```python
# Program to multiply two matrices using nested loops

import numpy as np

X = []
C=int(input("Enter number column in matrix:"))
R=int(input("\n Enter number of rows in matrix:"))
for i in range(C):
    A=[]
    for j in range(R):
        A.append(int(input("Enter element of first matrix:")))
    X.append(A)
print("\nFirst Matrix is:")
M=np.matrix(X)
print(np.matrix(X))

Y = []
C1=int(input("Enter number column in matrix:"))
R1=int(input("\n Enter number of rows in matrix:"))
for i in range(C1):
    A1=[]
    for j in range(R1):
        A1.append(int(input("Enter element:")))
    Y.append(A1)
print("\n2nd  MAtrix is:")
print(np.matrix(Y))


result = []

for i in range(C1):
    A2=[]
    for j in range(R1):
        A2.append(int(0))
    result.append(A2)
print("\n3rd  MAtrix is:")
print(np.matrix(result))


# iterate through rows of X
for i in range(R):
   # iterate through columns of Y
   for j in range(C1):
       # iterate through rows of Y
       for k in range(R1):
           result[i][j] += X[i][k] * Y[k][j]

print("\n Matrix Multiplication")
print(np.matrix(result))


# Identity Program
import numpy as np
R=int(input("Enter no of rows or colums of matrix:"))
```

```
X=np.identity(R,dtype=int)
print(X)

Y = []
C1=int(input("Enter number column in matrix:"))
R1=int(input("\n Enter number of rows in matrix:"))
for i in range(C1):
    A1=[]
    for j in range(R1):
        A1.append(int(input("Enter element:")))
    Y.append(A1)
print("\n2nd  MAtrix is:")
print(np.matrix(Y))


result = []

for i in range(C1):
    A2=[]
    for j in range(R1):
        A2.append(int(0))
    result.append(A2)
print("\n3rd  MAtrix is:")
print(np.matrix(result))


# iterate through rows of X
for i in range(R):
   # iterate through columns of Y
   for j in range(C1):
       # iterate through rows of Y
       for k in range(R1):
           result[i][j] += X[i][k] * Y[k][j]

print("\n Matrix Multiplication")
print(np.matrix(result))

# Reflection Program
import numpy as np
R=int(input("Enter no of rows or colums of first matrix matrix:"))
X=np.identity(R,dtype=int)
print(np.matrix(X))
def Reflection_about_X():
    X[1][1]=X[1][1]*-1
def Reflection_about_Y():
    X[0][0]=X[0][0]*-1
def Reflection_about_Origin():
    X[1][1]=X[1][1]*-1
    X[0][0]=X[0][0]*-1
print("\n Press 1 for Reflection about X-axis,\n Press 2 for Reflection about Y-
axis,\nPress 3 for Reflection about Origin")
Choice=int(input("Enter your choice:"))

if(Choice==1):
```

```python
    Reflection_about_X()
elif(Choice==2):
    Reflection_about_Y()
elif(Choice==3):
    Reflection_about_Origin()
else:
    print("\nInvalid Selection...")


print(np.matrix(X))
Y = []
C1=int(input("Enter number column in matrix:"))
R1=int(input("\n Enter number of rows in matrix:"))
for i in range(C1):
    A1=[]
    for j in range(R1):
        A1.append(int(input("Enter element:")))
    Y.append(A1)
print("\n2nd  MAtrix is:")
print(np.matrix(Y))


result = []

for i in range(C1):
    A2=[]
    for j in range(R1):
        A2.append(int(0))
    result.append(A2)
print("\n3rd  MAtrix is:")
print(np.matrix(result))


# iterate through rows of X
for i in range(R):
   # iterate through columns of Y
   for j in range(C1):
       # iterate through rows of Y
       for k in range(R1):
           result[i][j] += X[i][k] * Y[k][j]

print("\n Matrix Multiplication")
print(np.matrix(result))

# Scaling Program
import numpy as np
R=int(input("Enter no of rows or colums of first matrix matrix:"))
X=np.identity(R,dtype=int)
print(np.matrix(X))
Sx=int(input("\nEnter scaling factor in X:"))
Sy=int(input("\nEnter scaling factor in Y:"))

X[0][0]=Sx
X[1][1]=Sy
```

```
print("\n Scaling Transformation matrix is:")
print(np.matrix(X))
Y = []
C1=int(input("Enter number column in matrix:"))
R1=int(input("\n Enter number of rows in matrix:"))
for i in range(C1):
    A1=[]
    for j in range(R1):
        A1.append(int(input("Enter element:")))
    Y.append(A1)
print("\n2nd  MAtrix is:")
print(np.matrix(Y))


result = []

for i in range(C1):
    A2=[]
    for j in range(R1):
        A2.append(int(0))
    result.append(A2)
print("\n3rd  MAtrix is:")
