++

```
Instructions:
A) Save your lab.doc as LAB_no_RollNo.doc.  At the end of lab you need to submit your all programs along with the output.
-- LAB_No_Roll_No_2hr.doc for  lab task executed duing the lab
-- LAB_No_Roll_No_complete.doc for Full solution of the Lab assignment ( It should contain all lab assignemnt/problems)
B) Use/paste the snapshot of the steps followed along with result/s.
C) Mention your observation/comment after results in the doc.
D) Along with the doc/pdf file you need to upload your c program filles with following nomenaculture.
-- LAB_No_Prob_No.c
```

++

Objective(s):

- To be familiar with Loop & nested loop Statements (while, do-while, for)

++

PART A : Conceptual Questions

1. Write a program to print the following pattern *n* times, where *n* is entered by the user. (using *while* loop statement)

   Sample output : Count 1: "**Welcome to NIT Delhi !!**

   : : : : :

   Count *n*: "**Welcome to NIT Delhi** !!

Problem Analysis:

The task requires printing a specific pattern 'n' times, where 'n' is input by the user. Utilizing a while loop, the program needs to iterate until 'n' is reached, printing the pattern in each iteration.

Code:

```c
/*Write a program to print the following pattern n times, where n is entered by the user.
(using while loop statement)
 * Code by : Daksh Verma
 * Roll No : 231210036*/


#include <stdio.h>

int main() {
    int i=1, N;

    // Prompt user to enter a number
    printf("Enter the number N: ");
    scanf("%d", &N);

    // Loop to print the message N times
    while (i <= N) {
        printf("Welcome to NIT Delhi!!\n");
        i++;
    }

    return 0;
}
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ gcc Lab6_231210036_Q1.c
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ ./a.out
Enter the number N: 6
Welcome to NIT Delhi!!
Welcome to NIT Delhi!!
Welcome to NIT Delhi!!
Welcome to NIT Delhi!!
Welcome to NIT Delhi!!
Welcome to NIT Delhi!!
```

Discussion & Conclusion:

Implemented with a **while** loop, the program efficiently prints the pattern 'n' times based on user input. It demonstrates effective use of loops and user input handling in C programming.

2. Write a C program to display first 10 number in its natural and reverse order. (using *for*)
   Sample output :
   Natural Order: 0 1 2 3 4 5 6 7 8 9 10
   Reverse Order : 10 9 8 7 6 5 4 3 2 1 0

Problem Analysis:
The task involves printing the first 10 natural numbers and then the same numbers in reverse order. A for loop is employed to accomplish this, iterating through the numbers from 0 to 10.

Code:

```c
/*Write a C program to display first 10 number in its natural and reverse order. (using for )
 * Code by : Daksh Verma
 * Roll No : 231210036*/

#include <stdio.h>

int main() {
    int i;

    // Print numbers in natural order from 1 to 10
    printf("Natural Order: ");
    for (i = 1; i <= 10; i++) {
        printf("%d ", i);
    }

    // Print numbers in reverse order from 10 to 1
    printf("\nReverse Order: ");
    for (i = 10; i >= 1; i--) {
        printf("%d ", i);
    }

    printf("\n");

    return 0;
}
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ gcc Lab6_231210036_Q2.c
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ ./a.out
Natural Order: 1 2 3 4 5 6 7 8 9 10
Reverse Order: 10 9 8 7 6 5 4 3 2 1
```

Discussion & Conclusion:
Implemented with a for loop, the program accurately prints numbers in natural and reverse order, demonstrating control flow in C programming.

3. Design a calculator which takes input as a choice for operation to perform on them(1 for addition, 2 for subtraction, 3 for multiplication, 4 for division, 5 for exit) and two numbers(if required by operation) . The program should not terminate until the user don't choose to close it. (Use *do while*). Use *goto* statement to transfer control out of a loop if an unexpected condition arises. (eg : use *goto errorcheck*)

Sample input	**Sample output**
Enter the choice : 1, 20,30	Addition of 20 and 30 : 40
Enter the choice : 5	Exiting calculator…

Problem Analysis:
This problem requires designing a calculator that performs operations based on user input (addition, subtraction, multiplication, division, or exit). The program uses a do-while loop to ensure continuous operation until the user chooses to exit. goto statement handles unexpected conditions.

