17 Feb, 2022

29 March, 2022

DATE / /20
PAGE NO.

Python

- **Python :-**

Python is a programming language that was developed by Guido Van Rossum at National Research Institute for mathamatics and computer Science in 1991.

Python is an object oriented high level programming language that is used to develop games, web application, etc. Its file extension is .py. Python is a loosely type it means you need not to declare datatype of variable

Features of Python :-

1. It is easy to use and learn because of its simplicity.
2. Python is an open source. It means python is freely available, you don't need to spent money on it.
3. It is plateform independent. It means you can run python on any operating system.
4. Program written in Python is Portable. It means program written on one machine can be run on another machine but (Python should installed on that machine) there should be python installed on another machine.

DATE: / /20
PAGE NO.

5. Python is a case-sensitive language. It means small letter variable and capital letter variable are two different things.
6. Python is object-oriented as well as procedure oriented.

variable:-

A variable is a name where the value of that variable is stored.

identifier:-

Identifier refers to name of variable, name of module etc.

Some rules for declaring a variable:-

1. You can use underscore during declaration of a variable
eg:- _a;
2. Keywords can not be used as variable name. eg:- if = 60; ✗
3. Name of variable should be start from an alphabet or underscore but not a number. eg:- a = 60; ✓ b = 40; ✓ 1num = 5; ✗
4. variable name are case-sensitive.

Data type:-

Data types defines a variable can store which type of value.

DATE: / /20
PAGE NO.

1. int
2. float
3. Complex
4. string
5. List
6. tuple
7. Dict
8. Boolean.

1. **int** :- int data type consist of numerical value without a decimal point.
   eg :- x=10;

2. **float** :- float data type consist of numerical value with a decimal point.
   eg :- x = 10.10;

3. **Complex** :- complex data type consist of imaginary part represented by 'j' eg :- y = 10j;

4. **String** :-
   String consist of number of character enclosed within double quotes.
   z = "welcome to python";

5. **dict** :- { }
   eg :- x = {"name": "abc", "age": 35}

6. **List** :- [ ]
   eg :- y = ["apple", "Mango", "Grapes"]

7. **tuple** :- ( )
   eg :- z = ("apple", "Mango", "grapes")

8. Boolean:- It consist of two values either True or False.
eg: d = True;

print() :-
Print statement is used to print some message or value.

→ To print some message :-

```
print ("Welcome to Zedking");
```

→ To print some value :-

```
x = 10;
print (x);
```

O/P → 10

→ To print some message & value :-

```
x = 10;
print ("value of x = ", x);
```

Features of Python :-

7. It's structure is similar like English. So it is easy to understand.

8. It has a features of both procedural oriented language as well as features of object oriented language.

9. It is compatible with c & c++.

10. It has its own library in which meaning of some predefined function is defined.

11. It has some keywords which are used for some specific purpose.

Application of Python

- Amazon uses to python to store the user bying habits.
- Facebook uses python to process the images.
- Google uses python to handle the search activity.
  ---- etc.

Some Rules for declaring a variable

- Special symbols are not allowed during declaration of a variable.
  eg:- $a = 60; ✗
- Variable name should be small or easily understandable.
- blank space are not allowed during declaration of a variable. eg:- ab = 60; ✗

Keywords :- if elif, True, False, None.

DATE: / /20
PAGE NO.

type():- This is used to check the datatype of a variable.

eg:- a = 60,
print (type(a));

- Data type :-

- dict :- dictonary consist of key value pair. Key must be unique and value may not be unique. It can be modify. So, It is mutable. To create dict we use curly braces {}.

eg:- f = {'name' : 'abc', 'age' : 30};
(key → 'name', value → 'abc'; key value pair)

- list :- list is a collection of value which may be of same type or may be of different type value is called the element of list. A list can be modify. So, It is mutable. To create a list we use square braces. []

eg:- y = ["Apple", 10];

- tuple :- tuple is a collection of values. This value is known as elements of tuple. It can not be modify. So, It is immutable. To create a tuple we

use paranthesis ().
eg:- x = ("Apple", "Banana");

- set :- A set is a data type which does not allow duplicate elements. The elements are placed in curly braces {}. It can be modified. It does not provide any order of insertion. It means we don't know in which order the elements of sets will be displayed. It does not provide indexing & slicing It provide dynamic size.
eg:-
{"a", "b", 1, 2}

→ pop() :- This is used to get or remove any randomly value from the set.
eg:- a = {1,2,3,4,5};
a.pop();
print(a);
o/p:- {2,3,4,5}.

