Hands-On Activities – Milestone 4

Activity 1:

Design an algorithm to accept a string from the user and print all duplicate character (which occur more than once) and their count. Characters which appear only once in the string must not be displayed. Also if there are no duplicate characters in the given string a message "**No duplicate characters**" must be printed.

For example if given String is "**Programming**" then your output should print
g: 2

r: 2

m: 2

Activity 2:

The product of two numbers is 120 and the sum of their squares is 289. Design an algorithm to find the sum of the two numbers.

Activity 3: *Magic Squares Puzzle*

Design an algorithm which finds the magic square for a given odd number and displays it.
A magic square is an arrangement of numbers from 1 to n^2 in an [n x n] matrix, with each number occurring exactly once, and such that the sum of the entries of any row, any column, or any main diagonal is the same.

For Example –

If the value of n is 5

The output must be as below:

17 24 1 8 15
23 5 7 14 16
4 6 13 20 22
10 12 19 21 3
11 18 25 2 9

If the value on n is: 6

The output must be as below:

Magic square works for an odd numbered size

Activity 4:

Design an algorithm to accept a string from the user. The algorithm must then check if the string has 2 characters which are consecutive and are vowels. If so the message "**consecutive vowels present**" must be printed, else the message "**no consecutive vowels**" must be printed.

For Example –

If the input string is "**team**"

Output must be "**consecutive vowels present**" as characters 'e' and 'a' are consecutive in the string and are vowels.

If the input string is "**the**"

Output must be "**no consecutive vowels**" as there are no consecutive characters which are vowels in the given string.

Activity 5:

Design an algorithm which accepts 2 numbers in binary format as input and prints their binary sum.

For Example –
Enter binary number1: 0110
Enter binary number2: 1011

The output must be **10001** (which is the sum of 0110 & 1011)

Activity 6:

Design an algorithm which accepts a string as input. The entered string must be alpha numeric. The algorithm must search for the numbers present in the entered string and display squares of those numbers as the output.

For Example –

If the input string is:
Abc23xyz45

The output must be:
4
9
16
25

Note:
If the input string does not have numbers, an error message "**string entered is not alpha numeric**" must be displayed.

Activity 7:

Design an algorithm to accept 15 integer elements for an array. The algorithm must then identify odd and even numbers from the array and display them separately.

Activity 8:

Design an algorithm which accepts values of radius R and height H of a cylinder. The algorithm must then calculate the volume and surface area of the cylinder.

Note:
The values entered for R and H are positive and non-zero. Also both radius and height belong to same unit (may be cm or meters)

Formulas –
Surface area = 2 * Pi * r * (r + h)
Volume = Pi * r * r * h

Value of Pi is 22/7

Activity 9:

Design an algorithm to find the sum of the series given below.
1+1/2+1/3+1/4……..+1/N
The value of N is given as input by the user.

N must be a positive non-zero integer value.

For Example –

If the value of N is 4

The output must be 2.08

(Calculation: 1+1/2+1/3+1/4)

Activity 10:

Design an algorithm which calculates the simple interest given the principal amount, rate of interest and time. The values for principal amount (p), rate of interest (r) and time (t) must be taken as inputs from the user. The simple interest calculated must be displayed as output.

Formula-

Simple Interest = (p * t * r) / 100

Note:

The values entered for p, t and r are positive and non-zero.

Activity 11: *Happy Numbers*

Let the sum of the square of the digits of a positive integer S_0 be represented by S_1. In a similar way, let the sum of the squares of the digits of S_1 be represented by S_2 and so on. If $S_i = 1$ for some i >= 1, then the original integer S_0 is said to be Happy number. A number, in which S_i does not come to 1, is not happy and called 'Unhappy number'.

For Example –

Consider a number 7.

$7^2 = 49 \rightarrow 4^2 + 9^2 = 97 \rightarrow 9^2 + 7^2 = 130 \rightarrow 1^2 + 3^2 = 10 \rightarrow 1^2 = 1$

Therefore 7 is a "**Happy number**"

Consider another number 4.

4 is an "**Unhappy number**" since:

$4^2 = 16 \rightarrow 1^2 + 6^2 = 37 \rightarrow 3^2 + 7^2 = 58 \rightarrow 5^2 + 8^2 = 89 \rightarrow 8^2 + 9^2 = 145 \rightarrow 1^2 + 4^2 + 5^2 = 42 \rightarrow 4^2 + 2^2 = 20 \rightarrow 2^2 + 0^2 = 4$

Design an algorithm which accepts a number (which is non-zero and positive) from the user and then checks if it is a happy number or not.
If the number entered by the user is a happy number then the message "**Happy number**" must be printed otherwise the message "**Unhappy number**" must be printed.

Activity 12: *Find Good Numbers*

A number is termed as a "**GOOD NUMBER**" if meets the 2 criteria's mentioned below;

a. All the digits in the number must be only 1 or 2 or 3
b. No two adjacent substrings of the number is same

Example for good numbers: 1, 121, 1213, 1213121

Example for bad numbers: 11, 1212, 1221, 1231231

- 11 is a bad number because 1 follows 1
- Similarly 1212 is bad because the substring "12" is present adjacent to the substring "12"
- 1221 is a bad number because the substring "2" follows "2"
- 1231231 is a bad number because the substring "123" is adjacent to substring "123"

Design an algorithm to accept an input value N (which is a positive non-zero number). The algorithm must then find out the **smallest good number** which can be formed and has only N digits in it. The number must be displayed as output.

For Example –

If the value on N entered by the user is 4

The smallest possible good number which can be formed having 4 digits is **1213**

Activity 13:

Design an algorithm to accept 2 numbers as input from the user. The algorithm must compute the **Lowest Common Multiple** (LCM) for the 2 numbers and print the number as output.

For Example –

If the numbers entered by the user are: 456 and 12
The output must be 456 which is the LCM of 456 and 12

Activity 14:

Design an algorithm to accept 2 numbers as input from the user. The algorithm must compute the **Greatest Common Divisor** (GCD) for the 2 numbers and print the number as output.

For Example –

If the numbers entered by the user are: 100 and 50

The output must be 10 which is the GCD of 100 and 50

Activity 15:

Design an algorithm to accept 2 numbers as input from the user. The algorithm must compute the **Highest Common Factor** (HCF) for the 2 numbers and print the number as output.

For Example –

If the numbers entered by the user are: 24 and 36

The output must be 12 which is the HCF of 24 and 36