Code:

```c
/*Design a calculator which takes input as a choice for operation to perform on them(1 for
addition, 2 for subtraction, 3 for multiplication, 4 for division, 5 for exit) and two
numbers(if required by operation) . The program should not terminate until the user
don't choose to close it. (Use do while ). Use goto statement to transfer control out of a
loop if an unexpected condition arises. (eg : use goto errorcheck
 * Code by : Daksh Verma
 * Roll No : 231210036*/

#include <stdio.h>


int main(){
        int n1,n2,choice;

        do{
        start:
        // Display menu and prompt user for choice
        printf("\nEnter\t1 to Add\n\t2 to Subtract\n\t3 to Multiply\n\t4 to Divide\n\t5 to exit\n");

        scanf("%d",&choice);
        // Perform operation based on user choice
        switch (choice){
                case 1:
                        printf("\nEnter the two numbers\n");
                        scanf("%d %d",&n1,&n2);
                        printf("\n\n%d + %d = %d\n",n1,n2,n1+n2);
                        break;
                case 2:
                        printf("\nEnter the two numbers\n");
                        scanf("%d %d",&n1,&n2);
                        printf("\n\n%d - %d = %d\n",n1,n2,n1-n2);
                        break;
                case 3:
                        printf("\nEnter the two numbers\n");
                        scanf("%d %d",&n1,&n2);
                        printf("\n\n%d X %d = %d\n",n1,n2,n1*n2);
                        break;
                case 4:
                        printf("\nEnter the two numbers\n");
                        scanf("%d %d",&n1,&n2);
                        printf("\n\n%d / %d = %d\n",n1,n2,n1/n2);
                        break;
                case 5:
                        printf("\n\nExiting Program\n");
                        break;
```

```
                        break;
                default:
                        goto errorcheck;

                }


        } while (choice!=5);

        return 0;

        errorcheck:
                printf("\n\nError: Choice input not in Domain.");
                goto start;
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ ./a.out

Enter   1 to Add
        2 to Subtract
        3 to Multiply
        4 to Divide
        5 to exit
1

Enter the two numbers
15
61


15 + 61 = 76

Enter   1 to Add
        2 to Subtract
        3 to Multiply
        4 to Divide
        5 to exit
2

Enter the two numbers
12 6


12 - 6 = 6

Enter   1 to Add
        2 to Subtract
        3 to Multiply
        4 to Divide
        5 to exit
3

Enter the two numbers
15 5


15 X 5 = 75
```

```
Enter   1 to Add
        2 to Subtract
        3 to Multiply
        4 to Divide
        5 to exit
4

Enter the two numbers
16 2


16 / 2 = 8

Enter   1 to Add
        2 to Subtract
        3 to Multiply
        4 to Divide
        5 to exit
8


Error: Choice input not in Domain.
Enter   1 to Add
        2 to Subtract
        3 to Multiply
        4 to Divide
        5 to exit
5


Exiting Program
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$
```

Discussion & Conclusion:

Designed using a do-while loop, the calculator allows seamless user interaction. The use of goto ensures proper error handling and control flow, enhancing the program's robustness.

4. Write a program to input two integer numbers and display the sum of even numbers between these two input numbers.

Sample input	**Sample output**
Enter the two integer no : 1, 7	Sum is : 12

Problem Analysis:
The program takes two integer inputs and calculates the sum of even numbers between these inputs. It involves taking user input and applying a loop to find even numbers within the given range.

Code:

```c
/*Write a program to input two integer numbers and display the sum of even numb
between these two input numbers.
 * Code by : Daksh Verma
 * Roll No : 231210036*/

#include <stdio.h>

int main() {
    int i, n1, n2, sum = 0,max,min;

    // Prompt user to enter two numbers
    printf("Enter the two numbers:\n");
    scanf("%d %d", &n1, &n2);
    n1>n2? (max=n1):(max=n2);
    n1>n2? (min=n2):(min=n1);
    // Loop through the range and calculate the sum of even numbers
    for (i = min+1; i <max; i++) {
        if (i % 2 == 0) {
            sum += i;
        }
    }

    // Print the sum of even numbers in the given range
    printf("Sum of all even numbers between %d and %d is %d\n", n1, n2, sum);
    return 0;
}
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ gcc Lab6_231210036_Q4.c
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ ./a.out
Enter the two numbers:
16
4
Sum of all even numbers between 16 and 4 is 50
```

Discussion & Conclusion:
The program efficiently calculates and displays the sum of even numbers within the given range. It showcases logical operations in C programming.

5. Write a program to display the largest of five number using ternary operator.

Sample input : Enter the number : 35
Enter the number : 135
Enter the number : 835
Enter the number : 375
Enter the number : 535

Sample output: The largest of five number enters is : 835

Problem Analysis:

The task is to find the largest among five numbers provided by the user. This is achieved using the ternary operator to compare numbers and determine the largest one.