→ remove() :-
This is used to remove particular value from the sets. If the value which you are removing does not exist in the set then it raises an error.
eg:- a = {1,2,3,4,5,6};
a.remove(4);
print(a); o/p:- {1,2,3,5,6}

→ discard():- discard is also used to remove the value from sets. If the value does not exist which you are removing from the set then it does not raise an error.

eg:-
```
x = {1,2,3,4};
x.discard(3);
print(x);
```
o/p:- {1,2,4}

→ add():- This is used to add any element into the set. Only single element is allowed to add at one time.

eg:-
```
x = {1,2,3,4};
x.add(5);
print(x);
```
o/p:- {1,2,3,4,5}

→ update():- Update is used to add number of elements at a time in the set.

eg:-
```
x = {1,2,3,4};
x.update([5,6]);
print(x);
```
o/p:- {1,2,3,4,5,6}

→ len():- This returns the number of elements present in the set.

eg:- x = {1,2,3,4}

print(len(x));
o/p :- 4

Identifier :-
Identifier reffers to the name of variable, module or function and so on.
eg :- x = 10;
↓
identifier

Literal :-
It reffers to the value or element asign to the variable. It can be numeric, string, complex and so on.
eg :- x = 10;
↓
literal

Type Casting :-
i) Implicit type casting :-
eg :- x = 10;
y = 20.20;
z = x+y;
print ('sum =', z);
print ('datatype of z =' type (z));
o/p :- Sum = 30.2
datatype of z = <class 'float'>

ii) Explicit type Casting :-

```
x = 10;
print ("value of x = ", x);
print ("datatype of x = ", type (x));

print ("value of x = ", str(x));
print ("data type of x = ", type(str(x)));
```

o/p:- value of x = 10
data type of x = <class 'int'>

value of x = 10
data type of x = <class 'string'>

clear():-

This function is used to clear all the elements of set.

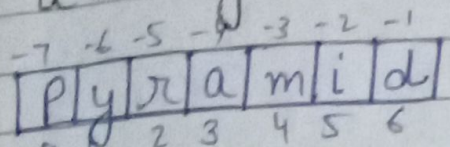
```
x = {1, 2, 3, 4, 5};
x.clear();
print(x);
```

o/p:- set()

string :- String consist number of characters enclosed within double quotes or single quotes.

```
x = "zedking";
y = 'zedking';
```

a = "Pyramid"



Slicing :- Slicing is a part of an input string. It means you can get any part of Input string. To do slicing we use [] bracket. In square bracket we pass three arguments beg index, end index & step size.

→ beg index :-

This index is used to set the index from where slicing is to be start. If you does not specify the beg index then by default slicing will be start from 0 index in case of +ve indexing.

→ end index :-

This index is used to specify the indexing where the slicing will be stop. It does not include the end index. It means end point of slicing will be end index -1. If you does not specify end index then slicing will be done up to the end index of input string.

beg index include [1:5:] end index exclude

→ step :- This is used to set the increment in index during slicing by default step size is 1 in case +ve indexing & -ve indexing increment -1.

- In case of +ve indexing :-

→ beg index will be 0. [by default]
→ step size will be 1.

- In case of -ve indexing :-

→ beg index is -1. [by default]
→ step size will be -1.

Note :-

Slicing will not generate array index out of bound error or string index out of range; if end index which you are mention in slicing exceeds the original end index of input string.

Example of Slicing :-

```
x = "Zedking";
print(x[::-1]);
```

DATE : / /20
PAGE NO.

```
y = "hartron";
print(y[1::]);
```

```
z = "Kaithal";
print (z[1:5:]);
```

```
a = "Karnal";
print(a[1:4:1]);
```

```
b = "Jasika";
print(b[1:4:2]);
```

```
c = "Kurukshetra";
print(c[::2]);
```

Indexing :-

In Indexing we retrieve a particular element or character from an input string by giving the index number in [] bracket. If we give index in square bracket that exceeds the original index of input string then it raises an error string index out of range.

eg:-

```
x = "zedking";
print(x[1]);
```

Operation on String :-

1. Concatenation :-

This operation is used to concatenate two string we use '+' symbol to concatenate two strings

eg :- x = "zedking";
y = "hartron";
print (x+y);

o/p :- Zedkinghartron.

⇒ x = "zedking";
y = 100;
print (x+y); o/p :- error.

can't convert 'int' object to str implicitly

x = "zedking";
y = 100;
print (x + str(y));

o/p :- Zedking100.

x = "Kaithal";
print (x * 4);
o/p :- KaithalKaithalKaithalKaithal

x = "Kaithal";
print (x * "Kaithal");

o/p :- error.

Note:- 1. string or string dono string value multiply nhi hogi.
2. ek string or dusri int value we dono value multiply hogi.

format() :-

This method is used to merge no of string into a single object The merge text will be displayed where the placeholder {} are.

Example :-

1. str1 = " welcome to {name}";
print (str1.format(name = "zedking"));

o/p :- welcome to zedking.

2. str1 = " welcome to {name}. It is placed near the {place}";
print (str1.format(name = "zedking", place = "Hanuman Vatika"));

o/p :- welcome to zedking. It is placed near the Hanuman Vatika.

Alternate method :-

3. str1 = " welcome to {0}. It is placed near the {1}";
print (str1.format("zedking", "Hanuman Vatika"));

DATE: / /20
PAGE NO.

Print Statement :-

→ Print ("1", "2", "3", "4")

o/p → 1 2 3 4

→ print ("1", "2", "3", "4", sep = "-");

o/p → 1-2-3-4

No. ke bich me symbol ko add krne ke liye hm sep ka use krte h koi bhi symbol le skte h. @, $ ---

→ print ("welcome to", end = "zking !!!");

o/p:- wlkm to zking!!!

Input():- This method is used to get the input from user during run time.

→ x = input ("enter value of x = ");
print (x);

→ x = input ("enter value of x = ");
print ("value you enter is = ", x);
print ("data type of x is = ", type(x));

o/p:- value you enter is = 60
data type of x is = <class 'str'>

- Jo input de rhe h ve as a string leta h.
- console pe jo input dete h ve as a string leta h.
- explicit convert Ke liye →

```
x = int(input("enter value of x = "));
print("value you enter is = ", x);
print("data type of x is = ", type(x));
```

o/p :- value you enter is = 60
data type of x is = <class 'int'>

* Operator in Python :- Same as Java Script

i) Arithmatic Operator
ii) Comparison Operator
iii) Bitwise Operator
iv) Assignment Operator
v) Logical Operator

i) Arithmatic Operator :-

This operator are used to perform Arithmatic operations like addition, Multiply, division, and so on.

operator	Example	Description
/	a/b;	divide a/b. gives result in fraction. (result come with decimal)

* *	a * * b;	It gives exponent eg:- a = 3 , b = 3 c = a * * b = $(3)^3$ = 27
// (integer fraction)	a // b	divide a & b and gives result as integer. eg:- 5//2 = 2 int value.

ii) Comparison Operator & Relational :-

This operator are used to compare value of variable
Same as Java script.

iii) Logical Operator :-

		symbol
• and operator	→	and
• or operator	→	or
• not operator	→	not

eg:- a = 60;
b = 40;
c = 20;

if a > b and a > c :
print ("A is largest");

iv) Bitwise operator:

- bitwise And → &
- bitwise OR → |
- bitwise XOR → ^
- bitwise not → ~ [one's compliment]
- bitwise left shift → <<
- bitwise right shift → >>

→ bitwise And :-

Symbol :- &

```
x = 5;
y = 3;
print(x&y);
```

5 → 0 1 0 1
3 → 0 0 1 1
x&y 0 0 0 1 → (1)

o/p → 1.

→ bitwise OR :-

Symbol :- |

```
x = 5;
y = 3;
print(x|y);
```

5 → 0 1 0 1
3 → 0 0 1 1
x|y 0 1 1 1 → (7)

o/p :- 7.

→ bitwise XOR :-

Symbol :- ^

```
x = 5;
y = 3;
```

print (x ^ y);

o/p :- 6.

5 → 0 1 0 1
3 → 0 0 1 1
x^y 0 1 1 0

→ one's complement :-

Symbol - ~

(~x) = -(x+1) (one's complement)

eg :- x = 5;
print (~x);

o/p :- -6

[.. -(x+1) = -(5+1)
= -6

→ bitwise left shift :-

Symbol :- <<

eg :-
x = 5;
print (x << 1);

o/p :- 10

5 = 0 1 0 1
1 0 1 0
5<<1 = ↑
= 10

→ bitwise right shift :-

Symbol :- >>

eg :- x = 5
print (x >> 1)

o/p :- 2

5 = 0 1 0 1
0 0 1 0
5>>1 = (2)

print (x ^ y);

o/p :- 6.

5 → 0 1 0 1
3 → 0 0 1 1
x^y 0 1 1 0

→ one's complement :-

Symbol - ~

one's complement
(~x) = -(x+1)

eg :- x = 5;
print (~x);
o/p :- -6

[∵ -(x+1) = -(5+1)
= -6]

→ bitwise left shift :-

Symbol :- <<

eg :-
x = 5;
print (x << 1);
o/p :- 10

5 = 0 1 0 1
1 0 1 0
5<<1 = ↑
= 10

→ bitwise right shift :-

Symbol :- >>

eg :- x = 5
print (x >> 1)
o/p :- 2

5 = 0 1 0 1
0 0 1 0
5>>1 = (2)

WAP to add two numbers.

```
a = int(input(" Enter value of a = "));
b = int(input(" Enter value of b = "));
s = a+b;
print (" Sum = ", s);
```

WAP to find simple Interest.

```
p = int(input(" enter principal = "));
r = int(input(" enter rate of interest = "));
t = int(input(" enter time period = "));

si = (p*r*t)/100;
amt = p+si;
print (" simple interest = ", si);
print (" Amount = ", amt);
```

