Day 3 Python programming

1)Armstrong number

def is\_armstrong\_number(num):

digits = [int(digit) for digit in str(num)]

power = len(digits)

sum\_of\_powers = sum(digit \*\* power for digit in digits)

return sum\_of\_powers == num

number = 153

if is\_armstrong\_number(number):

print(f"{number} is an Armstrong number.")

else:

print(f"{number} is not an Armstrong number.")

2)Happy number

def is\_happy\_number(num):

def get\_sum\_of\_squares(n):

return sum(int(digit) \*\* 2 for digit in str(n))

seen = set()

while num != 1 and num not in seen:

seen.add(num)

num = get\_sum\_of\_squares(num)

return num == 1

number = 19

if is\_happy\_number(number):

print(f"{number} is a happy number.")

else:

print(f"{number} is not a happy number.")

3) Simple interest

def calculate\_simple\_interest(principal, rate, time):

return (principal \* rate \* time) / 100

principal\_amount=1000

annual\_rate=5

time\_years=2

interest = calculate\_simple\_interest(principal\_amount, annual\_rate, time\_years)

print(f"Principal: ${principal\_amount}")

print(f"Rate: {annual\_rate}% per annum")

print(f"Time: {time\_years} years")

print(f"Simple Interest: ${interest}")

4)Factorial and print 1st n factorial

def factorial(n):

if n == 0:

return 1

else:

return n \* factorial(n - 1)

def print\_first\_n\_factorials(n):

for i in range(1, n + 1):

print(f"Factorial of {i} is {factorial(i)}")

n = 5

print\_first\_n\_factorials(n)

5) Square and Cube of the first decimal number

def square(x):

return x \*\* 2

def cube(x):

return x \*\* 3

def print\_squares\_and\_cubes(n):

for i in range(1, n + 1):

decimal\_number = float(i)

print(f"Decimal Number: {decimal\_number}, Square: {square(decimal\_number)}, Cube: {cube(decimal\_number)}")

n = 5

print\_squares\_and\_cubes(n)

6)Binary to decimal and Octal

def binary\_to\_decimal(binary\_str):

return int(binary\_str, 2)

def binary\_to\_octal(binary\_str):

decimal = binary\_to\_decimal(binary\_str)

return oct(decimal)[2:] # Remove '0o' prefix from octal representation

binary\_number = "1101" # Example binary number

decimal\_number = binary\_to\_decimal(binary\_number)

octal\_number = binary\_to\_octal(binary\_number)

print(f"Binary: {binary\_number}")

print(f"Decimal: {decimal\_number}")

print(f"Octal: {octal\_number}")

7)Add Binary Numbers

def add\_binary(a, b):

num1 = int(a, 2)

num2 = int(b, 2)

sum\_num = num1 + num2

return bin(sum\_num)[2:]

a = "11"

b = "1"

result = add\_binary(a, b)

print(f"Sum of binary strings '{a}' and '{b}' is '{result}'")

8)Find the greatest number of three binary digits

def binary\_to\_decimal(binary\_str):

return int(binary\_str, 2)

largest\_binary = "111"

largest\_decimal = binary\_to\_decimal(largest\_binary)

print(f"The largest number with three binary digits is {largest\_binary} (binary) or {largest\_decimal} (decimal).")

9)Matrix multiplication

def matrix\_multiply(A, B):

rows\_A, cols\_A = len(A), len(A[0])

rows\_B, cols\_B = len(B), len(B[0])

if cols\_A != rows\_B:

raise ValueError("Number of columns in A must be equal to number of rows in B")

# Initialize result matrix with zeros

result = [[0] \* cols\_B for \_ in range(rows\_A)]

for i in range(rows\_A):

for j in range(cols\_B):

for k in range(cols\_A): # or k in range(rows\_B)

result[i][j] += A[i][k] \* B[k][j]

return result

A = [

[1, 2],

[3, 4]

]

B = [

[5, 6],

[7, 8]

]

result = matrix\_multiply(A, B)

print("Matrix A:")

for row in A:

print(row)

print("\nMatrix B:")

for row in B:

print(row)

print("\nResult of A \* B:")

for row in result:

print(row)

10)Matrix addition

def matrix\_addition(A, B):

rows\_A, cols\_A = len(A), len(A[0])

rows\_B, cols\_B = len(B), len(B[0])

if rows\_A != rows\_B or cols\_A != cols\_B:

raise ValueError("Matrices must have the same dimensions for addition")

result = [[0] \* cols\_A for \_ in range(rows\_A)]

for i in range(rows\_A):

for j in range(cols\_A):

result[i][j] = A[i][j] + B[i][j]

return result

A = [

[1, 2, 3],

[4, 5, 6]

]

B = [

[7, 8, 9],

[10, 11, 12]

]

result = matrix\_addition(A, B)

print("Matrix A:")

for row in A:

print(row)

print("\nMatrix B:")

for row in B:

print(row)

print("\nResult of A + B:")

for row in result:

print(row)