print(np.matrix(result))


# iterate through rows of X
for i in range(R):
   # iterate through columns of Y
   for j in range(C1):
       # iterate through rows of Y
       for k in range(R1):
           result[i][j] += X[i][k] * Y[k][j]

print("\n Matrix Multiplication")
print(np.matrix(result))


# Scaling Program
import numpy as np
R=int(input("Enter no of rows or colums of first matrix matrix:"))
X=np.identity(R,dtype=int)
print(np.matrix(X))
tx=int(input("\nEnter translating factor in X:"))
ty=int(input("\nEnter translating factor in Y:"))

X[0][2]=tx
X[1][2]=ty
print("\n Translation Transformation matrix is:")
print(np.matrix(X))
Y = []
C1=int(input("Enter number column in matrix:"))
R1=int(input("\n Enter number of rows in matrix:"))
for i in range(C1):
    A1=[]
```

```python
    for j in range(R1):
        A1.append(int(input("Enter element:")))
    Y.append(A1)
print("\n2nd  MAtrix is:")
print(np.matrix(Y))
'''
#Homogenous Matrix
I=np.identity(3,dtype=int)
I[0][0]=Y[0][0]
I[0][1]=Y[0][1]
I[1][0]=Y[1][0]
I[1][1]=Y[1][1]
I[2][0]=1
I[2][1]=1
print("\nHomogenous Matrix Is:")
print(np.matrix(I))
'''
result = []

for i in range(C1):
    A2=[]
    for j in range(R1):
        A2.append(int(0))
    result.append(A2)
print("\n3rd  MAtrix is:")
print(np.matrix(result))


# iterate through rows of X
for i in range(R):
   # iterate through columns of Y
   for j in range(C1):
       # iterate through rows of Y
       for k in range(R1):
           result[i][j] += X[i][k] * Y[k][j]

print("\n Matrix Multiplication")
print(np.matrix(result))

# Shearing  Program
import numpy as np
R=int(input("Enter no of rows or colums of first matrix matrix:"))
X=np.identity(R,dtype=int)
print(np.matrix(X))
Shx=int(input("\nEnter Shearing factor in X:"))
Shy=int(input("\nEnter Shearing factor in Y:"))

X[0][1]=Shx
X[1][0]=Shy
print("\n Shearing Transformation matrix is:")
print(np.matrix(X))
Y = []
C1=int(input("Enter number column in matrix:"))
R1=int(input("\n Enter number of rows in matrix:"))
```

```
for i in range(C1):
    A1=[]
    for j in range(R1):
        A1.append(int(input("Enter element:")))
    Y.append(A1)
print("\n2nd  MAtrix is:")
print(np.matrix(Y))

result = []

for i in range(C1):
    A2=[]
    for j in range(R1):
        A2.append(int(0))
    result.append(A2)
print("\n3rd  MAtrix is:")
print(np.matrix(result))


# iterate through rows of X
for i in range(R):
   # iterate through columns of Y
   for j in range(C1):
       # iterate through rows of Y
       for k in range(R1):
           result[i][j] += X[i][k] * Y[k][j]

print("\n Matrix Multiplication")
print(np.matrix(result))

# Rotation  Program
import numpy as np
import math
R=int(input("Enter no of rows or colums of first matrix matrix:"))
X=np.identity(R,dtype=int)
print(np.matrix(X))
Degree=int(input("Enter degree of rotation:"))
print("\nEnter 1 for clockwise direction:\nEnter 2 for Counter clockwise direction:")
CCD=int(input("Enter your choice:"))
if(CCD==1):
    X[0][0]=math.cos(math.radians(Degree))
    X[0][1]=math.sin(math.radians(Degree))
    X[1][0]=-(math.sin(math.radians(Degree)))
    X[1][1]=math.cos(math.radians(Degree))

elif(CCD==2):
    X[0][0]=math.cos(math.radians(Degree))
    X[0][1]=-(math.sin(math.radians(Degree)))
    X[1][0]=math.sin(math.radians(Degree))
    X[1][1]=math.cos(math.radians(Degree))

print("\n Rotation Transformation matrix is:")
print(np.matrix(X))
Y = []
```

```
C1=int(input("Enter number column in matrix:"))
R1=int(input("\n Enter number of rows in matrix:"))
for i in range(C1):
    A1=[]
    for j in range(R1):
        A1.append(int(input("Enter element:")))
    Y.append(A1)
print("\n2nd  MAtrix is:")
print(np.matrix(Y))

result = []

for i in range(C1):
    A2=[]
    for j in range(R1):
        A2.append(int(0))
    result.append(A2)
print("\n3rd  MAtrix is:")
print(np.matrix(result))


# iterate through rows of X
for i in range(R):
   # iterate through columns of Y
   for j in range(C1):
       # iterate through rows of Y
       for k in range(R1):
           result[i][j] += X[i][k] * Y[k][j]

print("\n Matrix Multiplication")
print(np.matrix(result))
```