Index Page:

PRACTICAL 1: BASICS

➢ Executing simple statements like expression statements (numeric and Boolean type)

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 2+18
20
>>> 56-30
26
>>> 35/7
5.0
>>> 5*6
30
>>> 2**9
512
>>> x=10
>>> y="10"
>>> x==y
False
>>> x!=y
True
```

```
>>> z=50
>>> x>z
False
>>> x<z
True
>>>
```

➢ OPERATOR PRECEDENCE AND ASSOCIATIVITY

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 23-45+32
10
>>> 23*34/3
260.6666666666667
>>> 34**5/2
22717712.0
>>> 23>34 and 23>22 or 34>64
False
>>> 23<35 and 34>53 or 34>43
False
>>> 56<34 and 74>45 or 34<34
False
>>> (56*67)/6786
0.5529030356616563
>>> 76/68*56
62.58823529411765
```

➢ ASSERT STATEMENTS

```
IDLE Shell 3.11.0
File Edit Shell Debug Options Window Help
    Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
    on win32
    Type "help", "copyright", "credits" or "license()" for more information.
>>> #assert
>>> #if condition returns True,then nothing happens:
>>> x="hello"
>>> assert x=="hello"
>>> #if condition returns False, AssertionError is raised:
>>> assert x=="goodbye"
    Traceback (most recent call last):
      File "<pyshell#5>", line 1, in <module>
        assert x=="goodbye"
    AssertionError
>>>
```

```
IDLE Shell 3.11.0
File Edit Shell Debug Options Window Help
    Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
    on win32
    Type "help", "copyright", "credits" or "license()" for more information.
>>> no=10
>>> no+=3
>>> no
    13
>>> no-=5
>>> no
    8
>>> no*=5
>>> no
    40
>>> no/=2
>>> no
    20.0
>>>
```

- **ASSIGNMENT STATEMENT**

- **DELETE STATEMENT**

IDLE Shell 3.11.0

File Edit Shell Debug Options Window Help

```
    Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
    on win32
    Type "help", "copyright", "credits" or "license()" for more information.
>>> a=50
>>> b=70
>>> c=10
>>> a,b,c
    (50, 70, 10)
>>> del a
>>> a
    Traceback (most recent call last):
      File "<pyshell#5>", line 1, in <module>
        a
    NameError: name 'a' is not defined
>>> b
    70
>>> c
    10
>>>
```

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> x="hello world"
>>> print(x)
hello world
>>> y="abc"
>>> print('hello',y)
hello abc
>>>
```

➢ PRINT FUNCTION

1. Write a python program to declare three variables with values (i.e, principle,Rate of interest and No of years) and display the simple interest amount(Script mode).

```
#Program to implement simple interest calculator.
p=int(input("Enter the principle Amount:"))
n=int(input("Enter the No. of years:"))
r=int(input("Enter the Rate of interest:"))
SI=p*n*r/100
print("Simple Interest is",SI)
```

2. Write a python program to declare a variable “Name check” assign a string value to it.Demonstrate the use of assert statements to check the correct name stored in variable.

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> name='Saif'
>>> assert (name=='Saif'),'invalid name'
>>> #nothing happens as condition is true
>>> assert (name=='Yadav'),'invalid name'
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
    assert (name=='Yadav'),'invalid name'
AssertionError: invalid name
```

3. Demonstrate working of del()

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> a=['apple','banana','pear','kiwi']
>>> a
['apple', 'banana', 'pear', 'kiwi']
>>> del a[2]
>>> a
['apple', 'banana', 'kiwi']
>>>
```

4. Write a python program to display the value stored in variables (Names,Class,Address,Date of Birth).

```
fname="Mary"
lname="Poppins"
course="FYBSC-CS"
sem=1
address=="Mayfair apt,Bandra(west)"
dob="12/12/2005"
hsc_percent = 83.50

print("*****STUDENTS DETAILS*****")
print("Full name: {} {}".format(fname,lname))
print("Course/Sem: {}/ {}".format(course,sem))
print("Address:",address)
print("Date of Birth:",dob)
print("HSC %: ",hsc_percent)
```

```
*****STUDENTS DETAILS*****
Full name: Mary Poppins
Course/Sem: FYBSC-CS/ 1
Address: Mayfair apt,Bandra(west)
Date of Birth: 12/12/2005
HSC %:  83.5
```

INPUT FUNCTION

1. Write a python program to take input form user and perform basic arithmetic operations.

```python
x=int(input("Enter a number:"))
y=int(input("Enter a number:"))
add=x+y
sub=x-y
div=x/y
mul=x*y
print("Addition is:",add,"\n Subtraction is:",sub,"\n Division is:",div,
      "\n Multiplication is:",mul)
```

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================= RESTART: C:/Saif works/add and sub.py ================
Enter a number:12
Enter a number:6
Addition is: 18
 Subtraction is: 6
 Division is: 2.0
 Multiplication is: 72
```

2. Write a python program to take the input (i.e, radius of a circle) from the user and display the area of a circle and circumference of a circle.

```python
print("$$$$$Area and Circumference of circle$$$$$")
radius=float(input("Enter Radius:"))
area=3.14*radius**2
circumference=2*3.14*radius
print("Area of a circle:",area)
print("Circumference of a circle",circumference)
```

```
>>>
================= RESTART: C:/Saif works/add and sub.py ================
$$$$$Area and Circumference of circle$$$$$
Enter Radius:5
Area of a circle: 78.5
Circumference of a circle 31.400000000000002
>>>
```

3. Write a python program to take a input (i.e. length and breadth) from the user and display the area of a rectangle.

```
l=int(input("Enter length of rectangle:"))
b=int(input("Enter breadth of rectangle:"))
area=l*b
print("Area of a rectangle",area)
```

```
>>>
    ================= RESTART: C:/Saif works/add and sub.py ================
    Enter length of rectangle:10
    Enter breadth of rectangle:3
    Area of a rectangle 30
>>>
```

4. Write a python program to take the input (i.e. table of) from the user and display multiplication table of it.

```
print("#####Multiplication Table#####")
n=int(input("Enter a number:"))
print(n,'X 1 =',n*1)
print(n,'X 2 =',n*2)
print(n,'X 3 =',n*3)
print(n,'X 4 =',n*4)
print(n,'X 5 =',n*5)
print(n,'X 6 =',n*6)
print(n,'X 7 =',n*7)
print(n,'X 8 =',n*8)
print(n,'X 9 =',n*9)
print(n,'X 10 =',n*10)
```

```
    Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
    on win32
    Type "help", "copyright", "credits" or "license()" for more information.
>>>
    ================= RESTART: C:/Saif works/add and sub.py ================
    #####Multiplication Table#####
    Enter a number:12
    12 X 1 = 12
    12 X 2 = 24
    12 X 3 = 36
    12 X 4 = 48
    12 X 5 = 60
    12 X 6 = 72
    12 X 7 = 84
    12 X 8 = 96
    12 X 9 = 108
    12 X 10 = 120
```

Get Multiple inputs from a user in one line

5. Write a python program to take the input (i.e. Student name, Class, Roll no, Date of birth, Address, Contact no) from the user and display the ID card Details.

```
name,course=input("Enter name,course separated by space:").split()
print("Name,Course",name,course)
dob=input("Enter your date of birth:")
addr=input("Enter address:")
rollno=int(input("Enter rollno:"))
contact=int(input("Enter contact:"))
print("******ID CARD DETAILS******")
print("Roll no:",rollno)
print("Students name:",name)
print("Course:",course)
print("Date of birth:",dob)
print("Contact:",contact)
```

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================= RESTART: C:/Saif works/add and sub.py ================
Enter name,course separated by space:SAIF FYBSC-CS
Name,Course SAIF FYBSC-CS
Enter your date of birth:30/10/2005
Enter address:kurla(w)
Enter rollno:14
Enter contact:7453058211
******ID CARD DETAILS******
Roll no: 14
Students name: SAIF
Course: FYBSC-CS
Date of birth: 30/10/2005
Contact: 7453058211
```

PRACTICAL 2: CONDITIONAL STATEMENTS

1. Take a number from user and write a program to check whether a number is odd or even.

CODE AND OUTPUT:

```
num=int(input("Enter a number:"))
if num%2==0:
    print("The Number is Even")
else:
    print("The Number is Odd")
```

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============= RESTART: C:/Saif works/student report card.py ============
Enter a number:5
The Number is Odd
>>>
============= RESTART: C:/Saif works/student report card.py ============
Enter a number:6
The Number is Even
```

2. Take three inputs from user for three sides of triangle. Write a program to check whether triangle is equilateral, isosceles or scalene.

CODE AND OUTPUT:

```
s1=int(input("Enter a first side of triangle:"))
s2=int(input("Enter a second side of triangle:"))
s3=int(input("Enter a third side of triangle:"))
if s1==s2==s3:
    print("The Triangle is Equilateral")
elif s1==s2 or s2==s3 or s1==s3:
    print("The Triangle is Isosceles")
else:
    print("The Triangle is Scalene")
```

```
>>>
============= RESTART: C:/Saif works/student report card.py ============
Enter a first side of triangle:4
Enter a second side of triangle:4
Enter a third side of triangle:5
The Triangle is Isosceles
```

3. Generate a Student Report Card, by taking inputs from the student as follows: Check percentage with grade, Student first name and last name, Course, sem, roll no, and Marks of 5 Subjects. Calculate the following and provide the: i) Percentage

percent

>=80

>=70

>=60

>=50

>=40

<40 ii)

Grade

grades

O

A

B

C

D

F

iii) Status: PASS or FAIL.

<u>CODE AND OUTPUT:</u>

student report card.py - C:/Saif works/student report card.py (3.11.0)

File Edit Format Run Options Window Help

```
fname=input("Enter your first name:")
lname=input("Enter your last name:")
course=input("Enter your course:")
sem=int(input("Enter your semester:"))
rollno=int(input("Enter your rollno:"))
sub1=int(input("Enter your English marks:"))
sub2=int(input("Enter your Physics marks:"))
sub3=int(input("Enter your Chemistry marks:"))
sub4=int(input("Enter your Maths marks:"))
sub5=int(input("Enter your Biology marks:"))
percent=(sub1+sub2+sub3+sub4+sub5)/500*100
grade=''
if percent>=80:
    grade="O"
elif percent>=70 and percent<80:
    grade="A"
elif percent>=60 and percent<70:
    grade="B"
elif percent>=50 and percent<60:
```

```
elif percent>=40 and percent<50:
    grade="D"
else:
    grade="F"
status="FAIL"
if grade=="O" or grade=="A" or grade=="B" or grade=="C" or grade=="D":
    status="PASS"

print("******STUDENT REPORT CARD******")
print("Full Name: {} {}".format(fname,lname))
print("Course/Sem: {}/{}".format(course,sem))
print("Rollno:",rollno)
print("Percentage:",percent)
print("Status:",status)
```

IDLE Shell 3.11.0

File Edit Shell Debug Options Window Help

```
>>>
    ============= RESTART: C:/Saif works/student report card.py ============
    Enter your first name:Chaudhary Mohammad
    Enter your last name:Saif
    Enter your course:FYBSC-CS
    Enter your semester:1
    Enter your rollno:14
    Enter your English marks:76
    Enter your Physics marks:72
    Enter your Chemistry marks:67
    Enter your Maths marks:70
    Enter your Biology marks:78
    ******STUDENT REPORT CARD******
    Full Name: Chaudhary Mohammad Saif
    Course/Sem: FYBSC-CS/1
    Rollno: 14
    Percentage: 72.6
    Status: PASS
```

4. Write a python program to accept two numbers from the user and check if it is a positive or negative integer. If it is positive then compare the 2 numbers and print the result(greatest).

CODE AND OUTPUT:

```
*positive.py - C:/Saif works/positive.py (3.11.0)*
File Edit Format Run Options Window Help
a=int(input("Enter a number a:"))
b=int(input("Enter a number b:"))
if a>0 or b>0:
    print("The Number is Positive")
    if a>b:
        print("a is greater")
    else:
        print("b is greater")
else:
    print("The Number is negative")
```

```
IDLE Shell 3.11.0
File Edit Shell Debug Options Window Help
    Type "help", "copyright", "credits" or "license()" for more information.
>>>
    ================== RESTART: C:/Saif works/positive.py ==================
    Enter a number a:2
    Enter a number b:4
    The Number is Positive
    b is greater
>>>
    ================== RESTART: C:/Saif works/positive.py ==================
    Enter a number a:-1
    Enter a number b:3
    The Number is Positive
    b is greater
>>>
    ================== RESTART: C:/Saif works/positive.py ==================
    Enter a number a:-1
    Enter a number b:-3
    The Number is negative
```

5. Python Program to implement the Rock, Paper, Scissor game.

CODE AND OUTPUT:

```
p1=int(input("'1.ROCK, 2.PAPER, 3.SCISSOR'PLAYER 1 Enter your choice:"))
p2=int(input("'1.ROCK, 2.PAPER, 3.SCISSOR'PLAYER 2 Enter your choice:"))
if p1>3 or p2>3 or p1==0 or p2==0:
    print("Enter a valid option.")
else:
    if p1==p2:
        print("It's DRAW.")
    elif p1==1:
        if p2==2:
            print("PLAYER 2 WINS.")
        else:
            print("PLAYER 1 WINS.")
    elif p1==2:
        if p2==1:
            print("PLAYER 1 WINS.")
        else:
            print("PLAYER 2 WINS.")
    else:
        if p2==1:
            print("PLAYER 2 WINS.")
        else:
            print("PLAYER 1 WINS.")
```

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: C:/Saif works/positive.py ==================
'1.ROCK, 2.PAPER, 3.SCISSOR'PLAYER 1 Enter your choice:1
'1.ROCK, 2.PAPER, 3.SCISSOR'PLAYER 2 Enter your choice:2
PLAYER 2 WINS.
>>>
================== RESTART: C:/Saif works/positive.py ==================
'1.ROCK, 2.PAPER, 3.SCISSOR'PLAYER 1 Enter your choice:2
'1.ROCK, 2.PAPER, 3.SCISSOR'PLAYER 2 Enter your choice:3
PLAYER 2 WINS.
>>>
================== RESTART: C:/Saif works/positive.py ==================
'1.ROCK, 2.PAPER, 3.SCISSOR'PLAYER 1 Enter your choice:3
'1.ROCK, 2.PAPER, 3.SCISSOR'PLAYER 2 Enter your choice:1
PLAYER 2 WINS.
```

PRACTICAL 3: FOR STATEMENTS

1. Write a program to the sum of numbers from 1 to n where n is user input.

Code:

for loops.py - C:/Saif works/for loops.py (3.11.0)

File Edit Format Run Options Window Help

```
number=int(input("Enter the number:"))
sum=0
for i in range(1,number+1):
    sum=sum+i
print(sum)
```

Output:

IDLE Shell 3.11.0

File Edit Shell Debug Options Window Help

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======================== RESTART: C:/Saif works/for loops.py ========================
Enter the number:12
78
>>>
======================== RESTART: C:/Saif works/for loops.py ========================
Enter the number:4
10
>>>
```

2. Write a program to print the sum of series y=x^1+x^2+x^3+…x^n where x and n are user inputs.

Code:

for loops.py - C:/Saif works/for loops.py (3.11.0)

File Edit Format Run Options Window Help

```
x=int(input("Enter the base number:"))
n=int(input("Enter the power:"))
sum=0
for i in range(1,n+1):
    sum=sum+x**i
print("Sum of the series is",sum)
```

Output:

```
IDLE Shell 3.11.0
File Edit Shell Debug Options Window Help
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====================== RESTART: C:/Saif works/for loops.py ======================
Enter the base number:2
Enter the power:3
Sum of the series is 14
>>>
====================== RESTART: C:/Saif works/for loops.py ======================
Enter the base number:3
Enter the power:4
Sum of the series is 120
>>>
```

3. Write a program to accept a number and check whether it is prime.

Code:

```
for loops.py - C:/Saif works/for loops.py (3.11.0)
File Edit Format Run Options Window Help
n=int(input("Enter the number:"))
for i in range(2,n):
    if n%i==0:
        print("The number is not a prime number")
        break
else:
    print("The number is a prime number")
```

Output:

```
IDLE Shell 3.11.0
File Edit Shell Debug Options Window Help
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====================== RESTART: C:/Saif works/for loops.py ======================
Enter the number:2
The number is a prime number
>>>
====================== RESTART: C:/Saif works/for loops.py ======================
Enter the number:4
The number is not a prime number
>>>
====================== RESTART: C:/Saif works/for loops.py ======================
Enter the number:5
The number is a prime number
>>>
```

4. Write a program to print Fibonacci series for n terms.

Code:

for loops.py - C:/Saif works/for loops.py (3.11.0)

File Edit Format Run Options Window Help

```
a=0
b=1
print(a)
print(b)
for i in range(1,9):
    c=a+b
    print(c)
    a,b=b,c
```

Output:

IDLE Shell 3.11.0

File Edit Shell Debug Options Window Help

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========================= RESTART: C:/Saif works/for loops.py =========================
0
1
1
2
3
5
8
13
21
34
>>>
```

5. Write a program to accept a string and display characters that are present at even index.

Code:

for loops.py - C:/Saif works/for loops.py (3.11.0)

File Edit Format Run Options Window Help

```
string=input("Enter a string:")
for i in string:
    if (string.index(i))%2==0:
        print(i)
```

Output:

```
IDLE Shell 3.11.0
File Edit Shell Debug Options Window Help
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====================== RESTART: C:/Saif works/for loops.py ====================
Enter a string:coding
c
d
n
>>>
======================= RESTART: C:/Saif works/for loops.py =====================
Enter a string:python
p
t
o
>>>
```

6. Write a program to print the below given pattern

```
*

**

***

****

*****
```

Code:

```
for loops.py - C:/Saif works/for loops.py (3.11.0)
File Edit Format Run Options Window Help
for i in range(1,6):
    print("*"*i)
```

Output:

```
IDLE Shell 3.11.0
File Edit Shell Debug Options Window Help
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====================== RESTART: C:/Saif works/for loops.py ====================
*
**
***
****
*****
>>>
```

PRACTICAL 4: WHILE STATEMENTS

1. Write a program to accept a number and print a factorial of a number.

Code:

while loop.py - C:/Saif works/while loop.py (3.11.0)

File Edit Format Run Options Window Help

```
n=int(input("Enter a number:"))
if n>=0:
    if n==0 or n==1:
        print("Factorial is 1")
    else:
        fact=1
        i=1
        while i<=n:
            fact=fact*i
            i+=1
        print(fact)
```

Output:

IDLE Shell 3.11.0

File Edit Shell Debug Options Window Help

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========================= RESTART: C:/Saif works/while loop.py =========================
Enter a number:5
120
>>>
========================= RESTART: C:/Saif works/while loop.py =========================
Enter a number:7
5040
>>>
========================= RESTART: C:/Saif works/while loop.py =========================
Enter a number:1
Factorial is 1
>>>
```

2. Write a program to accept a number and check if it is palindrome or not.

Code:

```
while loop.py - C:/Saif works/while loop.py (3.11.0)
File  Edit  Format  Run  Options  Window  Help
n=int(input("Enter a number:"))
rev=0
num=n
while n>0:
    d=n%10
    rev=rev*10+d
    n=n//10
if num==rev:
    print("The number is a palindrome")
else:
    print("The number is not a palindrome")
```

Output:

```
IDLE Shell 3.11.0
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===================== RESTART: C:/Saif works/while loop.py =====================
Enter a number:1551
The number is a palindrome
>>>
======================= RESTART: C:/Saif works/while loop.py =======================
Enter a number:1234
The number is not a palindrome
>>>
======================= RESTART: C:/Saif works/while loop.py =======================
Enter a number:1331
The number is a palindrome
>>>
```

3. Write a program to create a menu driven banking program. The user should be able to deposit, withdraw, and check balance with necessary conditions in place.