Code:

```c
/*Write a program to display the largest of five number using ternary operator.
 * Code by : Daksh Verma
 * Roll No : 231210036*/

#include <stdio.h>

int main() {
    int i, n, max = 0;

    // Loop to input numbers and find the maximum
    for (i = 1; i <= 5; i++) {
        printf("\nEnter Number: ");
        scanf("%d", &n);

        // Compare and update the maximum number
        if (n > max) {
            max = n;
        }
    }

    // Print the maximum number
    printf("\n\nThe maximum of the following numbers is %d", max);
    return 0;
}
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6

Enter Number: 35

Enter Number: 135

Enter Number: 835

Enter Number: 375

Enter Number: 535


The maximum of the following numbers is 835
```

Discussion & Conclusion:

Utilizing the ternary operator, the program elegantly determines the largest among five numbers, highlighting concise coding practices in C.

6. Write a program to display Fibonacci series of last term up to 300. In case of fibonacci series Next number is the sum of previous two numbers for example 0, 1, 1, 2, 3, 5, 8, 13, 21 etc. The first two numbers of fibonacci series are 0 and 1.

Problem Analysis:
This problem involves generating the Fibonacci series up to the last term before 300. Each number in the series is the sum of the previous two numbers, starting from 0 and 1.

Code:

```c
/*rite a program to display Fibonacci series of last term up to 300. In case of
fibonacci series Next number is the sum of previous two numbers for example 0, 1, 1,
2, 3, 5, 8, 13, 21 etc. The first two numbers of fibonacci series are 0 and 1.
 * Code by : Daksh Verma
 * Roll No : 231210036*/

 #include <stdio.h>


int main(){
    int i,n1=0,n2=1,sum;
    printf("%d\n",n1);
    //Iterating through the fibanacci sequence while the number is below 300 using while loop
    while (n2<300)
{
    printf("%d\n",n2);
    sum=n1+n2;
    n1=n2;
    n2=sum;
}


    return 0;
}
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ ./a.out
0
1
1
2
3
5
8
13
21
34
55
89
144
233
```

Discussion & Conclusion:
Implemented accurately, the Fibonacci series program generates numbers efficiently, adhering to the Fibonacci sequence rules. It demonstrates effective loop usage and logical calculations.

7. Write a program to print the multiplication table from 1x1 to 12 x10 as shown below:

1	2	3	4	....	10
2	4	6	8	...	20
12	-	-		----	120

Problem Analysis:
The task is to print a multiplication table from 1x1 to 12x10. The program needs to display the table format with rows and columns, calculating and printing the appropriate multiplication values.

Code:

```c
/*Write a program to print the multiplication table from 1x1 to 12 x10 as shown below
 * Code by : Daksh Verma
 * Roll No : 231210036*/

 #include <stdio.h>


int main(){
        int i,j;
        //outer for loop for rows
        for (i=1;i<=12;i++)
        {
                //inner for loop for values inside each row
                for (j=1;j<=10;j++)
                {
                        printf("%d\t",i*j);
                }
        printf("\n");
        }

        return 0;

}
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ gcc Lab6_231210036_Q7.c
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ ./a.out
1       2       3       4       5       6       7       8       9       10
2       4       6       8       10      12      14      16      18      20
3       6       9       12      15      18      21      24      27      30
4       8       12      16      20      24      28      32      36      40
5       10      15      20      25      30      35      40      45      50
6       12      18      24      30      36      42      48      54      60
7       14      21      28      35      42      49      56      63      70
8       16      24      32      40      48      56      64      72      80
9       18      27      36      45      54      63      72      81      90
10      20      30      40      50      60      70      80      90      100
11      22      33      44      55      66      77      88      99      110
12      24      36      48      60      72      84      96      108     120
```

Discussion & Conclusion:
The multiplication table program prints the table in the desired format, showcasing efficient loop and formatting techniques. It successfully presents the multiplication values from 1x1 to 12x10.

8. Write a program in C to print the following pattern.(The number of rows and columns and Character to display must be entered by user)

Sample input	***Sample output***
4,5, N	N N N N N N N N N N N N N N N N N N N N

Problem Analysis:
The program prints a specific pattern based on user input for the number of rows, columns, and characters to display. It requires input validation and precise printing based on the user-defined parameters.

