

## ASSSIGNMENT - 1

**Name: KHUSHI JAIN**

**Enrollment:0176CD231081**

**Batch: 5**

**Batch Time: 10:30 A.M. TO 12:10 P.M.**

### **BASIC IF-ELSE PROBLEM:**

**Q1.** Write a program to check whether a number is positive, negative, or zero.

**Sol:**

```
num=int(input("enter a number:"))
```

```
if num < 0:
```

```
    print("number is negative")
```

```
elif num == 0:
```

```
    print("number is zero")
```

```
else:
```

```
    print("number is positive")
```

**output:** enter a number: 4

number is positive

**Q2.** Write a program to check whether a number is even or odd.

**Sol:**

```
num=int(input("enter a number:"))
```

```
if num % 2 ==0:
```

```
    print("number is even")
```

```
else:
```

```
    print("number is odd")
```

**output:** enter a number: 43

number is odd

**Q3.** Write a program to check if a given year is a leap year or not.

**Sol:**

```
year = int(input("Enter a year to check: "))
```

```
if year % 400 == 0:
```

```
    print(f"{year} is a leap year.")
```

```
elif year % 100 == 0:
```

```
    print(f"{year} is not a leap year.")
```

```
elif year % 4 == 0:
```

```
    print(f"{year} is a leap year.")
```

```
else:
```

```
    print(f"{year} is not a leap year.")
```

**output:** Enter a year to check: 2024

2024 is a leap year.

**Q4.** Write a program to find the greatest of two numbers.

**Sol:**

```
num1 = float(input("Enter the first number: "))
```

```
num2 = float(input("Enter the second number: "))
```

```
if num1 > num2:
```

```
    print(f"The greatest number is: {num1}")
```

```
elif num2 > num1:
```

```
    print(f"The greatest number is: {num2}")
```

```
else:
```

```
    print("Both numbers are equal.")
```

**output: Enter the first number: 5.7**

**Enter the second number: 7.9**

**The greatest number is: 7.9**

**Q5.** Write a program to check whether a person is eligible to vote (age >= 18).

**Sol:**

```
age = int(input("Enter your age: "))
```

```
if age >= 18:
```

```
    print("You are eligible to vote.")
```

```
else:
```

```
    print("You are not eligible to vote yet.")
```

**output:**

Enter your age: 20

You are eligible to vote.

**Q6.** Write a program to check whether a given character is a vowel or consonant.

**Sol:**

```
char = input("Enter a single character: ")
```

```
if len(char) == 1 and char.isalpha():
```

```
    char_lower = char.lower()
```

```
if char_lower in ['a', 'e', 'i', 'o', 'u']:
```

```
    print("The character '{0}' is a vowel.".format(char))
```

```
else:
```

```
    print("The character '{0}' is a consonant.".format(char))
```

```
else:
```

```
    print("Invalid input. Please enter a single alphabet character.")
```

**output: Enter a single character: f**

**The character 'f' is a consonant.**

Q7. Write a program to check if a number is divisible by 5.

Sol:

```
number = int(input("Enter an integer: "))
```

```
if number % 5 == 0:
```

```
    print(f"The number {number} is divisible by 5.")
```

```
else:
```

```
    print(f"The number {number} is not divisible by 5.")
```

output: Enter an integer: 65

The number 65 is divisible by 5.

Q8. Write a program to determine whether a given number is a single-digit, two-digit, or more than two-digit number.

Sol:

```
number = int(input("Enter an integer: "))
```

```
if number >= 0 and number <= 9:
```

```
    print(f"The number {number} is a single-digit number.")
```

```
elif number >= 10 and number <= 99:
```

```
    print(f"The number {number} is a two-digit number.")
```

```
else:
```

```
    print(f"The number {number} is a number with more than two digits.")
```

output: Enter an integer: 44

The number 44 is a two-digit number.

Q9. Write a program to check whether a student has passed or failed (passing marks = 40).

Sol:

```
PASSING_MARKS = 40
```

```
marks = int(input("Enter the student's marks: "))
```

```
if marks >= PASSING_MARKS:
```

```
    print("The student has passed.")
```

```
else:
```

```
    print("The student has failed.")
```

output:

Enter the student's marks: 34

The student has failed.