Code:

```
while loop.py - C:/Saif works/while loop.py (3.11.0)
File  Edit  Format  Run  Options  Window  Help
acc_no=int(input("Enter your account no:"))
amt=int(input("Enter current amount:"))
while True:
    print("Choose one among the following options")
    print("1.Withdraw 2.Deposit 3.Check balance")
    ch=input("Enter your choice (1,2or3):")
    if(ch=='1'):
        wamt=int(input("Enter amount to withdraw:"))
        if wamt<=amt:
            amt-=wamt
        else:
            print("insufficient balance")
    elif(ch=='2'):
        damt=int(input("Enter amount to deposit:"))
        amt+=damt
    elif(ch=='3'):
        print(f"Account no{acc_no} Current balance is{amt}")
    else:
        print("Enter a valid option")
    x=input("Do you want to continue (Y,N)")
    if x=='N':
        break
```

Output:

```
IDLE Shell 3.11.0
File  Edit  Shell  Debug  Options  Window  Help
    Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
    Type "help", "copyright", "credits" or "license()" for more information.
>>>
    ===================== RESTART: C:/Saif works/while loop.py =====================
    Enter your account no:56437850
    Enter current amount:6000
    Choose one among the following options
    1.Withdraw 2.Deposit 3.Check balance
    Enter your choice (1,2or3):1
    Enter amount to withdraw:4000
    Do you want to continue (Y,N)Y
    Choose one among the following options
    1.Withdraw 2.Deposit 3.Check balance
    Enter your choice (1,2or3):2
    Enter amount to deposit:2000
    Do you want to continue (Y,N)Y
    Choose one among the following options
    1.Withdraw 2.Deposit 3.Check balance
    Enter your choice (1,2or3):3
    Account no56437850 Current balance is4000
    Do you want to continue (Y,N)N
>>>
```

4. Write a program to accept numbers from user and calculate sum of numbers until user enters zero.

Code:

```
while loop.py - C:/Saif works/while loop.py (3.11.0)
File  Edit  Format  Run  Options  Window  Help
sum=0
while True:
    n=int(input("Enter a number:"))
    if n==0:
        break
    else:
        sum=sum+n
print("The sum of the number is",sum)
```

Output:

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===================== RESTART: C:/Saif works/while loop.py =====================
Enter a number:12
Enter a number:34
Enter a number:2
Enter a number:3
Enter a number:5
Enter a number:0
The sum of the number is 56
>>>
```

PRACTICAL 5: STRING FUNCTIONS

Aim: String operations & String Functions.

String Operations

1. Assignment Operator "="

   Code and Output:

```
In [1]:  str1='hello'
         str2="hello"
         str3='''hello'''
         print(str1)
         print(str2)
         print(str3)

         hello
         hello
         hello
```

2. Concatenate Operator “+”.

Code and Output:

```
In [3]:  s1="Welcome"
         s2="Home"
         s3=s1+s2
         print(s3)

         WelcomeHome
```

3. String Repetition Operator “*”.

Code and Output:

```
In [6]:  str1="WelcomeHome"
         print(str1*2)
         print(str1*3)
         print(str1*4)

         WelcomeHomeWelcomeHome
         WelcomeHomeWelcomeHomeWelcomeHome
         WelcomeHomeWelcomeHomeWelcomeHomeWelcomeHome
```

4. String Slicing Operator “[ ]”.

Code and Output:

```
In [7]:  str1="PythonWorld"
         print(str1[1])
         print(str1[-3])
         print(str1[1:5])
         print(str1[1:-3])
         print(str1[2:])
         print(str1[:5])
         print(str1[:-2])
         print(str1[-2:])
         print(str1[::-1])

         y
         r
         ytho
         ythonWo
         thonWorld
         Pytho
         PythonWor
         ld
         dlroWnohtyP
```

5. String Comparison Operator “==” & “!=”.

Code and Output:

```
In [9]:  str1= "Welcome"
         str2 = "Welcome,Home"
         str3 = "Welcome,Home"
         str4 = "Home"
         print(str1==str4)
         print(str2==str3)
         print(str1!=str4)
         print(str2!=str3)

         False
         True
         True
         False
```

6. Membership Operator “in” & “not in”.

Code and Output:

```
In [10]:  str1= "helloworld"
          print("w" in str1)
          print("w" in str1)
          print("t" in str1)
          print("t" not in str1)
          print("hello" in str1)
          print("Hello" in str1)
          print("hello" not in str1)

          True
          True
          False
          True
          True
          False
          False
```

String Functions

1. upper ():

   Code and Output:

```
In [1]:  text = "sweet home"
         result = text.upper()
         print(result)

         SWEET HOME
```

2. lower()

   Code and Output:

```
In [2]:  text = "SWEET HOME"
         result = text.lower()
         print(result)

         sweet home
```

3. capitalize()

   Code and Output:

```
In [3]:  text = "python programming"
         result = text.capitalize()
         print(result)

         Python programming
```

4. len()

Code and Output:

```
In [4]:  text = "Hello, how are you?"
         length = len(text)
         print(length)

         19
```

5. count()

Code and Output:

```
         19

In [5]:  text = "Lazy Dog"
         count = text.count("o")
         print(count)

         1
```

6. find()

Code and Output:

```
In [6]:  text = "Python programming are object oriented."
         index = text.find("object")
         print(index)

         23
```

7. center()

Code and Output:

```
In [7]:  text = "Jupyter Notebook"
         ct = text.center(20, '-')
         print(ct)

         --Jupyter Notebook--
```

8. replace()

Code and Output:

```
In [8]:  text = "I like apples, but I don't like bananas."
         new_text = text.replace("apples", "oranges")
         print(new_text)

         I like oranges, but I don't like bananas.
```

9. join()

Code and Output:

```
In [9]:  words = ["Hello", "Python","World"]
         result = " ".join(words)
         print(result)

         Hello Python World
```

10. strip()

Code and Output:

```
In [10]:  text = "   Hello, World!   "
          stripped_text = text.strip()
          print(stripped_text)

          Hello, World!
```

11. islower()

Code and Output:

```
In [11]:  text ="python programming"
          result = text.islower()
          print(result)

          True
```

12. isupper()

Code and Output:

```
In [12]:  text = "PYTHON PROGRAMMING"
          result = text.isupper()
          print(result)

          True
```

13. isalnum()

Code and Output:

```
In [10]:  text = "   Hello, World!   "
          stripped_text = text.strip()
          print(stripped_text)

          Hello, World!
```

14. isdigit()

Code and Output:

```
In [14]:  text = "12345"
          result = text.isdigit()
          print(result)

          True
```

15. isspace()

Code and Output:

```
In [15]:  text = ""
          result = text.isspace()
          print(result)

          False
```

16. isprintable()

Code and Output:

```
In [16]:  text = "Hello, World!"
          result = text.isprintable()
          print(result)

          True
```

17. format()

Code and Output:

```
In [17]:  name = "Alice"
          age = 20
          ft = "My name is {} and I am {} years old.".format(name, age)
          print(ft)

          My name is Alice and I am 20 years old.
```

PRACTICAL 6: LISTS FUNCTIONS

▪ LISTS OPERATORS

#Length(len)

LIST OPERATORS:-

```
In [1]:  #Length (len)
         len([2,4,6,8,10])

Out[1]: 5
```

#Concatention

```
In [2]:  #CONCATENATION
         l1=[1,2,3]
         l2=[6,7,8]
         l3=l1+l2
         l3

Out[2]: 13
```

#Repetition

```
In [3]:  #Repetition
         l4=l1*3
         l4

Out[3]: [1, 2, 3, 1, 2, 3, 1, 2, 3]
```

#Membership

```
In [4]:  #Membership
         6 in l2

Out[4]: True
```

Slicing

```
In [5]:  #Slicing -
         l5=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]
         l5[8]

Out[5]: 9

In [6]:  l5[-5]

Out[6]: 11

In [7]:  l5[2:8]

Out[7]: [3, 4, 5, 6, 7, 8]

In [8]:  l5[::-2]

Out[8]: [15, 13, 11, 9, 7, 5, 3, 1]
```

LIST GENERAL FUNCTIONS:-

Length

LIST GENERAL FUNCTIONS:-

```
In [9]:  #len(list)
         len([15,16,17,18,19])

Out[9]: 5
```

#Max & #Min

```
In [10]:  #max(list)
          max(15)

Out[10]: 15

In [11]:  #min(list)
          min(15)

Out[11]: 1
```

#List(seq)

```
In [12]:  #list(seq)
          l1=(2,3,4,5)
          list(l1)

Out[12]: [2, 3, 4, 5]
```

List Specific Functions:-

#Append

List Specific Functions:-

```
In [13]:  l1=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]
          #.append()
          l1.append(25)
          l1

Out[13]: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 25]
```

#Count

```
In [14]:  #count()
          l2=[1,2,3,1,2,3,1,2,3,1,2,3]
          l2.count(3)

Out[14]: 4
```

Extend

```
In [15]:  #extend()
          l3=[16,17,18]
          l1.extend(l3)
          l1

Out[15]: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 25, 16, 17, 18]
```

#Index

