**Practical Assignment – I**

1. Display the difference in dates

#Display the difference in dates

**from** datetime **import** datetime

dt1\_str=input("\n enter date 1 in yyyy-mm-dd format")

dt2\_str=input("\n enter date 2 in yyyy-mm-dd format")

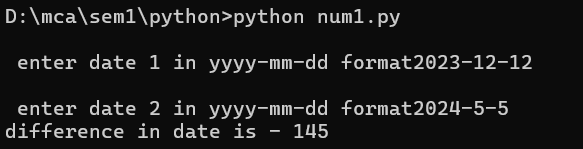
dt1=datetime.strptime(dt1\_str,"%Y-%m-%d")

dt2=datetime.strptime(dt2\_str,"%Y-%m-%d")

new\_date=dt2-dt1

print(f"difference in date is - {new\_date.days}")

Output:



2. Display time since epoch in hours and minutes

#Display time since epoch in hours and minutes

**import** time

**from** datetime **import** datetime

epoch\_time=time.time()

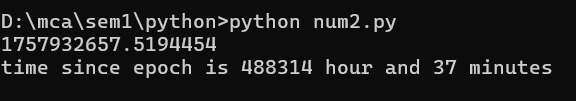
print(epoch\_time)

hrs=int(epoch\_time//3600)

mins=int((epoch\_time%3600)//60)

print(f"time since epoch is {hrs} hour and {mins} minutes")

Output:



3. Display your age in years, months and days

**from** datetime **import** datetime

dt\_str=input("\n enter birth date in yyyy-mm-dd format")

dt=datetime.strptime(dt\_str,"%Y-%m-%d")

dt\_now=datetime.now().date()

yr=dt\_now.year-dt.year

mon=dt\_now.month-dt.month

day=dt\_now.day-dt.day

**if**(day<0):

    mon-=1

**from** calendar **import** monthrange

    days\_in\_month=monthrange(dt\_now.year,dt\_now.month-1 **if** dt\_now.month>1 **else** 12)[1]

    day+=days\_in\_month

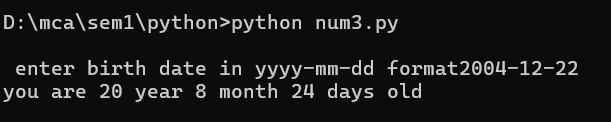
**if** (mon<0):

    yr-=1

    mon+=12

print(f"you are {yr} year {mon} month {day} days old")

Output:



4. Display trigonometric table of sin, cos and tan

# Display trigonometric table of sin, cos and tan

**import** math

print(f"{'angle':>6} | {'sin':>6} | {'cos':>6}| {'tan':>6}")

**for** angle **in** range(0,361,30):

    rad=math.radians(angle)

    sin\_val=math.sin(rad)

    cos\_val=math.cos(rad)

**if** math.isclose(cos\_val,0.0,*abs\_tol*=1e-9):

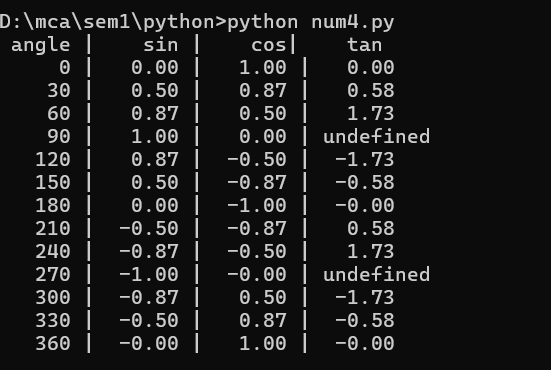
        tan\_val="undefined"

**else**:

        tan\_val=f"{(math.tan(rad)):.2f}"

    print(f"{angle:>6} | {sin\_val:>6.2f} | {cos\_val:>6.2f} | {tan\_val:>6}")

Output:



5. Generate 10 random numbers

#Generate 10 random numbers

**import** random

**for** i **in** range(1,10):

    print(random.randint(1,101),*end*="  ")

output  


6. Authentication: Ask username, password and compare

# Authentication: Ask username, password and compare

stored\_username="admin"

stored\_password="12345"

uname=input("enter username- ")

passwrd=input("enter password- ")