WAP to find perimeter and area of rectangle.

```
l = int(input(" enter length = "));
b = int(input(" enter breadth = "));
per = 2*(l+b);
area = l*b;
print (" Perimeter = ", per);
print (" Area = ", area);
```

Area and Perimeter of Square

```
a = int(input("enter side of square="));
per = 4*a;
area = a*a;
print("perimeter of square = ", per);
print("area = ", area);
```

circumfrence and area of circle

```
r = int(input("Enter radius of circle="));
area = 22/7 * r * r;
circum = 2 * 22/7 * r;
print("area of circle is = ", area);
print("circumfrence = ", circum);
```

volume of cube

```
a = int(input("Enter edge of cube ="));
vol = a**3; [or a*a*a]
print("volume of cube is = ", vol);
```

volume, LSA and TSA of cuboid

```
w = int(input("Enter width of cuboid="));
h = int(input("Enter height of cuboid ="));
l = int(input("Enter length of cuboid ="));
vol = w*h*l;
```

```
LSA = 2*h*(l+w);
TSA = 2*(l*w + w*h + l*h);
print ("LSA of cuboid = ", LSA);
print ("TSA of cuboid = ", TSA);
```

Volume, LSA and TSA of Cylinder

```
r = int(input("Enter radius of cylinder="));
h = int(input("Enter height of cylinder ="));

Vol = 22/7 * r * r * h;
LSA = 2* 22/7 * r * h;
TSA = 2* 22/7 * r * (r + h);

print ("volume of cylinder = ", Vol);
print ("LSA of cylinder = ", LSA);
print ("TSA of cylinder = ", TSA);
```

Control Statement :-

Control statement is used to control the flow of execution of the program. It means which part of program should be executed and how it is executed or which part of program should not be executed depends upon control statement.

1. Conditional Statement :-

In this statement which part of program should be

executed depends upon some condition.
eg:- if else.
if elif else

WAP to check wheter a no. is even or odd.

```
num = int(input("enter number ="));
if num % 2 == 0:
    print("Number is even");
else:
    print("Number is odd");
```

WAP to check wether a no is positive, negative or zero.

```
num = int(input("enter number ="));
if num < 0:
    print("Number is Negative");
elif num > 0:
    print("Number is Positive");
else:
    print("Number is zero");
```

WAP to find larger of two number

```
a = int(input("enter value of a ="));
b = int(input("enter value of b ="));

if a > b:
    print("A is larger");
```

```
else:
    print ("B is larger");
```

WAP to find larger of three numbers

```
a = int(input("enter value of a = "));
b = int(input("enter value of b = "));
c = int(input("enter value of c = "));

if a>b and a>c:
    print ("A is largest");

elif b>a and b>c:
    print ("B is largest");

else:
    print ("C is largest");
```

WAP to find Grade

```
hin = int(input("Enter Marks in hindi = "));
eng = int(input("Enter Marks in English = "));
math = int(input("Enter Marks in math = "));

per = ((hin+eng+math)/300)*100;

if per >= 90:
    print("A+ grade");
elif per >=80 and per <90:
    print ("A grade");
```

```
elif per >= 70 and per < 80:
    print (" B+ grade ");
elif per >= 60 and per < 70:
    print (" B grade ");
elif per >= 50 and per < 60:
    print (" C+ grade ");
elif per >= 40 and per < 50:
    print (" C grade ");
else:
    print (" Fail ");
```

WAP to calculate electricity Bill payment.

```
nr = int(input(" Enter new reading = "));
oldr = int(input(" Enter old reading = "));

uc = nr - oldr;

if uc <= 150:
    amt = uc * 3;
elif uc > 150 and uc <= 250:
    amt = 150 * 3 + (uc - 150) * 4;
elif uc > 250 and uc <= 500:
    amt = 150 * 3 + 100 * 4 + (uc - 250) * 5;
elif uc > 500 and €uc <= 800:
    amt = 150 * 3 + 100 * 4 + 250 * 5 +
                (uc - 500) * 6;
elif uc > 800:
    amt = uc * 7;
print (" amount to be paid is = ", amt);
```

WAP to find a year is a leap year or not.

```
year = int(input ("enter year ="));
if year % 400 == 0:
        print (" year is a leap year");
elif year % 100 == 0:
        print (" year is not a leap year");
elif year % 4 == 0:
        print ("year is a leap year");
else:
        print ("year is not a leap year");
```

Loop :- Loop is used to perform repitition. If you want to execute a statement no. of times then you can place that statement within the loop.

i) while loop :-

→
```
i = 1;
while  i <= 5:
        print (i);
        i = i + 1;
```

→ WAP to print table of a number.

```
n = int(input("enter number ="));
i = 1;
while i <= 10:
        print(n*i);
        i = i+1;
```