### **Nested loop-**

Q1. Write a program to find the greatest among three numbers.

Sol:

```
print("--- Question 1: Greatest of Three Numbers ---")
```

```
num1 = float(input("Enter the first number: "))
```

```
num2 = float(input("Enter the second number: "))
```

```
num3 = float(input("Enter the third number: "))
```

```
if num1 >= num2 and num1 >= num3:
```

```
    greatest = num1
```

```
elif num2 >= num1 and num2 >= num3:
```

```
    greatest = num2
```

```
else:
```

```
    greatest = num3
```

```
print(f"The greatest number is: {greatest}\n")
```

output:

Enter the first number: 50

Enter the second number: 100

Enter the third number: 75

The greatest number is: 100.0

Q2. Write a program to classify a person based on age.

Sol:

```
print("--- Question 2: Age Classification ---")
```

```
age = int(input("Enter age: "))
```

```
if age >= 60:
```

```
    print("Senior (60+)")
```

```
elif age >= 20:
```

```
    print("Adult (20-59)")
```

```
elif age >= 13:
```

```
    print("Teenager (13-19)")
```

```
else:
```

```
    print("Child (<13)")
```

```
print()
```

output:

Enter age: 15

Teenager (13-19)

Q3. Write a program to assign grades based on marks.

Sol;

```
print("--- Question 3: Grade Assignment ---")
```

```
marks = float(input("Enter marks (out of 100): "))
```

```
if marks >= 90:
```

```
    grade = 'A'
```

```
elif marks >= 75:
```

```
    grade = 'B'
```

```
elif marks >= 50:
```

```
    grade = 'C'
```

```
elif marks >= 35:
```

```
    grade = 'D'
```

```
else:
```

```
    grade = 'Fail'
```

```
print(f"The grade is: {grade}\n")
```

output:

Enter marks (out of 100): 85

The grade is: B

Q 4. Write a program to check the type of triangle (equilateral, isosceles, or scalene) based on sides.

Sol:

```
print("--- Question 4: Triangle Type Checker ---")
```

```
side1 = float(input("Enter length of side 1: "))
```

```
side2 = float(input("Enter length of side 2: "))
```

```
side3 = float(input("Enter length of side 3: "))
```

```
if (side1 + side2 > side3) and (side1 + side3 > side2) and (side2 + side3 > side1):
```

```
    if side1 == side2 == side3:
```

```
        print("The triangle is Equilateral.")
```

```
    elif side1 == side2 or side1 == side3 or side2 == side3:
```

```
        print("The triangle is Isosceles.")
```

```
    else:
```

```
        print("The triangle is Scalene.")
```

```
else:
```

```
    print("The given side lengths do not form a valid triangle.")
```

```
print()
```

output:

Enter length of side 1: 3

Enter length of side 2: 4

Enter length of side 3: 5

The triangle is Scalene.

Q5. Write a program to check if a character is uppercase, lowercase, digit, or special symbol.

Sol:

```
print("--- Question 5: Character Type Checker ---")
```

```
char = input("Enter a single character: ")
```

```
if len(char) == 1:
```

```
    if 'A' <= char <= 'Z':
```

```
        print(f"'{char}' is an uppercase character.")
```

```
    elif 'a' <= char <= 'z':
```

```
        print(f"'{char}' is a lowercase character.")
```

```
    elif '0' <= char <= '9':
```

```
        print(f"'{char}' is a digit.")
```

```
    else:
```

```
        print(f"'{char}' is a special symbol.")
```

```
else:
```

```
    print("Please enter a single character.")
```

```
print()
```

output:

Enter a single character: a

'a' is a lowercase character.

Q 6. Write a program to check if a given year is a century year and also a leap year.

Sol:

```
year = int(input("Enter a year to check: "))
```

```
is_century = False
```

```
is_leap = False
```

```
if year % 100 == 0:
```

```
    is_century = True
```



```
    if year % 400 == 0:
        is_leap = True
else:
    if year % 4 == 0:
        is_leap = True
if is_century and is_leap:
    print(f"The year {year} is a century year and a leap year.")
elif is_century and not is_leap:
    print(f"The year {year} is a century year but not a leap year.")
elif is_leap:
    print(f"The year {year} is a leap year but not a century year.")
else:
    print(f"The year {year} is neither a century year nor a leap year.")
print()
output:
Enter a year to check: 2024
The year 2024 is a leap year but not a cent
```

Q7. Write a program to determine the largest of four numbers using nested if.