```
In [16]: #index
         l1.index(13)
Out[16]: 12
```

#Insert

```
In [17]: #insert()
         l1.insert(15,24)
         l1
Out[17]: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 24, 25, 16, 17, 18]
```

#Pop

```
In [18]: #pop()
         l1.pop()
Out[18]: 18

In [19]: l1
Out[19]: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 24, 25, 16, 17]
```

#remove

```
In [20]: #remove()
         l1.remove(24)
         l1
Out[20]: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 25, 16, 17]
```

#reverse

```
In [21]: #.reverse()
         l1.reverse()
         l1
Out[21]: [17, 16, 25, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
```

#sort

```
In [22]: #sort()
         marks=[70,11,90,88,61,59]
         marks.sort()
         marks
Out[22]: [11, 59, 61, 70, 88, 90]

In [23]: marks.sort(reverse=True)
         marks
Out[23]: [90, 88, 70, 61, 59, 11]
```

#Sort & Reverse

```
In [24]: l4=["Hello","World","Python"]
         l4.sort(key=len)
         l4
Out[24]: ['Hello', 'World', 'Python']

In [25]: l4.sort(key=len,reverse=True)
         l4
Out[25]: ['Python', 'Hello', 'World']
```

#copy

```
In [26]:  #Copy()
          l5=l4
          l5

Out[26]: ['Python', 'Hello', 'World']
```

#append(list)

```
In [27]:  #append(list)
          l4.append("Progrmming")
          l4

Out[27]: ['Python', 'Hello', 'World', 'Progrmming']

In [28]:  l5

Out[28]: ['Python', 'Hello', 'World', 'Progrmming']
```

#List comprehension

```
In [29]:  l8 = [i for i in range (1,11)]
          l8

Out[29]: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

In [32]:  l9=[i for i in range (len(l8))if i%2==0]
          l9

Out[32]: [0, 2, 4, 6, 8]
```

 # Update

```
In [37]:  l10=["apple", "cherry", "orange", "mango", "orange"]
          up_l10=["custard" if fruit=="orange" else fruit for fruit in l10]
          print(up_l10)

          ['apple', 'cherry', 'custard', 'mango', 'custard']
```

#Lambda

```
In [38]: l2=[100,50,40,20]
         l2.sort(key=lambda n:abs(n-50))
         l2

Out[38]: [50, 40, 20, 100]
```

Upper and lower()

```
In [40]: students=['Mona','Mohan','Meena','Manish']
         students1=[name.upper() for name in students]
         students1

Out[40]: ['MONA', 'MOHAN', 'MEENA', 'MANISH']

In [41]: students=['Mona','Mohan','Meena','Manish']
         students1=[name.lower() for name in students]
         students1

Out[41]: ['mona', 'mohan', 'meena', 'manish']
```

Sortlist()

```
In [45]: def sortlist(n):
             return abs(n-50)
         diff=[]
         for i in range (len(l6)):
             diff.append(sortlist(l6[i]))
         diff

Out[45]: [50, 49, 47, 46, 45, 44, 43, 42, 41, 40]

In [46]: l6=[100,50,20,80,40]
         l6.sort(key=sortlist)
         l6

Out[46]: [50, 40, 20, 80, 100]
```

PRACTICAL 7: DICTIONARY

1. #create dictionary

```
In [1]:  1. #create dictionary
         dict1={1:"Mohan",2:"Manish",3:"Madan"}
         dict2= dict(name = "John", age = 36, country =
         "Norway")
         dict3 = {
          "brand": "Ford",
          "electric": False,
          "year": 1964,
          "colors": ["red", "white", "blue"]
         }
```

2. #len()-length of dictionary

```
In [2]:  2.#len()- length of dictionary
         len(dict1)

Out[2]: 3
```

3. #Access dictionary items

```
In [3]:  3. #Access dictionary items
         dict2['name']
         dict2.get('name')

Out[3]: 'John'
```

4. #keys() - get all keys of a dictionary

```
In [4]:  4. #keys() - get all keys of a dictionary
         dict1.keys()

Out[4]: dict_keys([1, 2, 3])
```

5. #values() - get all values of a dictionary

```
In [5]:  5. #values() - get all values of a dictionary
         dict1.values()

Out[5]: dict_values(['Mohan', 'Manish', 'Madan'])
```

6. #items() - will return each item in a dictionary, as tuples in a list

```
In [6]:  6. #items() -  will return each item in a dictionary, as tuples in a list
         dict3.items()

Out[6]: dict_items([('brand', 'Ford'), ('electric', False), ('year', 1964), ('colors', ['red', 'white', 'blue'])])
```

7. #Change items

```
In [7]:  7. #Change items
         dict2[age] = 40
         dict2.update({"age":50})

In [8]:  dict2

Out[8]: {'name': 'John', 'age': 50, 'country': 'Norway'}
```

8. #Add dictionary items

```
In [9]:  8. #Add dictionary items
         dict2["gender"] ="male"
         dict2.update({"contact":"9541230200"})

In [10]: dict2

Out[10]: {'name': 'John',
          'age': 50,
          'country': 'Norway',
          'gender': 'male',
          'contact': '9541230200'}
```

9. #Remove dictionary items

```
In [11]:  9.#Remove dictionary items
          dict2.pop("gender")    #removes the item with the specified key name
          dict2.popitem()      #removes the last inserted item
          del dict2["age"]     #removes the item with the specified key name
          del dict2          #del keyword can also delete the dictionary completely
```

10. #clear()- clear dictionary content

```
In [12]:  11. #clear()- clear dictionary content
          dict1.clear()

In [13]:  dict1

Out[13]:  {}
```

11. #copy()

```
In [14]:  12.#copy()
          dict4= dict3.copy()
          dict5 = dict(dict1)

In [15]:  dict5

Out[15]:  {}
```

12. #Nested Dictionaries
    #Access nested dictionary items

```
In [16]:  #Nested Dictionaries
          #Access nested dictionary items
          myfamily = {
          "child1": {
              "name" :"Emil",
              "year" :2004
              },
              "child2" : {
                  "name": "Tobias",
                  "year": 2007
              },
              "child3": {
                  "name": "Lirus",
                  "year":  2011
              }

          }
          print("Child 2 name :" ,myfamily["child2"]["name"])
          print("Child 3 name :" ,myfamily["child3"]["year"])

          Child 2 name : Tobias
          Child 3 name : 2011
```

PRACTICAL 8: FUNCTIONS

1) Implement factorial using anonymous function.

CODE AND OUTPUT:

```
In [9]: x = lambda num : 1 if num <= 1 else num*x(num-1)

        number = int(input('Enter number: '))

        print('%d != %d' %(number, x(number)))

        Enter number: 5
        5 != 120
```

2) Write a program to display the payslip of an employee, where employee id and basic pay is to the function; DA=90%basicpay, HRA=40%basicpay, TA=20%basicpay and NET salary= DA+TA+HRA+basicpay.

CODE AND OUTPUT:

```
In [4]: def payslip():
            empID=int(input("Enter employee id:"))
            BP=int(input("Enter your basicsalary:"))
            DA=(BP*90)/100
            TA=(BP*20)/100
            HRA=(BP*40)/100
            NET=BP+DA+TA+HRA
            print(f"*****EMPLOYEE DETAILS*****\nDearness Allowance:{DA}\nHouse Rent Allowance:{HRA}\nTravel Allowance:{TA}\nNet Salary:{N
        payslip()

        Enter employee id:1
        Enter your basicsalary:50000
        *****EMPLOYEE DETAILS*****
        Dearness Allowance:45000.0
        House Rent Allowance:20000.0
        Travel Allowance:10000.0
        Net Salary:125000.0
```

```
,nHouse Rent Allowance:{HRA}\nTravel Allowance:{TA}\nNet Salary:{NET}")
```

3) Implement a function in python to check if a number is Krishnamurthy or not.

CODE AND OUTPUT:

```
In [1]: def factorial(n):
            fact=1
            for i in range(1,n+1):
                fact=fact*i
                n=n-1
            return fact
        def is_krishnum(n):
            sum=0
            temp=n
            while(temp!=0):
                rem=temp%10
                sum=sum+factorial(rem)
                temp=temp//10
            return(sum==n)
        n=int(input("Enter a number:"))
        if (is_krishnum(n)):
            print("YES")
        else:
            print("NO")

        Enter a number:123
        NO
```

```
In [2]: def factorial(n):
            fact=1
            for i in range(1,n+1):
                fact=fact*i
                n=n-1
            return fact
        def is_krishnum(n):
            sum=0
            temp=n
            while(temp!=0):
                rem=temp%10
                sum=sum+factorial(rem)
                temp=temp//10
            return(sum==n)
        n=int(input("Enter a number:"))
        if (is_krishnum(n)):
            print("YES")
        else:
            print("NO")

        Enter a number:145
        YES
```

4) Implement conversion from decimal to binary using recursive function.

CODE AND OUTPUT:

```
In [5]: def convertToBinary(n):
            if n>1:
                convertToBinary(n//2)
            print(n%2,end=" ")
        #decimal number
        n=int(input("Enter a number:"))
        convertToBinary(n)
        print()

        Enter a number:34
        1 0 0 0 1 0
```

PRACTICAL 9: FILE HANDLING

Aim: - File handling in Python.

- To read the file and close.

```
In [1]: #to read and close the file
        file1=open("C:\\Users\\user\\Desktop\\Python files\\weekdays.txt","r")
        #the file is opened with the help of open() function and set on read mode with the help of parameter "r"
        x=file1.read() #read() function is used to read the file.
        print(x)
        file1.close() #close() function is used to close the file.

        Monday
        Tuesday
        Wednesday
        Thursday
        Friday
        Saturday
        Sunday
```

- To close the file without using close () function.