→ WAP to find factorial of a no.

```
n = int(input("enter number = "));
f = 1;
i = n;
while i >= 1:
        f = f * i;
        i = i-1;
print("factorial is =", f);
```

→ WAP to find Power of a number.

```
a = int(input("enter value of a = "));
b = int(input("enter value of b = "));

p = 1;
i = 1;
while i <= b:
        p = p * a;
        i = i+1;

print("power is =", p);
```

→ Find sum and average of first 10 natural numbers.

```
s = 0;
i = 1;
while i <= 10:
        s = s+i;
        i = i+1;

a = s/10;
print(" Sum of first 10 natural number
        is = ", s);
print(" average of first 10 natural
        number is = ", a);
```

→ Find sum and average of first 'n' natural number.

```
n = int(input(" enter value of n = "));
 s = 0;
 i = 1;
 while i <= n:
         s = s+ i;
          i = i+1;
 a = s/n;
 print (" sum is = ", s);
 print (" Average is = ", a);
```

→ Find sum of even number b/w first 'n' natural number.

```
n= int(input("enter value of n = "));
s=0;
i=1;
while i<=n:
        if i%2==0:
            s=s+i;
        i=i+1;
print("Sum is =", s);
```

Find sum and average of even number between first 'n' natural number.

```
n= int(input("enter value of n = "));
S= 0;
c=0;
i=1;
while i<=n:                    for odd !=
        if i%2==0:
            s=s+i;
            c=c+1;
        i= i+1;
a= s/c;
print("Sum is =", s);
print("Average is =", a);
```

Find sum of all digits of a number.

```
num = int(input("enter value of n="))
S= 0;
```

```
n= num
while n>0:
        r= n%10;
        s = s+r;
        n = n//10; → also n= int (n/10);
print ("sum is = ", s);
```

Find reverse of a number.

```
num = int (input ("enter value of n="));
rev = 0;
n= num;
while n>0:
        rem = n%10;
        rev = rev*10 + rem;
        n = n//10;
print ("Reverse is = ", rev);
```

WAP to check wheter a no. is palindrom or not.

```
num = int (input ("enter value of n="));
rev = 0;
n = num;
while n>0:
        rem = n%10;
        rev = rev*10 + rem;
        n = n//10;
if num == rev:
        print ("Number is Palindrom");
```

```
else:
    print ("Number is not Palindrom");
```

WAP to check wheter a no. is armstrong or not.

```
num = int(input("enter value of n="));
s = 0;
n = num;
while n>0:
        r = n%10;
        s = s+r*r*r;
        n = n//10;

if num==s:
        print("Number is Armstrong");
else:
        print("Number is not Armstrong");
```

For Loop:-

i)

→
```
fruits = ["Apple", "Banana", "Orange", "Kiwi"]
for i in fruits:
        print(i);
```

→
```
food = ("Maggie", "Spring Rolls", "TiKKi");
for i in food:
        print(i);
```

→ car = {"duster", "verna", "swift"};

```
for i in car:
        print(i);
```

ii) String:-

→ n = "zedking";

```
for i in n:
        print(i);
```

iii) range() :- range contain three arguments

- start :- from where you start the loop.
- end :- where you want to end the loop.
  The end point will not be included.
- step :- step is the value for increment of the counter. By default the increment is 1.

eg:-

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

eg:-

```
for i in range(1,5):
        print(i);
```

WAP to print table of a number.

```
n = int(input("Enter number = "));
for i in range(1,11,1):
        print(n*i);
```

WAP to print counting from 10 to 1.

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

WAP to find factorial of a number.

```
n = int(input (" Enter number = "));

f = 1;
for i in range (n, 0, -1):
        f = f * i;
print (" factorial is = ", f);
```

find sum and average of first 10 natural number.

```
S = 0;
for i in range ( 1, 11, 1):
        S = S + i;
a = S/10;
print(" Sum is = ", S);
print(" average is = ", a);
```

find sum and average of first n natural number.

```
n = int(input(" Enter value of n = "));
S = 0;
for i in range (1, n+1, 1);
        S = S + i;
```

```
a = s/n;
print("Sum is = ", s);
print("average is = ", a);
```

find sum and average of even no b/w first n natural number.

```
n = int(input("Enter value of n = "));
s = 0;
c = 0;
for i in range (1, n+1, 1):
                    → (!= odd)
        if i%2 == 0:
            s = s+i;
            c = c+1;
a = s/c;
print("Sum is = ", s);
print("average is = ", a);
```

break statement :-

break statement is used to move out from loop.

```
for i in range (1, 11):
        print(i);
        if i == 5:
            break;
if i == 5:
        print ("exit from for loop");
```

Continue Statement :-

```
for i in range (1,11):
        if i == 5:
            continue;
        print (i);
```

pass statement :-

pass is used to define an empty loop.

eg :-

```
for i in range (1,11):
        pass;
    print (i);
```

o/p :- 10

- Pattern Based Questions

1.
```
*
* *
* * *
* * * *
```

```
i = 1;
while i <= 4:
        j = 1;
        while j <= i:
            print ("*", end = " ");
            j = j+1;
        print();
        i = i+1;
```

DATE: / /20
PAGE NO.