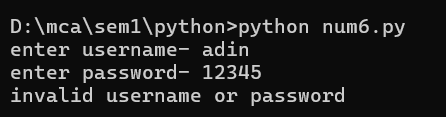
if(uname==stored\_username and passwrd==stored\_password):

    print("login successful")

else:

    print("invalid username or password")

Output:



7. Authentication: Ask username, password and compare with encryption

**from** cryptography.fernet **import** Fernet

key = Fernet.generate\_key()

cipher = Fernet(key)

stored\_username = "admin"

stored\_password\_encrypted = cipher.encrypt("12345".encode())

username = input("Enter username: ")

password = input("Enter password: ")

stored\_password = cipher.decrypt(stored\_password\_encrypted).decode()

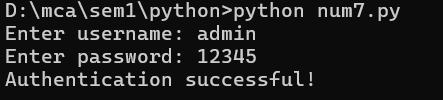
**if** username == stored\_username and password == stored\_password:

    print("Authentication successful!")

**else**:

    print("Authentication failed!")

Output:



8. Authentication: Ask username, password and compare with hashing

#Authentication: Ask username, password and compare with encryption

**import** hashlib

def *hash\_password*(*password*):

**return** hashlib.sha256(password.encode()).hexdigest()

stored\_username="admin"

stored\_password\_hash=hash\_password("12345")

uname=input("enter username- ")

passwrd=input("enter password- ")

passwrd\_hash=hash\_password(passwrd)

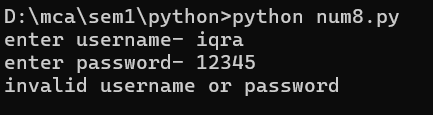
**if**(uname==stored\_username and passwrd\_hash==stored\_password\_hash):

    print("login successful")

**else**:

    print("invalid username or password")

Output:



9. Convert string "Hello$World" into Base64

# Convert string "Hello$World" into Base64

**import** base64

str="Hello$World"

str\_bytes=str.encode("utf-8")

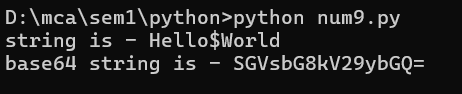
base64\_bytes=base64.b64encode(str\_bytes)

base64\_str=base64\_bytes.decode("utf-8")

print(f"string is - {str}")

print(f"base64 string is - {base64\_str}")

Output:



10. Write a program to create class ‘Search’ having data members (int a[ ], x) and

define member functions as void input(), void output(), void search(int position),void add(int value).

#include <iostream>

using namespace std;

class Search {

int \*a,x;

public:

Search(int n=10)

{

x=n;

a=new int[x];

}

~Search(){

delete[] a;

}

void input() {

cout <<"Enter "<<x<<" elements: ";

for (int i = 0; i < x; i++) {

cin>>a[i];

}

}

void display()

{

cout <<" elements: ";

for (int i = 0; i < x; i++) {

cout<<a[i]<<" ";

}

}

void search\_element(int val)

{

int index=-1;

for(int i=0;i<x;i++)

{

if(a[i]==val)

{

index=i;

}

}

if(index!=-1)

{

cout<<"element is found at "<<index<<" index";

}

else

{

cout<<"element not found";

}

}

void add(int val)

{

int \*tmp=new int[x+1];

for(int i=0;i<x;i++)

{

tmp[i]=a[i];

}

tmp[x]=val;

delete[] a;

a=tmp;

x++;

}

};

int main()

{

int i,n,ch,num;

cout <<"\n enter size- ";

cin>>n;

Search obj(n);

obj.input();

obj.display();

do

{

cout<<"\n press 1- to add";

cout<<"\n press 2- to search";

cout<<"\n press 3- to display";

cout<<"\n press 0- to exit";

cout<<"\n enter choice - ";

cin>>ch;

switch(ch)

{

case 0:

cout<<"exiting.....";

break;

case 1:

cout<<"\n enter element you wish to add- ";

cin>>num;

obj.add(num);

break;

case 2:

cout<<"\n enter element you wish to search- ";

cin>>num;

obj.search\_element(num);

break;

case 3:

obj.display();

break;

default:

cout<<"invalid choice";

break;

}

}

while(ch!=0);

}

Output:

