ML-Assignment 1

Name: *Swayam Swarup Jethi*

Section: *2C1*

Registration Number: *2341010085*

Serial Number: *31*

Date of Submission: *23/09/2025*

1. Take marks as input (0–100). Use if-elif-else to print the grade:

   90–100 → A

   75–89 → B

   50–74 → C

   Below 50 → Fail

In [1]:

```
mark=int(input("Enter your marks:"))
print(f"Marks: {mark}")
if mark<=100 and mark>=90:
    print("Grade: A")
elif mark<=89 and mark>=75:
    print("Grade: B")
elif mark<=74 and mark>=50:
    print("Grade: C")
else:
    print("Fail")
```

```
Marks: 78
Grade: B
```

2. Accept a number from the user and use a while loop to find the sum of its digits.

In [2]:

```
num=int(input("Enter a number:"))
sum=0
temp=num
while num>0:
    d=num%10
    sum+=d
    num=num//10
print(f"Number: {temp}\nSum={sum}")
```

```
Number: 1234
Sum=10
```

3. Write a function is_prime(n) that returns True if a number is prime, otherwise False.Test it for multiple values

In [3]:

```
def is_prime(n):
    f=0
    for i in range(1,n+1):
        if n%i==0:
            f=f+1
    return f==2
```

```
print(is_prime(17))
```

```
True
```

4. Define a function is_palindrome(word) that checks whether a string is a palindrome or not. Example: "madam" → Palindrome.

In [4]:

```
def is_palindrome(s: str):
    s=s.lower()
    return s==s[ : :-1]
print(is_palindrome("madam"))
```

```
True
```

5. Create two 3×3 matrices and perform:

a) Matrix multiplication

b) Determinant of a matrix

c) Inverse of a matrix (if possible)

In [5]:

```
import numpy as np
X=np.array([[1,0,0],
            [3,3,0],
            [5,2,-1]])

Y=np.array([[3,2,1],
            [3,2,1],
            [3,2,1]])
#Matrix Multiplication :
print("Multiplication: ")
Z=np.dot(X,Y)
print(Z)
#Determinant Of A Matrix :
det_X=np.linalg.det(X)
print("Determinant of X:",det_X)
det_Y=np.linalg.det(Y)
print("Determinant of Y:",det_Y)
#Inverse Of A Matrix(If Exists) :
print("Original Mat: \n",X)
if np.isclose(det_X,0):
    print("Matrix is Singular.Inverse is not possible")
else:
    inv_X=np.linalg.inv(X)
    print("Inverse: \n",inv_X)
    print("Verifying: \n",np.round(np.dot(inv_X,X),decimals=8))
```

```
Multiplication:
[[ 3  2  1]
 [18 12  6]
 [18 12  6]]
Determinant of X: -2.9999999999999996
Determinant of Y: 0.0
Original Mat:
 [[ 1  0  0]
 [ 3  3  0]
 [ 5  2 -1]]
Inverse:
 [[ 1.00000000e+00  2.22044605e-17  0.00000000e+00]
 [-1.00000000e+00  3.33333333e-01  0.00000000e+00]
 [ 3.00000000e+00  6.66666667e-01 -1.00000000e+00]]
Verifying:
 [[ 1.  0.  0.]
 [-0.  1.  0.]
 [ 0.  0.  1.]]
```

6. Generate two NumPy arrays of 10 random integers each. Perform element-wise
comparison (>, <, ==). Count how many times values in the first array are greater than the second.

In [6]:

```
import numpy as np
import random
arr1=[]
arr2=[]
for i in range(10):
    arr1.append(random.randint(1,20))
    arr2.append(random.randint(1,20))
arr1=np.array(arr1)
arr2=np.array(arr2)
count=0
for i in range(10):
    if arr1[i]>arr2[i]:
        count=count+1
print(count)
```

```
6
```

In [7]:

```
import random
arr1=np.random.randint(0, 10, size=10)
arr2=np.random.randint(0,10,size=10)
print("arr1= ",arr1,"\narr2= ",arr2)
greater_than=arr1>arr2
g=np.sum(greater_than)
print(g)
```

```
arr1=  [2 9 9 4 5 4 0 1 1 0]
arr2=  [5 9 3 8 0 4 4 8 9 1]
2
```

7. Suppose you have exam marks of 10 students stored in a Python list.
Convert it into a NumPy array and compute:

a) Average marks of the class

b) Students who scored above average

In [8]:

```
import numpy as np
marks=[70,80,50,90,78,57,39,78,58,49]
marks=np.array(marks)
mean=np.mean(marks)
above_mean=marks[marks>mean]
print("Mean: ",mean)
print("Marks above average:", above_mean)
```

```
Mean:  64.9
Marks above average: [70 80 90 78 78]
```

8. Create a 4×4 NumPy array with values from 1 to 16. Perform the following:

a) Extract the second row

b) Extract the third column

c) Slice the sub-matrix of shape (2×2) from the bottom-right corner

In [9]:

```
import numpy as np
arr=np.arange(1,17).reshape(4,4)
print("Array:\n",arr)
sec_row=arr[1]
print("\nSecond row:",sec_row)
third_column=arr[:,2]
print("\nThird column:",third_column)
sliced_matrix = arr[2:,2:]
print("\n2x2 bottom-right sub-matrix:\n", sliced_matrix)
```

```
Array:
 [[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]
 [13 14 15 16]]

Second row: [5 6 7 8]

Third column: [ 3  7 11 15]

2x2 bottom-right sub-matrix:
 [[11 12]
 [15 16]]
```

9. Create a (3×4) array with consecutive numbers from 1 to 12. Demonstrate the difference between reshape() and ravel().

In [10]:

```
import numpy as np

arr=np.arange(1,13).reshape(3,4)
print("Array:\n", arr)
reshaped = arr.reshape(4, 3)
print("Reshaped array (4x3):\n",reshaped)
raveled=arr.ravel()
print("Raveled array:\n",raveled)
```

```
Array:
 [[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]
Reshaped array (4x3):
 [[ 1  2  3]
 [ 4  5  6]
 [ 7  8  9]
 [10 11 12]]
Raveled array:
 [ 1  2  3  4  5  6  7  8  9 10 11 12]
```

10. Simulate rolling a six-sided dice 1,000 times using NumPy’s random.randint(). Count the frequency of each outcome and display it as a dictionary (face → count).

In [11]:

```
import numpy as np
r=np.random.randint(1,7,1000)
counts = {}
for i in range(1,7):
    counts[i] = list(r).count(i)
print(counts)
```

```
{1: 147, 2: 179, 3: 193, 4: 166, 5: 162, 6: 153}
```