Alternative Method :-

```
for i in range(1,5,1):
        for j in range(1,i+1,1):
            print("*",end=" ");
        print();
```

2.
```
* * * *
* * *
* *
*
```

```
i = 4;
while i >= 1:
        j = 1;
        while j <= i:
            print("*",end=" ");
            j = j+1;
        print();
        i = i-1;
```

Alternative Method :-

```
for i in range(4,0,-1):
        for j in range(1,i+1,1):
            print("*",end=" ");
        print();
```

3.
```
      *
    * *
  * * *
* * * *
```

```
i = 1;
while i <= 4:
    j = 1;
    while j <= 4-i:
        print(" ", end=" ");
        j = j+1;
    K = 1;
    while K <= i:
        print("*", end=" ");
        K = K+1;
    print();
    i = i+1;
```

Alternative Method :-

```
for i in range(1, 5, 1):
    for j in range(1, 4-i+1, 1):
        print(" ", end=" ");
    for K in range(1, i+1, 1):
        print("*", end=" ");
    print();
```

4.

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

```
i = 1;
while i <= 4:
    j = 1;
    while j <= 4-i:
```

```
            print (" ", end = " ");
            j = j+1;
    K = 1;
    while K <= 2*i-1:
            print ("*", end = " ");
            K = K+1;
    print();
    i = i+1;
```

Alternative Method :-

```
for i in range (1,5,1):
        for j in range (1,4-i+1,1):
            print (" ", end = " ");
        for K in range (1,2*i-1+1,1):
            print ("*", end = " ");
        print();
```

5.
```
1
2 2
3 3 3
4 4 4 4
```

```
i = 1;
while i <= 4:
        j = 1;
        while j <= i:
            print (i, end = " ");
            j = j+1;
        print();
        i = i+1;
```

Alternative Method :-

```
for i in range (1, 5, 1):
        for j in range (1, i+1, 1):
            print (i, end = " ");
        print ();
```

6.
```
1
1 2
1 2 3
1 2 3 4
```

```
i = 1;
while i <= 4:
        j = 1;
        while j <= i:
            print (j, end = " ");
            j = j+1;
        print ();
        i = i+1;
```

Alternative Method :-

```
for i in range (1, 5, 1):
        for j in range (1, i+1, 1):
            print (j, end = " ");
        print ();
```

7. 4 4 4 4
3 3 3
2 2
1

```
i = 4;
while i >= 1:
    j = 1;
    while j <= i:
        print(i, end = " ");
        j = j+1;
    print();
    i = i-1;
```

Alternative Method :-

```
for i in range(4, 0, -1):
    for j in range(1, i+1, 1):
        print(i, end = " ");
    print();
```

8. 1 2 3 4
1 2 3
1 2
1

```
i = 4;
while i >= 1:
    j = 1:
    while j <= i:
```

```
            print(j, end = " ");
            j = j+1;
        print();
        i = i-1;
```

Alternative Method :-

```
for i in range(4, 0, -1):
        for j in range(1, i+1, 1):
            print(j, end = " ");
        print();
```

9. 4 3 2 1
   3 2 1
   2 1
   1

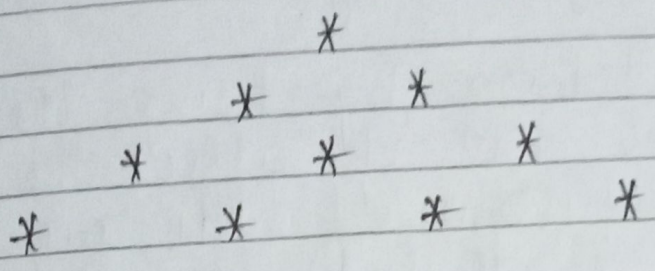
```
i = 4;
while i >= 1:
        j = i;
        while j >= 1:
            print(j, end = " ");
            j = j-1;
        print();
        i = i-1;
```

Alternative Method :-

```
for i in range(4, 0, -1):
```

```
for j in range (i, 0, -1):
        print (j, end = " ");
print();
```

10.



```
i = 1;
while i <= 4:
        j = 1;
        while j <= 4-i:
                print (" ", end = " ");
                j = j+1;
        K = 1;
        while K <= i:
                print ("*", " ", end = " ");
                K = K+1;
        print ();
        i = i+1;
```

Alternative Method :-

```
for i in range (1, 5, 1):
        for j in range (1, 4-i+1, 1):
                print (" ", end = " ");
        for K in range (1, i+1, 1):
                print ("*", " ", end = " ");
        print();
```

DATE: / /20
PAGE NO.