```
In [2]: #to close the file without using close function.
        with open("C:\\Users\\user\\Desktop\\Python files\\weekdays.txt")as file1:#read is always on default mode
            x=file1.readlines()#in readlines() function list contents are conprised as string and gets displayed linewise.
            print(x)

        ['Monday \n', 'Tuesday\n', 'Wednesday\n', 'Thursday\n', 'Friday \n', 'Saturday \n']
```

- To close the file with using for loop.

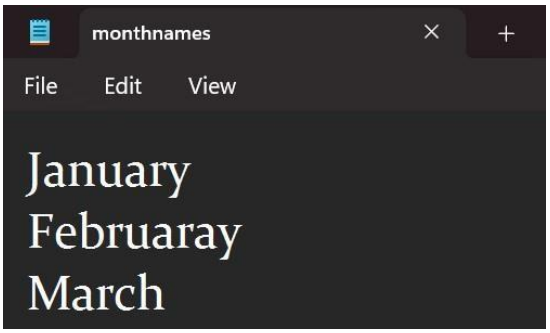
```
In [3]:  #to close the file without using close function.
         with open("C:\\Users\\user\\Desktop\\Python files\\weekdays.txt")as file1:#read is always on default mode
             x=file1.readlines()#in readlines() function list contents are conprised as string and gets displayed linewise.
             for day in x: #printing output using for loop
                 print(day)

         Monday

         Tuesday

         Wednesday

         Thursday

         Friday

         Saturday
```

- To write the file.

CODE:

```
In [4]:  #to write the file
         with open("C:\\Users\\user\\Desktop\\Python files\\monthnames.txt","w")as file1:
             file1.write("January\n")
             file1.write("Februaray\n")
             file1.write("March\n")
```

OUTPUT:

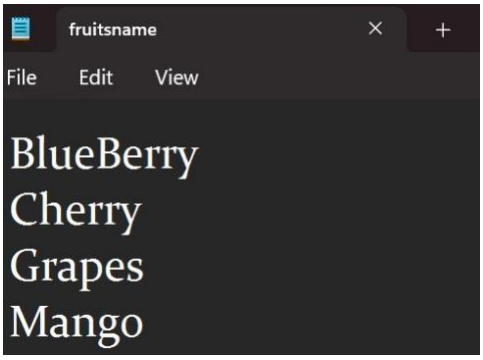


- To write with write lines () function.

CODE:

```
In [5]:  #to write with writelines()
         x=["BlueBerry\n","Cherry\n","Grapes\n","Mango\n"]
         with open("C:\\Users\\user\\Desktop\\Python files\\fruitsname.txt","w")as file1:
             file1.writelines(x)#writelines()function is used to write list in a file.
```

OUTPUT:

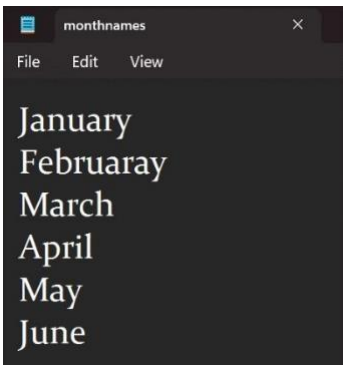


- To Append

CODE:

```
In [8]:  #to append
         mn=["April\n", "May\n","June\n"]
         with open("C:\\Users\\user\\Desktop\\Python files\\monthnames.txt","a")as file1:
                 filel.writelines(mn)
```

OUTPUT:



- To read file without using read () function. (using loop function)

```
In [9]:  #to read file without using read() function (using for loop)
         with open("C:\\Users\\user\\Desktop\\Python files\\weekdays.txt")as file1:
             for line in file1:
                 print(line.strip())

         Monday
         Tuesday
         Wednesday
         Thursday
         Friday
         Saturday
```

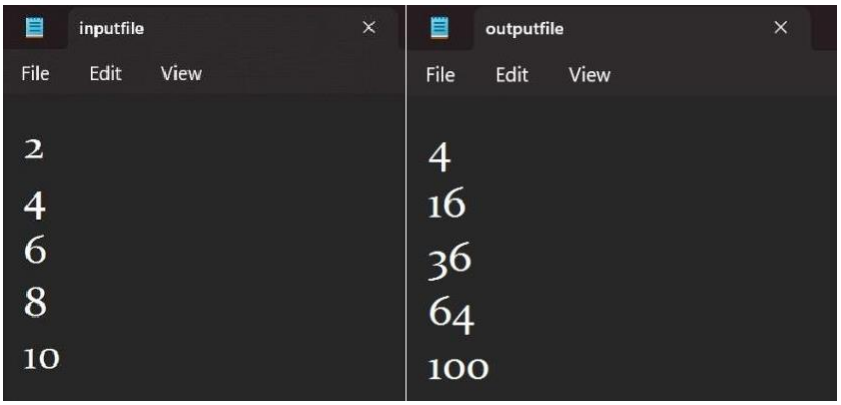
- Program to read numbers from input file and write its square to output file.

CODE:

```
In [10]: #program to read numbers from inputfile and write its square to outputfile.
         with open("C:\\Users\\user\\Desktop\\Python files\\inputfile.txt")as file1, open("C:\\Users\\user\\Desktop\\Python files\\out
             for line in file1:
                 x=int(line.strip())
                 y=x*x
                 file2.write(str(y)+"\n")
```

```
utputfile.
)as file1, open("C:\\Users\\user\\Desktop\\Python files\\outputfile.txt","w")as file2:
```

OUTPUT:

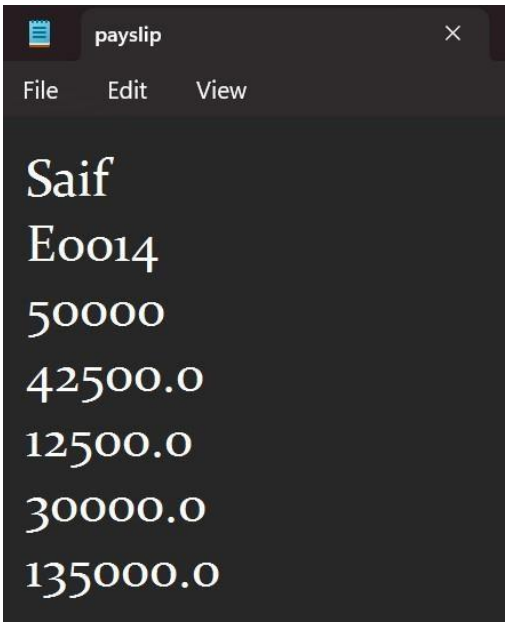


- Write a python program with an input file comprising of an employee details namely employee name, employee code, basic salary. Create an output file as pays lip will contain employee name, employee code, basic salary, da, ta, hra and net salary.

CODE:

```
In [15]:  with open ("C:\\Users\\user\\Desktop\\Python files\\employeedetails.txt")as file1, open("C:\\Users\\user\\Desktop\\Python fil
              x=file1.readlines()
              basic=int(x.pop(2))#popping the last item &converting it into integer to find the components of the basic salary.
              da=0.85*basic
              ta=0.25*basic
              hra=0.60*basic
              net=basic+da+ta+hra
              l1=[basic,da,ta,hra,net]#putting all the values in the list as it is neended to convert into string.
              for i in range (0,len(l1)):
                  l1[i]=str(l1[i])
              file2.writelines(x)
              y="\n".join(l1)#to make a list of string
              file2.writelines(y)
```

OUTPUT:



- To check the current working directory with the help of module and its function.

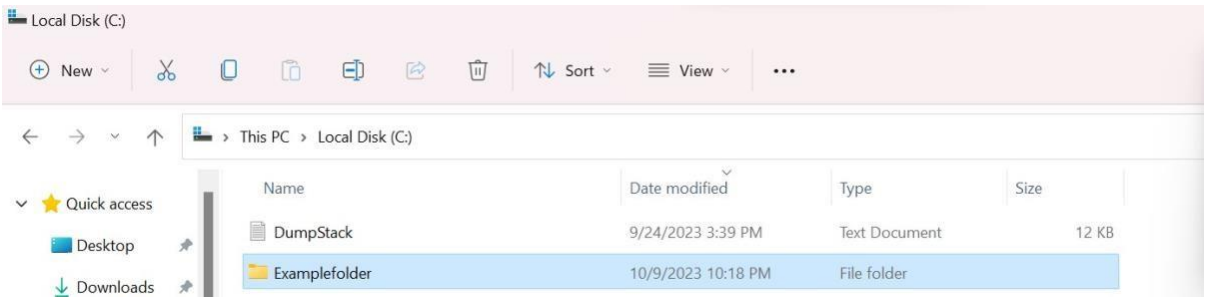
```
In [1]:  import os
         print(os.getcwd())

C:\Users\user\Jupyter
```

- To change the current working directory.

```
In [6]:  os.chdir("C:\\Users\\user\\Desktop\\Python files")
         print(os.getcwd())

C:\Users\user\Desktop\Python files
```

- To make directory.

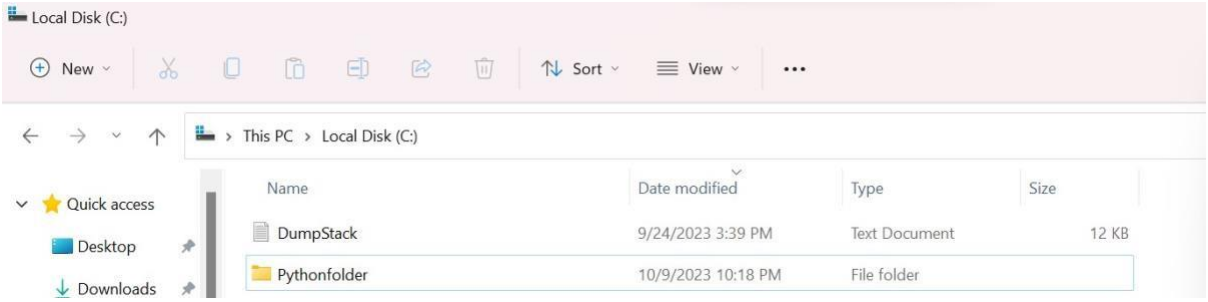
```
C:\Users\user\Desktop\Python files

In [7]:  os.mkdir("C:\\Examplefolder")
```



❖ To rename a folder.