Sol:

```
n1 = float(input("Enter first number: "))
n2 = float(input("Enter second number: "))
n3 = float(input("Enter third number: "))
n4 = float(input("Enter fourth number: "))
```

```
if n1 > n2:
    if n1 > n3:
        if n1 > n4:
            largest = n1
        else:
```

```

        largest = n4
    else:
        if n3 > n4:
            largest = n3
        else:
            largest = n4
    else:
        if n2 > n3:
            if n2 > n4:
                largest = n2
            else:
                largest = n4
        else:
            if n3 > n4:
                largest = n3
            else:
                largest = n4

print(f"The largest number is: {largest}\n")

```

output:

Enter first number: 10

Enter second number: 50

Enter third number: 30

Enter fourth number: 20

The largest number is: 50.0

Q8. Write a program to calculate electricity bill based on units.

```
units = float(input("Enter number of units consumed: "))
```

```
bill = 0
```

```
if units <= 100:
```

```
bill = units * 5
elif units <= 200:
    bill = (100 * 5) + ((units - 100) * 10)
else:
    bill = (100 * 5) + (100 * 10) + ((units - 200) * 15)
```

```
print(f"Total electricity bill: Rs. {bill:.2f}\n")
```

output:

Enter number of units consumed: 150

Total electricity bill: Rs. 1000.00

Q9. Write a program to classify BMI value.

Sol:

```
weight = float(input("Enter weight in kg: "))
height = float(input("Enter height in meters: "))
bmi = weight / (height ** 2)
```

```
if bmi < 18.5:
    category = "Underweight"
elif 18.5 <= bmi < 24.9:
    category = "Normal"
elif 25 <= bmi < 29.9:
    category = "Overweight"
else:
    category = "Obese"
```

```
print(f"Your BMI is {bmi:.2f}, which is classified as: {category}\n")
```

output:

Enter weight in kg: 70

Enter height in meters: 1.75

Your BMI is 22.86, which is classified as:

Q 10. Write a program to display the smallest number among three using nested if.

Sol:

```
n1 = float(input("Enter the first number: "))
```

```
n2 = float(input("Enter the second number: "))
```

```
n3 = float(input("Enter the third number: "))
```

```
if n1 <= n2:
```

```
    if n1 <= n3:
```

```
        smallest = n1
```

```
    else:
```

```
        smallest = n3
```

```
else:
```

```
    if n2 <= n3:
```

```
        smallest = n2
```

```
    else:
```

```
        smallest = n3
```

```
print(f"The smallest number is: {smallest}\n")
```

output:

Enter the first number: 10

Enter the second number: 5

Enter the third number: 15

The smallest number is: 5.0

### **LOOP-BASED PROBLEMS:**

**Q1.** Write a program using a for loop to print all Armstrong numbers between 100 and 999.

(Armstrong number: sum of cubes of digits equals the number itself. Example: 153 =>

13+53+33 = 153).

**SOL:**

```
def is_armstrong(n):  
    temp = n  
    sum_cubes = 0  
    while temp > 0:  
        digit = temp % 10  
        sum_cubes += digit ** 3  
        temp //= 10  
    return sum_cubes == n  
  
print("Armstrong numbers between 100 and 999:")  
for i in range(100, 1000):  
    if is_armstrong(i):  
        print(i)
```

**OUTPUT:**

Armstrong numbers between 100 and 999: 153 370 371 407

Q2. Write a program to generate and display the first n prime numbers using a for loop.

**SOL:**

```
def is_prime(n):  
    if n <= 1:  
        return False  
    for i in range(2, int(n**0.5) + 1):  
        if n % i == 0:  
            return False  
    return True  
  
def generate_n_primes(n):  
    primes_found = 0
```

```

num = 2
while primes_found < n:
    if is_prime(num):
        print(num, end=" ")
        primes_found += 1
    num += 1

```

```

n = int(input("Enter the number of prime numbers to generate: "))
print(f"The first {n} prime numbers are:")
generate_n_primes(n)
print()

```

OUTPUT:

Enter the number of prime numbers to generate: 10

The first 10 prime numbers are: 2 3 5 7 11 13 17 19 23 29

Q3. Write a program to display all numbers from 1 to 500 that are divisible by 3, but the sum of their digits should not exceed 10.