Assert :- Assert is a debugging tool.

⇒ debugging :-
debugging is a process of finding debugg and remove them.

⇒ We can debugg by using the print() statement but it affect the performance of the system. If you are using print() statement for debugging than you have to delete print() statement which you are using debugging. After debugging if you don't delete it affect the system performance.

1) Assert Condition
2) Assert Condition . error - message .

Condition :- Condition is false than execution of program is stop and it gives assertion error.

Error - Message :-
It is optional the error message is the message which you want to display when condition is false

Example of Condition :

eg: a = 10;
b = 0;
c = a/b;
print (c);
assert b! = 0.

o/p :- Assertion error.

. Example of error - message:

a = 10:
b = 0:

assert b! = 0, "B can not be zero"

→ If you are not deffining an error message in assert statement than it will give only assertion error.

→ If you are deffining error message than it will give Assertion error with error message which you are defining.

List :- A list is a sequencial data type which consist of elements are in ordered. The elements are written in square bracket [ ] with comma (,) seperator.

→ In list insertion order is preserved.
→ Element of list are mutable. It means it can be modified.
→ Indexing is used in list.
→ duplicate values or elements are allowed.
x = [1, 2, 3, 2, 4, 3] → 2&3 are duplicate values
→ to differentiate duplicate values indexing is used.
→ List provide dynamic size. It means we can increase or decrease the size of list at run time according to our requirement.
→ A list can consist of hetrogeneous elements.

· hytrogeneous :- diff. type ki value store kr skte h. like; String, numeric . . .

1. How to create an empty list.

```
x = [].
print (x);
print ("datatype of x =", type (x));
```

o/p :- []

2. How to create a list with single element.

```
x = ["welcome to zedking"];
print (x);
```

3. How to create a list with multiple elements.

```
x = ["hartron", "zedking", "spoken",
     "coaching"];
print (x);
```

4. How to create a list with different type of elements.

```
x = ["hartron", 2, "spoken", 4];
print (x);
```

5. How to create a list with duplicate elements.

```
x = [1, 2, 3, 4, 2, 4];
print (x);
```

6. How to create a multi-dimensional list.

```
x = [1, 2, [3, 4]];
print (x);
```

7. How to create a list using list() function.

→ x = "zedking";
print(list(x));

→ a = (1, 2, 3, 4);
print(list(a));

Multi-dimensional list:

x = [1, 2, [3, 4], 5];
0 1 2 3

print(x[2][0]); o/p :- 3.

8. Create a list using split() method.

- Split():

Syntax:- list.split(Seperator, max-split)

→ Seperator:- It is a delimeter used to specify from where you want to split the list. If you does not specify the seperator, then by default it will seperate the list from white space (Space, New line).

→ max-split :- This parameter is used to specify the maximum number of split which you want to do. By default value of max-split is -1. It means there is no limit.

Eg:

1. x = "welcome, to, zedking";
print ("before split value of x = ");
print(x);
print ("after split value of x = ");
print (x.split());

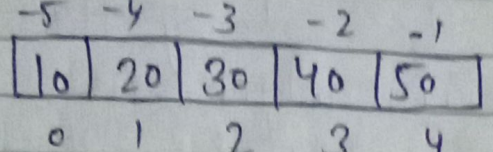
2. x = "welcome, to, zedking, group of institution";
print("before split value of x = ", x);
print("after split value of x = ");
print (x.split(", "));

3. x = "welcome, to, zedking, group, of, institution";
print ("before split value of x = ", x);
print ("after split value of x = ");
print (x.split(", ", 2));

4. x = "1:2:3:4:5:6:7:8:9";
print ("before split value of x = ", x);
print ("after split value of x = ");
print (x.split(":"));

→ (":", 2)

How to access element of a list.



by using indexing we can access element of a list.

eg:- +ve indexing:

```
x = [10, 20, 30, 40, 50];
print ("element of list = ");
print (x);
print ("value at x[0] = ", x[0]);
print ("value at x[1] = ", x[1]);
print ("value at x[2] = ", x[2]);
print ("value at x[3] = ", x[3]);
print ("value at x[4] = ", x[4]);
```

-ve indexing:

```
x = [10, 20, 30];
print ("element of list - ");
print (x);
print ("value at x[-1] = ", x[-1]);
print ("value at x[-2] = ", x[-2]);
print ("value at x[-3] = ", x[-3]);
```

How to add element into the list.

1. append()
2. insert()
3. extend()

1. append() :- This method is used to add a new element into the list. The new element will be stored at the last position of the list. append will take one parameter that is the element which you want to append.

eg:-

```
x= [];
print (x);
x. append (10);
x. append (20);
x. append (30);
x. append (40);
print ("List after appending =", x);
```

2. insert() :- This method is used to insert a new element at the specified position. insert will take two parameter. one specified the index where you want to add new element and another specify the value (element) which you want to add into the list.