```
In [10]: os.rename("C:\\Examplefolder",("C:\\Pythonfolder"))
```

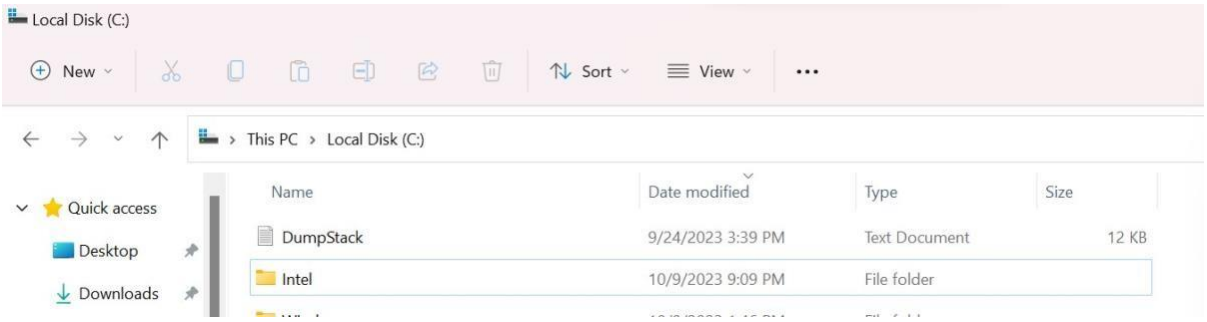
OUTPUT:



❖ To remove directory. CODE:

```
In [9]: os.rmdir("C:\\Pythonfolder")
```

OUTPUT:

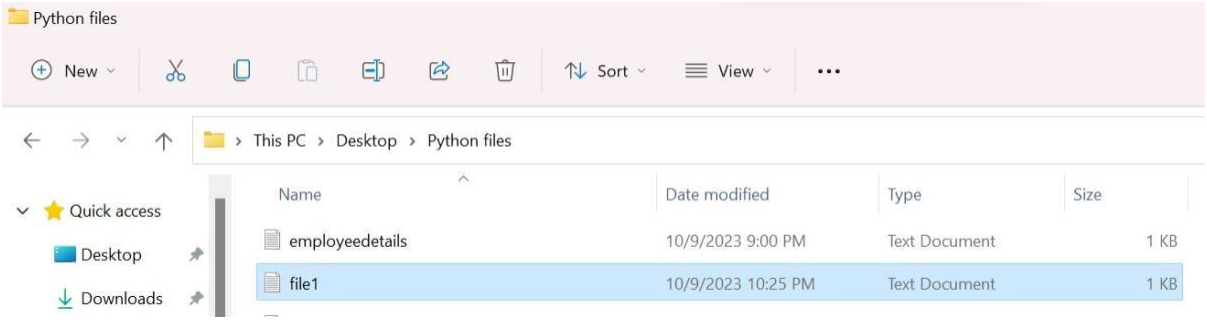


❖ To remove files. CODE:

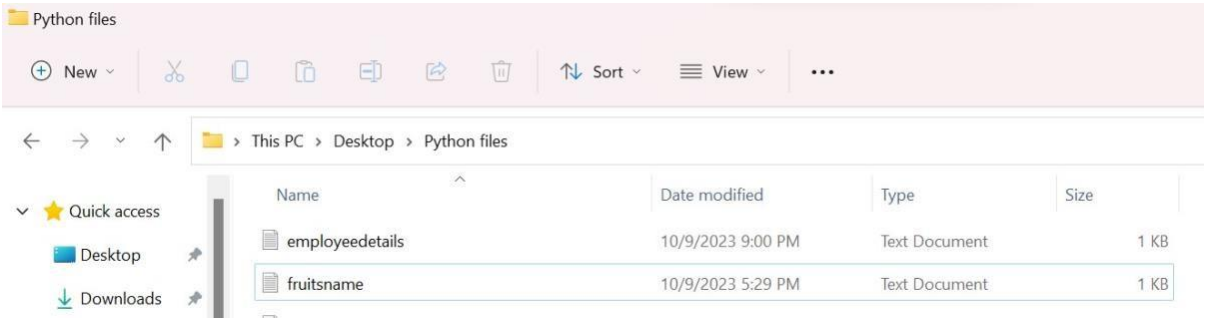
```
In [11]: os.remove("file1.txt")
```

Output: -

Before Execution: -



After execution: -



PRACTICAL 10: EXCEPTION HANDLING

1. Write a program to catch ZeroDivisionError, Name error

 TO CATCH THE ZERO DIVISION ERROR

```
In [4]:
#To Catch the ZeroDivisionError
try:
    a=6
    b=0
    res=a/b #ZeroDivisionError
    print(c)#NameError
except ZeroDivisionError:
    print("Denominator cannot be 0")
except NameError:
    print("Name is not defined")

Denominator cannot be 0
```

 TO CATCH THE NAME ERROR

```
In [5]:
#To Catch the Name Error
try:
    a=6
    b=0
    print(c)#NameError
    res=a/b #ZeroDivisionError
except ZeroDivisionError:
    print("Denominator cannot be 0")
except NameError:
    print("Name is not defined")

Name is not defined
```

2. Write a program to catch Indexerror, typeerror, valueerror and finallyblock

 TO CATCH INDEX ERROR

```
In [6]:
#To catch IndexError
try:
    a=[1,2,'x','y']
    print(a[5]) #IndexError
    print(x[1]+x[2]) #TypeError
    c=int(x[3])#ValueError
except IndexError:
    print("Error: Index out of bound")
except TypeError:
     print("Error: Type Error")
except ValueError:
     print("Error: Value Error")
finally:
    print("This is finally block")

Error: Index out of bound
This is finally block
```

 TO CATCH TYPE ERRROR

```
In [8]:  #To catch TypeError
         try:
             a=[1,2,'x','y']
             print(a[1]+a[2]) #TypeError
             print(a[5]) #IndexError
             c=int(x[3])#ValueError
         except IndexError:
             print("Error: Index out of bound")
         except TypeError:
              print("Error: Type Error")
         except ValueError:
              print("Error: Value Error")
         finally:
             print("This is finally block")

         Error: Type Error
         This is finally block
```

- TO CATCH VALUE ERROR

```
In [10]:  #To catch ValueError
          try:
              a=[1,2,'x','y']
              c=int(a[3])#ValueError
              print(a[1]+a[2]) #TypeError
              print(a[5]) #IndexError
          except IndexError:
              print("Error: Index out of bound")
          except TypeError:
               print("Error: Type Error")
          except ValueError:
               print("Error: Value Error")
          finally:
              print("This is finally block")

          Error: Value Error
          This is finally block
```

PRACTICAL 11: REGULAR EXPRESSION

Program to demonstrate the use of regular expressions.

1.1 Username-Starts with a lowercase alphabet and can contain minimum 8 character and maximum 15 character the after starting letter can be any alphanumeric character

1.2 Mobile number – contain 10 digits

1.3 Email I'd Format

CODE AND OUTPUT:

```
In [7]:  #1.1.UserName-starts with an lowercasealphabet and can contain a minimum 8 characters and maximum 15 characters
         #of alphanumeric characters.
         #1.2.Mobile number- contains 10 digits
         #1.3.email id format

         import re

         username_pattern = r'^[a-zA-Z]\w{7,14}$'
         user_input = input("Enter your Name: ")
         x = bool(re.match(username_pattern, user_input))

         if x:
             print(f"Username '{user_input}' is valid.")
         else:
             print(f"Username '{user_input}' is not valid.")

         mobile_pattern = r'^\d{10}$'
         mobile_input = input("Enter your Mobile number: ")
         y = bool(re.match(mobile_pattern, mobile_input))

         if y:
             print(f"Mobile number '{mobile_input}' is valid.")
         else:
             print(f"Mobile number '{mobile_input}' is not valid.")

         email_pattern = r'^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$'
         email_input = input("Enter your Email Id: ")
         z = bool(re.match(email_pattern, email_input))

         if z:
             print(f"Email address '{email_input}' is valid.")
         else:
             print(f"Email address '{email_input}' is not valid.")

