_feature_of_tuples():
   t=('1','2','3','4')
   #tupele are immutable ,values of tuple cant be changed
   #tuple indexing and slicing is same as list
   #how to form a tuple
   a=10
   a=tuple(a)
   a=()#empty tuple
```

MORE ABOUT DICTIONARIES:

```
def all_about_dictionaries():
   #dictionaries are like lists but they are not ordered
   #dictionaries are made up of key value pairs
   #keys are unique and values can be anything
   #dictionaries are used to store data
    #dictionaries
   d={'a':'1','b':'2','c':'3'}
   #for getting the key
   print(d.key('1'))
   #for getting value
   print(d.values('a'))

   #slicing and indexing
   print(d['a'])
```

IF / ELSE STATEMENT:

```
def if_else():
   #if else is used to compare values and execute a block of code
   #if <condition> :
   #   <block of code>
   #else:
   #   <block of code>
   a=10
   if a==10:
       print('a is 10')
   else:
       print('a is not 10')
```

IF / ELSE / ELIF STATEMENT:

```python
def if_else_elif():
    #if else elif is used to compare values and execute a block of code
    #if <condition> :
    #    <block of code>
    #elif <condition> :
    #    <block of code>
    #else:
    #    <block of code>
    a=10
    if a==10:
        print('a is 10')
    elif a==20:
        print('a is 20')
    else:
        print('a is not 10 or 20')
```

FOR LOOP:

```python
def for_loop():
    #for loop is used to iterate over a list/range
    #for <variable> in <list>:
    #    <block of code>
    #for <variable> in <range>:
    #    <block of code>
    l=['1','2','3','4']
    for i in l:
        print(i)

    for j in range(1,11):
        print(j)
```

WHILE LOOP:

```python
def while_loop():
    #while loop is used to loop using a condition
    #while <condition>:
    #    <block of code>
    i=1
    while i<=10:
        print(i)
        i+=1
```

BREAK AND CONTINUE:

```
def break_and_continue():
   #break and continue are used to stop a loop
   #break:
   #   <block of code>
   #continue:
   #   <block of code>
   i=1
   while i<=10:
       if i==5:
           i+=1
            continue # continue is used to skip the current iteration and go to the
      next iteration
       print(i)
       i+=1
   while i<=10:
       if i==5:
           break # breaks loop and goes to next line
       print(i)
       i+=1
```

DEF STATEMENT:

```
 def def_STATEMENT():
       #def is used to define a function
       #used to make new functions
       #def <function name>(<parameters>):
       #   <block of code>
```

CLASS:

```
def class_statement():
       #class is used to define a class(collection of functions)
       #example:
       #class <class name>:
       #   def <function name>(<parameters>):
       #       <block of code>
       #   def <function name>(<parameters>):
       #       <block of code>
```

MODULE:

```
def modules():
       #modules are used to group functions and classes together
       #example:
       #class <class name>:
       #   def <function name>(<parameters>):
       #       <block of code>
       #def <function name>(<parameters>):
       #   <block of code>
       #def <function name>(<parameters>):
       #   <block of code>

       # how to import modules
       #import <module name>
```

ARRAY:

```
def array():
   # array is just a fancy name for list
   array=[]
   #everything is same as list
```