DATE: / /20
PAGE NO.

eg:- x.insert (2,50);

→ x = [10, 20, 30, 40];
print ("List before inserting =", x);
x.insert (2,50);
print ("List after inserting =", x);

→ x = ["apple", "mango", "banana"];
print ("List before inserting =", x);
x.insert (2, "grapes");
print ("List after inserting =", x);

3. extend():- This method is used to insert number of element into the list. It will take a single parameter. The insertion of new element will take place at the end of list.

eg:- x = ["apple", "mango", "banana"];
print ("List before extending =", x);
x.extend (["grapes", "kiwi", "orange"]
print ("List after extending =", x);

How to remove the element from the list.

1. remove():- This method is used to remove a specific value from the list. This will take one parameter, that is the value which you want to remove from

the list.

eg:- x.remove(50);

If the value which you are removing from the list does not exist then it raises an error. It does not return any value.

eg:-
x = [10, 20, 30, 40, 50];
print("value of x before removing=", x);
x.remove(30);
print("value of x after removing=", x);

→ 80 [error]

2. pop() :- This will remove the last element value from the list. It return a value that is the value which you are removing from the list. when list is empty then pop() method raises an error.

eg:-
x = [10, 20, 30, 40, 50];
print("value of x before removing=", x);
print("value of x after removing=", x);
print("removed value is =", x.pop());

DATE: / /20
PAGE NO.

Operation on List :-

1. Concatenation :- This is used to concatenate two list. We use + symbol to concatenate two list.

eg:-

```
x = [10, 20, 30, 40, 50];
y = [60, 70, 80, 90];
print ("value of x =", x);
print ("value of y =", y);
print ("value after concatenation =", x+y);
```

2. Multiply :-

```
x = [10, 20, 30, 40, 50];
print ("value of x =", x);
print ("value after multiply =", x*3);
```

3. Reverse :-

This function is used to reverse the element of a list. This function does not return any value.

eg:-

```
x = [10, 20, 30, 40];
print ("value of x =", x);
x.reverse();
print ("value of x after reverse =", x);
```

eg:
```
x = ["apple", "banana", "mango"];
print ("value of x =", x);
x.reverse();
print ("value of x after reverse =", x);
```

- Reverse the element of list using slice.

```
x = ["apple", "banana", "mango"];
print ("value of x =", x);
x = x[::-1];
print ("value of x after reverse =", x);
```

- How to access individual element of a list.

→ reversed():-

This function is used to access the individual element of a list in reverse order. It returns iterator which is used to reverse the element of a list.

```
x = [10, 20, 30, 40];
print ("value of x =", x);
print ("value of x after reverse =",
                    list(reversed(x)));
```

→ sort():-

This function is used to sort the element of list either in ascending order or in descending

order. This function does not return any value. By default element of list will be sorted in ascending order if the element of list is numeric. If the element of list is string then it will be arranged in alphabatically order.

Parameter of sort():-

- reverse:- If the value of reverse is True then it will sort the element of list in descending order.
- Key:- function key uses for sort comparison

→ Arrange the element of list in ascending order.

- x = [20, 60, 10, 40];
print("value of x before sorting=", x);
x.sort();
print("value of x after sorting=", x);
- x = ["apple", "grapes", "orange", "banana"]
print("value of x before sorting=", x);
x.sort();
print("value of x after sorting=", x);

To sort the element of list in descending order.

- x = ["banana", "mango", "apple", "grapes"];
  print("value of x before sorting =", x);
  x.sort(reverse = True);
  print("value of x after sorting =", x);

- x = [10, 60, 80, 20]
  print("value of x before sorting =", x);
  x.sort(reverse = True);
  print("value of x after sorting =", x);

Note:- If the element of list is numeric and string both then sort function raises an error.

eg:-
```
x = [10, 60, 80, "abc"];
print(x);
```

→ sorted():- This function is used to sort the element of list either in ascending order or descending order. This function returns a value. It returns the element of list in sorted order. It takes one parameter. It takes the list as parameter which is to be sorted.

Parameter:-

- iterable :- It is the sequence which is to be sorted for.
eg:- list.

- reverse:- It is used to sort the element of list in descending order.
eg:- sorted(x, reverse = True);

- Key:- function Key Specify, used for sort comparison.
eg:-

```
x = [70, 60, 80, 20];
print ("value of x before sorting =", x);
print ("value of x after sorting =",
                    list(sorted(x)));
```

→ In descending order:-

```
x = [40, 60, 80, 20];
print ("value of x before sorting =", x);
print ("value of x after sorting =",
                    list(sorted(x, reverse=
                    True)));
```