         Enter your Name: mantasha
         Username 'mantasha' is valid.
         Enter your Mobile number: 9004955446
         Mobile number '9004955446' is valid.
         Enter your Email Id: shaikhmantasha05@gmail.com
         Email address 'shaikhmantasha05@gmail.com' is valid.
```

2. Write a Python function text_match() that matches a string that has an ‘a’ followed by zero or more occurrence #of anything ,ending in ‘b’.

<u>CODE AND OUTPUT:</u>

```
In [1]:
import re

def text_match(input_string):

    pattern = r'a.*b'

    matches = re.findall(pattern, input_string)


    return matches


input_string =(input("Enter a String"))
result = text_match(input_string)
print(result)

Enter a Stringabcd
['ab']
```

3. Write a python program to accept an address and replace occurrences of Road by Rd.,District by Dst. And Street by St.

CODE AND OUTPUT:

```
In [8]:
# Write a python program to accept an address and replace occurrences of Road by Rd.,District by Dst.,and Street by St.
patt=["Street","Road","District"]
code=["St.","Rd.","Dst."]
text=input("Enter address:")
i=0
while i!=3:
    if patt[i] in text:
        text=re.sub(patt[i],code[i],text)
    i+=1
print(text)

Enter address:A-001, Sunshine Apt, Maple Street, Mumbai District
A-001, Sunshine Apt, Maple St., Mumbai Dst.
```

PRACTICAL 12: DATABASE HANDLING

* **AIM** :- To learn database connectivity in python.

* **CODE & OUTPUT** :-

##Write a database program to perform the following:-

- Show all databases in DBMS.
- Create a database named “Company”.

- Create a table Employee in Company Database(empid int, empname varchar(50), designation varchar(50), basic int).

- Insert 10 records into employee table.
- Display all records of employee table.
- Display the employees details with designation as entered by the user at runtime.

- Display the employee details with basic < 20000.
- Update the basic of Manager by 30%.

● **<u>CODE & OUTPUT</u> :-**

##INSTALLATION OF MYSQL CONNECTOR.

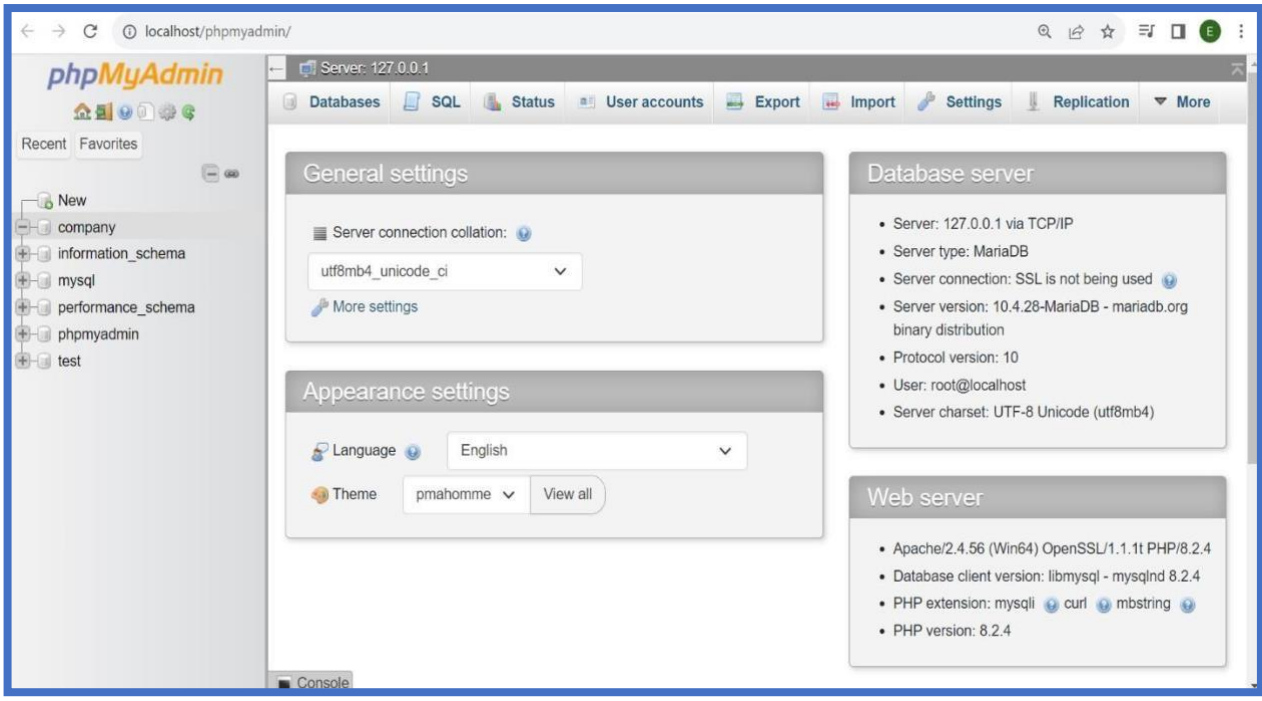
```
In [1]: !pip install mysql-connector-python

        Requirement already satisfied: mysql-connector-python in c:\eisha\lib\site-packages (8.1.0)
        Requirement already satisfied: protobuf<=4.21.12,>=4.21.1 in c:\eisha\lib\site-packages (from mysql-c
        onnector-python) (4.21.12)
```

##TO SHOW THE DATABASES IN 'DBMS'.

```
In [4]: import mysql.connector
        con=mysql.connector.connect(host='localhost',database='mysql',user='root',password='')
        mycur=con.cursor()
        mycur.execute("SHOW DATABASES")
        for data in mycur:
            print(data)
        mycur.close()

        ('information_schema',)
        ('mysql',)
        ('performance_schema',)
        ('phpmyadmin',)
        ('test',)

Out[4]: True
```

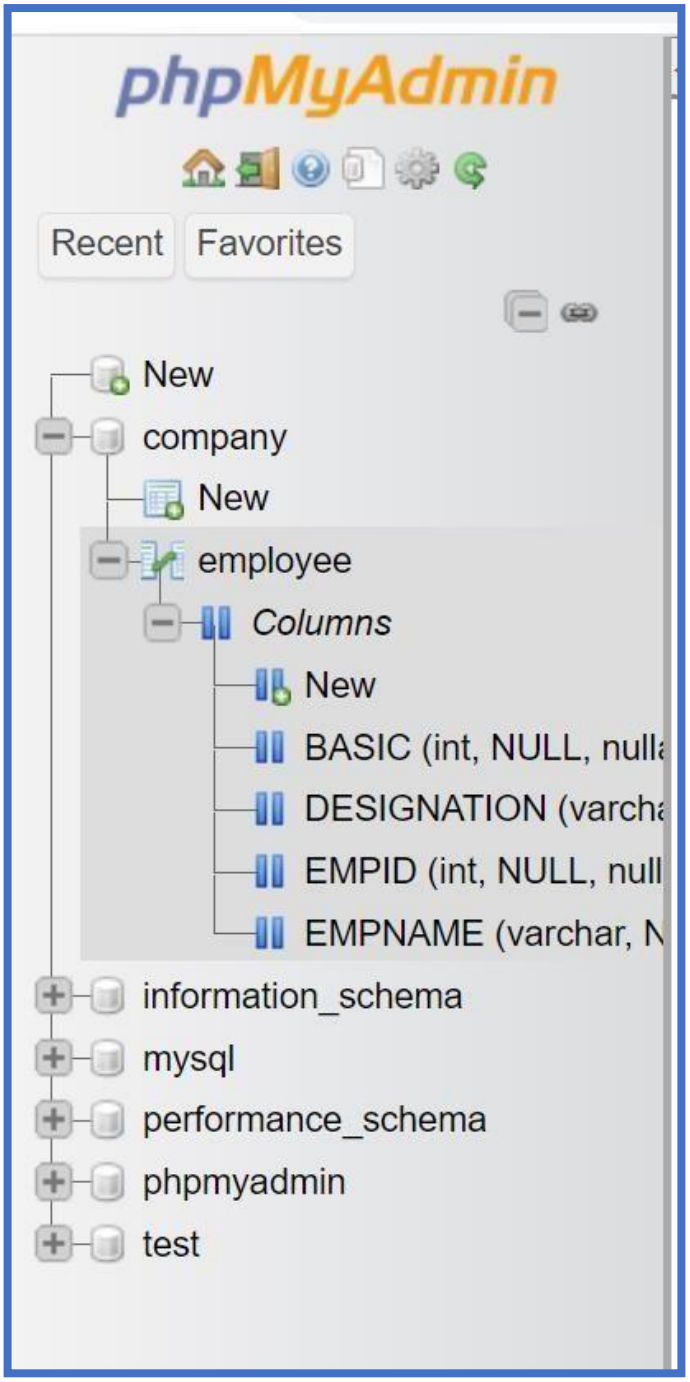
##TO CREATE DATABASE “COMPANY”.

```
In [5]: import mysql.connector
        con=mysql.connector.connect(host='localhost',database='mysql',user='root',password='')
        mycur=con.cursor()
        mycur.execute("CREATE DATABASE COMPANY")
        mycur.close()

Out[5]: True
```



##TO CREATE A TABLE EMPLOYEE IN COMPANY DATABASE(EMPID INT, EMPNAME VARCHAR(50), DESIGNATION VARCHAR(50),

```
import mysql.connector
con=mysql.connector.connect(host='localhost',database='mysql',user='root',password='')
mycur=con.cursor()
mycur.execute("CREATE TABLE EMPLOYEE(EMPID INT,EMPNAME VARCHAR(50),DESIGNATION VARCHAR(50),BASIC INT)")
mycur.close()
```



##TO INSERT 10 RECORDS INTO EMPLOYEE TABLE.

```
In [17]: import mysql.connector
         con=mysql.connector.connect(host='localhost',database='company',user='root',password='')
         mycur=con.cursor()
         empdetails=[(1,'TOM','CLERK',30000),(2,'BOB','CLERK',30000),(3,'HELEN','CLERK',30000),(4,'SWEETY','CLER
         mycur.executemany("INSERT INTO EMPLOYEE(EMPID,EMPNAME,DESIGNATION,BASIC) VALUES(%s,%s,%s,%s)",empdetail
         con.commit()
         mycur.close()

Out[17]: True
```

##TO DISPLAY ALL RECORDS OF EMPLOYEE TABLE.

```
In [18]: import mysql.connector
         con=mysql.connector.connect(host='localhost',database='company',user='root',password='')
         mycur=con.cursor()
         mycur.execute("SELECT*FROM EMPLOYEE")
         for data in mycur:
             print(data)
         mycur.close()

         (1, 'TOM', 'CLERK', 30000)
         (2, 'BOB', 'CLERK', 30000)
         (3, 'HELEN', 'CLERK', 30000)
         (4, 'SWEETY', 'CLERK', 15000)
         (5, 'AISHU', 'CLERK', 15000)
         (6, 'JERRY', 'CLERK', 30000)
         (7, 'JOEY', 'MANAGER', 90000)
         (8, 'SURYA', 'ASST.MANAGER', 50000)
         (9, 'SHREYA', 'CEO', 200000)
         (10, 'NICK', 'CLERK', 15000)
```

##TO DISPLAY THE EMPLOYEES DETAILS WITH DESIGNATION AS ENTERED BY THE USER AT RUNTIME.