SOL:

```

def sum_digits(n):
    s = 0
    while n > 0:
        s += n % 10
        n //= 10
    return s

print("Numbers from 1 to 500 divisible by 3 with sum of digits not exceeding 10:")
for i in range(1, 501):
    if i % 3 == 0 and sum_digits(i) <= 10:
        print(i, end=" ")
print()

```

OUTPUT:

Numbers from 1 to 500 divisible by 3 with sum of digits not exceeding 10:

3 6 9 12 15 18 21 24 27 30 33 36 42 45 51 54 60 63 72 81 90 102 105 108 111 114 117 120 123 126  
132 135 141 144 150 153 162 171 180 201 204 207 210 213 216 222 225 231 234 240 243 252 261  
270 300 303 306 312 315 321 324 330 333 342 351 360 402 405 411 414 420 423 432 441 450

Q4. Write a program using a for loop to print a pyramid of stars (\*) of height n. Example for n=4:

```
*  
  
***  
  
*****  
  
*****
```

SOL:

```
def print_star_pyramid(n):  
    for i in range(n):  
        print(" " * (n - i - 1), end="")  
        print("*" * (2 * i + 1))  
  
n = int(input("Enter the height of the pyramid: "))  
print_star_pyramid(n)
```

Q5. Write a program to accept a string and check whether it is a pangram (contains all 26 alphabets at least once) using a for loop.

SOL:

```
import string  
  
def is_pangram(s):  
    unique_chars = set(s.lower())  
    alphabet_set = set(string.ascii_lowercase)  
    return alphabet_set.issubset(unique_chars)  
  
s = input("Enter a string: ")  
if is_pangram(s):  
    print("The string is a pangram.")  
else:  
    print("The string is not a pangram.")
```

OUTPUT:

Enter a string: The quick brown fox jumps over the lazy dog The string is a pangram.

Q6. Write a program using a for loop to print all twin primes between 1 and 100. (Twin primes: pairs of prime numbers with a difference of 2, e.g., (3,5), (11,13)).

SOL:

```
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

print("Twin primes between 1 and 100:")
for i in range(1, 99):
    if is_prime(i) and is_prime(i + 2):
        print(f"({i}, {i + 2})")
```

OUTPUT:

Twin primes between 1 and 100: (3, 5) (5, 7) (11, 13) (17, 19) (29, 31) (41, 43) (59, 61) (71, 73)

Q7. Write a program that accepts a number from the user and prints whether it is a Harshad number (number divisible by the sum of its digits) using a for loop.

SOL:

```
def sum_digits(n):
    s = 0
    temp = abs(n)
    while temp > 0:
        s += temp % 10
        temp //= 10
    return s

def is_harshad(n):
    if n <= 0:
```



```

        return False

    digit_sum = sum_digits(n)

    return n % digit_sum == 0

num = int(input("Enter a number to check if it's a Harshad number: "))

if is_harshad(num):

    print(f"{num} is a Harshad number.")

else:

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

```

OUTPUT:

Enter a number to check if it's a Harshad number: 18 18 is a Harshad number.

Q8. Write a program to generate Pascal's Triangle up to n rows using a for loop.

SOL:

```

def print_pascals_triangle(rows):

    row_list = []

    for i in range(rows):

        temp_list = []

        for j in range(i + 1):

            if j == 0 or j == i:

                temp_list.append(1)

            else:

                temp_list.append(row_list[i - 1][j - 1] + row_list[i - 1][j])

        row_list.append(temp_list)

    max_width = len(" ".join(map(str, row_list[-1])))

    for row in row_list:

        row_str = " ".join(map(str, row))

        print(row_str.center(max_width))

n = int(input("Enter the number of rows for Pascal's Triangle: "))

print_pascals_triangle(n)

```

OUTPUT:

Enter the number of rows for Pascal's Triangle: 6

1  
1 1  
1 2 1  
1 3 3 1  
1 4 6 4 1  
1 5 10 10 5 1

Q9. Write a program using a for loop to display the sum of the series:

$1^2 + 2^2 + 3^2 + \dots + n^2$

SOL:

```
def sum_of_squares(n):  
    total_sum = 0  
    series_str = ""  
    for i in range(1, n + 1):  
        total_sum += i ** 2  
        series_str += f"{i}^2"  
        if i < n:  
            series_str += " + "  
    print(f"The series is: {series_str}")  
    return total_sum  
  
n = int(input("Enter the value of n: "))  
result = sum_of_squares(n)  
print(f"The sum of the series is: {result}")
```

OUTPUT:

Enter the value of n: 6

The series is:  $1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2$

The sum of the series is: 91

Q10. Write a program that accepts a number from the user and prints whether it is a Strong number (sum of factorials of digits = number itself) using a for loop. Example:  $145 \Rightarrow 1! + 4! + 5! = 145$ .

SOL:

```

import math

def is_strong_number(n):
    if n <= 0:
        return False

    temp = n
    sum_factorials = 0
    while temp > 0:
        digit = temp % 10
        sum_factorials += math.factorial(digit)
        temp //= 10
    return sum_factorials == n

num = int(input("Enter a number to check if it's a Strong number: "))
if is_strong_number(num):
    print(f"{num} is a Strong number.")
else:
    print(f"{num} is not a Strong number.")

```

OUTPUT:

Enter a number to check if it's a Strong number: 67676

67676 is not a Strong number.

### **WHILE-LOOP PROBLEM:**

Q11. Write a program using a while loop to find the reverse of a number and check if the reversed number is prime. Example: Input = 73 → Reverse = 37 → Prime.

SOL:

```

def is_prime(n):
    if n < 2:
        return False

    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:

```

```

        return False
    return True

def reverse_and_check_prime():
    num = int(input("Enter a number: "))
    original_num = num
    reversed_num = 0
    while num > 0:
        digit = num % 10
        reversed_num = reversed_num * 10 + digit
        num //= 10
    print(f"Reverse of {original_num} is {reversed_num}.")
    if is_prime(reversed_num):
        print(f"{reversed_num} is a prime number.")
    else:
        print(f"{reversed_num} is not a prime number.")
reverse_and_check_prime()

```

OUTPUT;

Enter a number: 34

Reverse of 34 is 43.

43 is a prime number

Q12. Write a program that continues to accept numbers from the user until the sum of digits of all numbers entered becomes greater than 100.

SOL:

```

def sum_until_100():
    total_sum = 0
    while total_sum <= 100:
        num = int(input(f"Current sum is {total_sum}. Enter a number: "))
        total_sum += num

```

```
    print(f"The final sum ({total_sum}) is now greater than 100.")
sum_until_100()
```

OUTPUT:

Current sum is 0. Enter a number: 4

Current sum is 4. Enter a number: 7

Current sum is 11. Enter a number: 6

The final sum (105) is now greater than 100.

Q13. Write a program using a while loop to check whether a number is a Duck number (a number containing zero but not starting with zero, e.g., 202, 1203).

SOL:

```
def is_duck_number():
    num_str = input("Enter a number: ")
    if num_str.startswith('0'):
        print(f"{num_str} is not a Duck number (cannot start with 0).")
        return
    if '0' in num_str:
        print(f"{num_str} is a Duck number.")
    else:
        print(f"{num_str} is not a Duck number.")
is_duck_number()
```

OUTPUT:

Enter a number: 203

203 is a Duck number.

Q14. Write a program using a while loop to accept a number and check if it is a Happy number. (A number is happy if repeatedly replacing it with the sum of squares of its digits eventually reaches 1). Example: 19 is a happy number.

SOL;

```
def is_happy_number():
```

```
num = int(input("Enter a number: "))
seen_numbers = set()
while num != 1 and num not in seen_numbers:
    seen_numbers.add(num)
    sum_of_squares = 0
    temp = num
    while temp > 0:
        digit = temp % 10
        sum_of_squares += digit ** 2
        temp //= 10
    num = sum_of_squares
if num == 1:
    print("The number is a Happy number.")
else:
    print("The number is not a Happy number.")
is_happy_number()
```

OUTPUT:

Enter a number: 19

The number is a Happy number.