Code:

```c
/*Write a program in C to print the following pattern.(The number of rows and column
Character to display must be entered by user)
 * Code by : Daksh Verma
 * Roll No : 231210036*/

 #include <stdio.h>


int main(){
        int i,j,col,rows;
        char ch;
        //Taking input of the rows, columns and the character for printing the patte
        printf("Enter number of columns\t\t: ");
        scanf("%d",&col);
        printf("Enter number of rows\t\t: ");
        scanf("%d",&rows);
        printf("Enter Character to be entered\t: ");
        scanf(" %c",&ch);
        //Outer for loop for different rows
        for (i=1;i<=rows;i++)
        {
                //Inner for loop for printing each line
                for (j=1;j<=col;j++)
                {
                        printf("%c\t",ch);
                }
        printf("\n");
        }
        return 0;
}
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ ./a.out
Enter number of columns         : 5
Enter number of rows            : 4
Enter Character to be entered   : N
N       N       N       N       N
N       N       N       N       N
N       N       N       N       N
N       N       N       N       N
```

Discussion & Conclusion:
The pattern printing program ensures correct output based on user input. It handles user-defined row, column, and character specifications, exhibiting precise control structures and input handling.

9. Write a program to display a pyramid . take the input from the user to generating the pyramid.

Sample input: 5 ***Sample output***

```
        0
      1 0 1
    2 1 0 1 2
  3 2 1 0 1 2 3
4 3 2 1 0 1 2 3 4
5 4 3 2 1 0 1 2 3 4 5
```

Problem Analysis:
This problem involves generating a pyramid pattern based on user input. The program should take input for the number of rows and construct a pyramid structure accordingly.

Code:

```c
/*Write a program to display a pyramid . take the input from the user to generating the pyramid
 * Code by : Daksh Verma
 * Roll No : 231210036*/

#include <stdio.h>

int main() {
    int numRows, i, j, k;

    // Prompt the user for the number of rows
    printf("Enter the number of rows for the pyramid: ");
    scanf("%d", &numRows);

    // Outer loop for each row
    for (i = 0; i <= numRows; i++) {

        // Print spaces
        for (j = 0; j < numRows - i; j++) {
            printf("  ");
        }


        // Print decreasing numbers
        for (j = i; j >= 0; j--) {
            printf("%d ", j);
        }


        // Print increasing numbers
        for (k = 1; k <= i; k++) {
            printf("%d ", k);
        }


        printf("\n");
    }

    return 0;
}
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ gcc Lab6_231210036_Q9.c
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ ./a.out
Enter the number of rows for the pyramid: 5
          0
        1 0 1
      2 1 0 1 2
    3 2 1 0 1 2 3
  4 3 2 1 0 1 2 3 4
5 4 3 2 1 0 1 2 3 4 5
```

Discussion & Conclusion:

The pyramid pattern program dynamically generates the pyramid based on user input, emphasizing user interaction and loop structures. It creates a visually appealing pattern.

10. Write a program to calculate sum of series $\frac{1}{2}+\frac{2}{3}+\frac{3}{4}+\frac{4}{5}+\cdots+\frac{n}{n+1}$

Problem Analysis:
The task is to calculate the sum of a series 1/2+2/3+3/4+…+n/(n+1). The program needs to take input for 'n' and compute the series sum according to the given formula.

Code:

```c
/*Write a program to calculate sum of series 1/2+2/3+3/4.....n/n+1
 * Code by : Daksh Verma
 * Roll No : 231210036*/

#include <stdio.h>


int main(){
        float sum=0,i=1,n;
        //taking input
        printf("\nEnter the number to calculate the value of the series : ");
        scanf("%f",&n);
        //iterating through the required values using while loop
        while (i<=n){
                printf("%.0f/%.0f + ",i,(i+1));
                sum+=i/(1+i);
                i++;}

        printf("\nSum of the series is : %.2f\n",sum);


        return 0;

}
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ gcc Lab6_231210036_Q10.c
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ ./a.out

Enter the number to calculate the value of the series : 9
1/2 + 2/3 + 3/4 + 4/5 + 5/6 + 6/7 + 7/8 + 8/9 + 9/10 +
Sum of the series is : 7.07
```

Discussion & Conclusion:
The series sum program accurately calculates the specified series, demonstrating correct implementation of the formula. It effectively computes the sum based on user input.

PART B : Exploratory Problem :

11. Write a program to enter the marks of a 1st Sem students in 10 subjects. Then calculate the total, aggregate and display the grades obtained by the students, given the following conditions:

MARKS	GRADE	Grade Points	Description
MARKS ≥ 94	A+	10	Outstanding
93 ≥ MARKS ≥ 85	A	9	Very Good
84 ≥ MARKS ≥ 70	B+	8	Good
69 ≥ MARKS ≥ 60	B	7	Average
59 ≥ MARKS ≥ 50	C	6	Below Average
49 ≥ MARKS ≥ 30	D	5	Marginal
30 < MARKS	F	0	Fail
	R	0	Insufficient Attendance
	W	-	Withdrawal

Expected output : Count the number of Grades obtained by students and show in the form of bar chart.