```
In [21]: import mysql.connector
         con=mysql.connector.connect(host='localhost',database='company',user='root',password='')
         mycur=con.cursor()
         mycur.execute("SELECT*FROM EMPLOYEE WHERE DESIGNATION='CEO'")
         for data in mycur:
             print(data)
         mycur.close()

         (9, 'SHREYA', 'CEO', 200000)
```

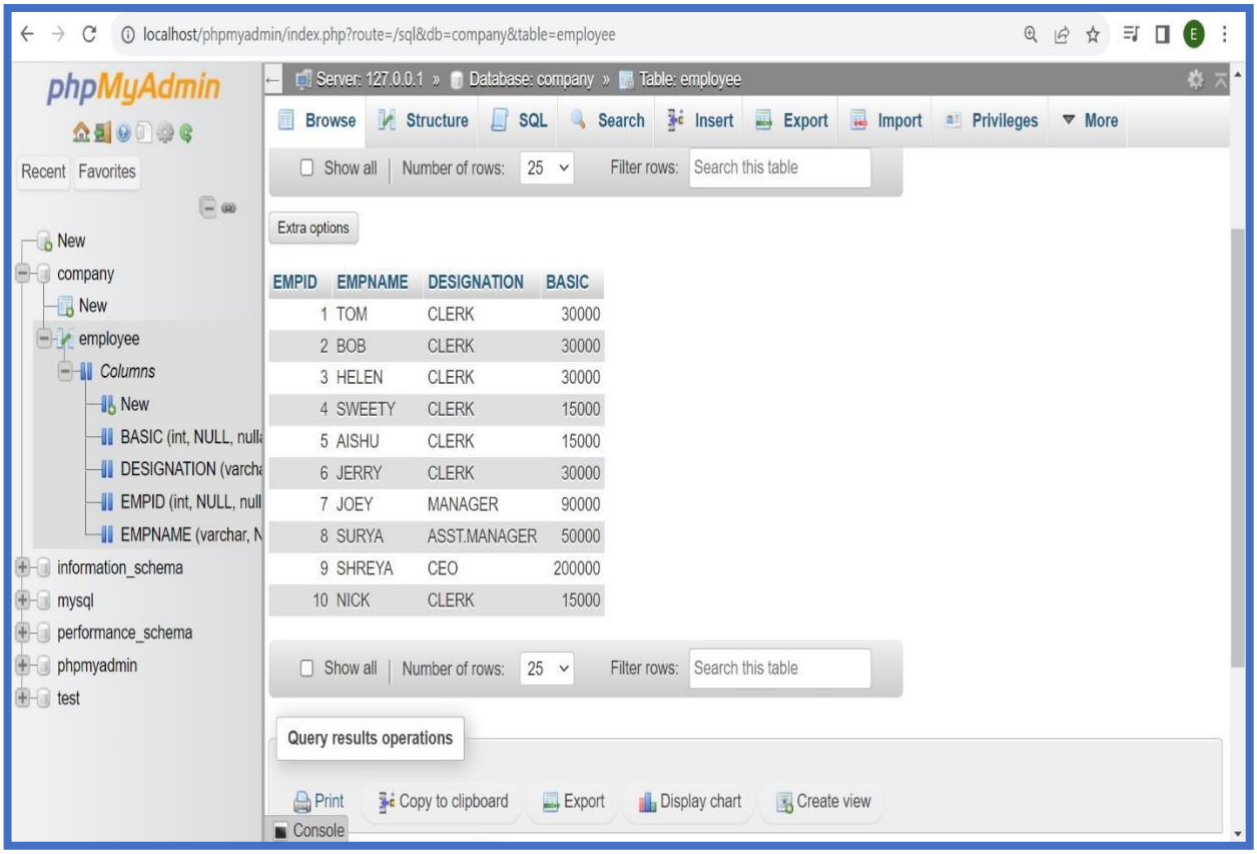
##TO DISPLAY THE EMPLOYEE DETAILS WITH BASIC < 20000.

```
In [22]: import mysql.connector
         con=mysql.connector.connect(host='localhost',database='company',user='root',password='')
         mycur=con.cursor()
         mycur.execute("SELECT*FROM EMPLOYEE WHERE BASIC<20000")
         for data in mycur:
             print(data)
         mycur.close()

         (4, 'SWEETY', 'CLERK', 15000)
         (5, 'AISHU', 'CLERK', 15000)
         (10, 'NICK', 'CLERK', 15000)
```

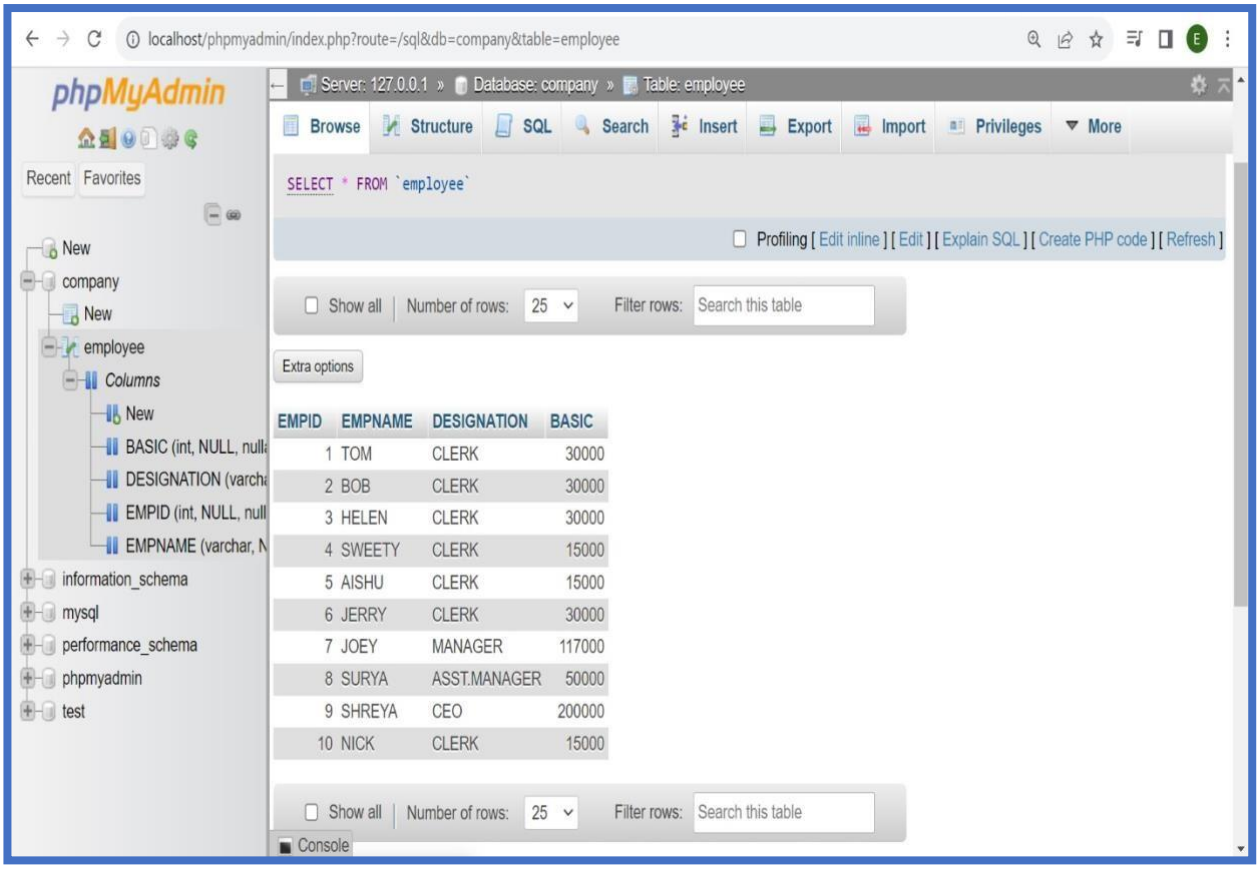
##TO UPDATE THE BASIC OF MANAGER BY 30%.

```
In [31]: import mysql.connector
         con=mysql.connector.connect(host='localhost',database='company',user='root',password='')
         mycur=con.cursor()
         mycur.execute("UPDATE EMPLOYEE SET BASIC=BASIC+0.3*BASIC WHERE DESIGNATION='MANAGER'")
         con.commit()
         mycur.close()

Out[31]: True
```

##BEFORE EXECUTION :-



##AFTER EXECUTION :-


localhost/phpmyadmin/index.php?route=/sql&db=company&table=employee
phpMyAdmin
Recent
Favorites
New
company
New
employee
Columns
New
BASIC (int, NULL, nulla
DESIGNATION (varcha
EMPID (int, NULL, null
EMPNAME (varchar, N
information_schema
mysql
performance_schema
phpmyadmin
test
Server: 127.0.0.1 » Database: company » Table: employee
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
More
SELECT * FROM `employee`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all
Number of rows: 25
Filter rows: Search this table
Extra options
EMPID EMPNAME DESIGNATION BASIC
1 TOM CLERK 30000
2 BOB CLERK 30000
3 HELEN CLERK 30000
4 SWEETY CLERK 15000
5 AISHU CLERK 15000
6 JERRY CLERK 30000
7 JOEY MANAGER 117000
8 SURYA ASST.MANAGER 50000
9 SHREYA CEO 200000
10 NICK CLERK 15000
Show all
Number of rows: 25
Filter rows: Search this table
Console