PART- B

6A * Pogram to multiply two matrıas using nested loops

```
⇒ take a 3x3 matrix
A = [[12, 7, 3],
     [4, 5, 6],
     [7, 8, 9]]
take a 3x4 matrix
B = [[5, 8, 1, 2],
     [6, 7, 3, 0],
     [4, 5, 9, 1]]
result = [[0, 0, 0, 0],
          [0, 0, 0, 0],
          [0, 0, 0, 0]
  Iterating by row of A
  for i in range (len (A)):
      Iterating by coloom by B
      for j in range (len (B[0])):
          Interating by rows of B
          for k in range (len(B))
              result [i][j] += A [i][k] * B[k][j]
      for r in result :
          print (r)
```

output :

```
[114, 160, 60, 27]
[74, 97, 73, 14]
[119, 157, 112, 23]
```

19761A0541

IA Incremental mdevelopment :-

incremetal development is a programming tactic i use everyday. the goal is to avoid debugging large complex programs. instead, i incrementally write code and then test to ensure each new code works as expected

steps for development :-

⇒ write preudocode line by line on how logic works.

→ write code line by in python

⇒ Run program and validtate code works with dummy data.

→ Repeat steps 2-3 until your function works as antipated.

* To deal with increasingly complex programs, you might want to try a process called incremental development.

As an example, suppose you wants to find the distance between two points given by the coordinates (x1, y1) and (x2, y2). By the ~~co~~ pythoryoner theorem the distance is

$$distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

The first step is to consider what a distance function should look like in python in other words, what are the inputs (parameter) and what is the output (return values).

in this case, the input are two points, which you can reprenent using four numbers. the return value is the distance, which is a floating points value.

```
ex:  point_one = (1,1)
     point_two = (2,3)
   + slope is 2
   def slope of line (point_one, point_two
 x- index = 0
 y- index = 1
numerator = point-two [y-index] - point-one
demominator = point-two [x- index ] - point-0
 return numerator, demominator
print (slope-of-line ( point-one, point-two))
 (2,1)
```

9A. Encapsulation and generalization :-

Encapsulation :-

Encapsulation is one of the fundamental concept in object-oriented programming (OOP). It describes the idea of wrapping data and the method that work on data within one unit.

⇒ This puts restriction on accessing variables and methods directly and can prevent the accidental modification of data.

⇒ To prevent accidental change, an object's variable can only be changed by an object's method.

⇒ Those types of variables are known as private variable.

A class is an example of encapsulation as encapsulates all the data that is member functions, variables, etc

generalization :

converting a subclass type into a super class type is called generalization because we are making the subclass to become more general and its scope is widening. this is also called widening or up casting widening is safe because the classes will become more general.

For example, if we say car is, it was a vechicle, there will be no objection thus Java compiler will not ast for cast operation to generization example for Encapulation:

```
clam Base:
  def. . . init-- (Self);
    # protected member
      self. _a=2.
  Creating a derivd clan
  clan derived (Base):
    def - - - init - (Self):
  Base - - - init - - - (self)
  print ('calling protected member of base clas.
  print (self --a)
  obj1 = Derived (')
  obj2 = Bao ( )
  print (obj2.a)
```

example for Generalization:

```
Clam father{
  Public void work()
{
  System.out.println ("Earning father");
}}
Clas son extende father{
  public void play ()
  {
    System.out.println ("Enjoying son");
  }}
```

```
Class main {
  Public static void main (string [] args)
 {
   father.work();
   // uncomment next line to see the error
   // father.play();
 }}
```

Output

```
Earning father.
```

ॐ

19781A0591 1

1	b
2	b
3	a
4	c
5	b
6	b
7	b
8	b
9	d
10	d
11	a
12	c
13	b
14	b
15	c
16	d
17	b
18	d
19	c
20	a

PART - A

1A Feature of python:

⇒ Easy to use.

⇒ High Level Language

⇒ Expressive Language

⇒ interpreted

⇒ platform independent

⇒ open source

⇒ Object - oriented language

⇒ High standard Libany

⇒ GUI programming

⇒ integrated

⇒ Extensible.

2A: Differenw Between interactive mode And Script mode.

interactive mode :-

⇒ instruction are given in fronut of pyton promt (e.g, >>> or In[]:) in pyton shell

⇒ python carries out the given instructions and shows the result there itself

Script mode:-

Python instructions are stored in a file generally with a '.py' extension name and are excecuted together in one go as a unit. the saved instruction are Known as python script or python program.

3A.

```
a = int (input (" enter firsut number:"))

b = int (input (" enter second number:"))

 sum = a+b

Print ( " Sum of two natural numbers is; ", Sum).
```

ex: a=10
b=20

Output = sum = a+b = 30

74A Four methods of turtle module in python

* forward ( )

Description :- Creates and returns a new tutle object

Parameter :- amount

* right ()

Des :- Turns the turtle clock wise

Parameter :- angle

* heading()

Des :- Returns the current heading

Parameter: None

* goto()

Des : move the turtle to paisition x,y

Parameter: x,y

5A importance of indentation in python

⇒ python uses indentation to highlight the blocks of code.

⇒ whitespace is used for indentation in python.

⇒ All statements with the same distance to the right belong to the block of code

⇒ If a block has to be more deeply neted, it is simply indented further to the right.