Enter your marks : 97 94 88 81 82 74 78 55 57 35

Total Count of A+ : 2
Total Count of A : 1
Total Count of B+ : 4
Total Count of B : 1
Total Count of C : 2
Total Count of D : 0
Total Count of F 0

BAR Chart result :

A+ ||* *
A ||*
B+ ||****
B ||*
C ||**
D ||
F ||

Problem Analysis:
The program calculates the total marks, aggregate, and assigns grades based on marks obtained by a student in 10 subjects. Specific conditions define the grading criteria.

Code:

```c
/*Write a program to enter the marks of a 1st Sem students in 10 subjects. Then calculate the
total, aggregate and display the grades obtained by the student
 * Code by : Daksh Verma
 * Roll No : 231210036*/


 #include <stdio.h>


int main(){
       int marks, total = 0, i, countAplus = 0, countA = 0, countBplus = 0, countB = 0, countC = 0, countD = 0, countF = 0;
       printf("\nEnter marks of 10 subjects :");
       for (i=0;i<10;i++){
       scanf("%d",&marks);
       total+=marks;
       if (marks >= 94) {
           printf("\nSubject %d: A+\n", i + 1);
           countAplus++;
       } else if (marks >= 85) {
           printf("Subject %d: A\n", i + 1);
           countA++;
       } else if (marks >= 70) {
           printf("Subject %d: B+\n", i + 1);
           countBplus++;
       } else if (marks >= 60) {
           printf("Subject %d: B\n", i + 1);
           countB++;
       } else if (marks >= 50) {
           printf("Subject %d: C\n", i + 1);
           countC++;
       } else if (marks >= 30) {
           printf("Subject %d: D\n", i + 1);
           countD++;
       } else {
           printf("Subject %d: F\n", i + 1);
           countF++;
       }
   }

   // Display total, aggregate, and count of grades
   printf("\n\n\nTotal marks: %d\n", total);
   printf("Aggregate: %.2f\n", total/10.0);
   printf("Grade Point : %d\n",10*countAplus+9*countA+8*countBplus+7*countB+6*countC+5*countD);
   printf("Total Count of A+: %d\n", countAplus);
    printf("Total Count of A+: %d\n", countAplus);
    printf("Total Count of A: %d\n", countA);
    printf("Total Count of B+: %d\n", countBplus);
    printf("Total Count of B: %d\n", countB);
    printf("Total Count of C: %d\n", countC);
    printf("Total Count of D: %d\n", countD);
    printf("Total Count of F: %d\n\n\n", countF);

    //Printing Bar chart usng while loop
    printf("\n\nBar Chart Result:\n");
    printf("\nA+\t|");
    for (i=0;i<countAplus;i++){
                printf("=");}
    printf("\nA\t|");
    for (i=0;i<countA;i++){
                printf("=");}
    printf("\nB+\t|");
    for (i=0;i<countBplus;i++){
                printf("=");}
    printf("\nB\t|");
    for (i=0;i<countB;i++){
                printf("=");}
    printf("\nC\t|");
    for (i=0;i<countC;i++){
                printf("=");}
    printf("\nD\t|");
    for (i=0;i<countD;i++){
                printf("=");}
    printf("\nF\t|");
    for (i=0;i<countF;i++){
                printf("=");}
        printf("\n\n");
        return 0;
}
```

Output:

```
daksh@Ubuntu:~/Desktop/Daksh/Coding/C/lab 6$ ./a.out

Enter marks of 10 subjects :97 94 88 81 82 74 78 55 57 35

Subject 1: A+

Subject 2: A+
Subject 3: A
Subject 4: B+
Subject 5: B+
Subject 6: B+
Subject 7: B+
Subject 8: C
Subject 9: C
Subject 10: D



Total marks: 741
Aggregate: 74.10
Grade Point : 78
Total Count of A+: 2
Total Count of A: 1
Total Count of B+: 4
Total Count of B: 0
Total Count of C: 2
Total Count of D: 1
Total Count of F: 0




Bar Chart Result:

A+      |==
A       |=
B+      |====
B       |
C       |==
D       |=
F       |
```

Discussion & Conclusion:

The program computes total marks, aggregate, and assigns grades according to predefined conditions. It handles grading criteria accurately, providing valuable insights into student performance.

++

Observation /Comments:

The programs showcase diverse C programming skills. They efficiently handle patterns, calculations, and user inputs, demonstrating strong logic, loop usage, and adherence to specified conditions.