Date: 02/02/2022

EXPERIMEMT NO. 4

Q. To study and demonstrate the Loop Control Structures.

PROGRAM STATEMENTS:

a) Write a C program to compute sum of integers provided by the user. Set the terminating condition for the program as a number -9999.

b) Write a C program using while loop to display all n-digit Armstrong number where value of ‘n’ should be in the range 2 to 6 (both inclusive).

c) Write a C program using for loop to print given numbers.

1

2 3 2

3 4 5 4 3

4 5 6 7 6 5 4

5 6 7 8 9 8 7 6 5

THEORY: For the all the programs listed above, we have to use loops-

For, while, do while, etc,.

* Syntax of while loop:

While (condition)

{

Statements;

}

* Syntax of for loop:

For (initialization expr; test expr ;update expr)

{

//body of the loop

//statements we want to execute

}

For example,

For (int counter=1; counter<=10; counter++)

* where “for” is the keyword, “counter” is the control variable name, “1” is the starting value of control variable, “counter<=10” is the loop where 10 is the final value of control variable for which the condition is true, (;) semicolon as the separator and ++ is the increment of the control variable.

Program a):

#include <stdio.h>

int main()

{

int num,stop=9999,Sum=0;

for(;;)

{

printf("Enter a number: ");

scanf("%d",&num);

if(num==stop)

{

break;

}

else

{

Sum=Sum+num;

}

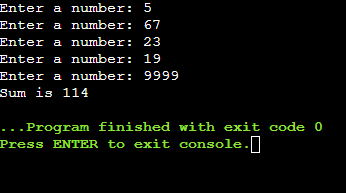
}

printf("Sum is %d",Sum);

return 0;

}

OUTPUT a):



Program b)

#include <stdio.h>

#include <math.h>

void main()

{

int num,originalNum,rem,ori,m,result=0,k,i;

originalNum=num;

while(num<1000000)

{

originalNum=num;

result=0;

for(i=0;originalNum!=0;i++)

{

originalNum/=10;

}

ori=num;

for(k=0;ori!=0;k++)

{

rem=ori%10;

result+=pow(rem,i);

ori=ori/10;

}

if (result==num && num>9)

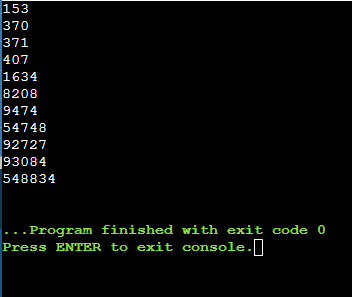
printf("%d\n",num);

num++;

}

}

OUTPUT b)



Program c)

#include <stdio.h>

void main()

{

int i,space,rows,k=0,c=0,c1=0;

printf("Enter the number of rows: ");

scanf("%d",&rows);

for(i=1;i<=rows;++i)

{

for(space=1;space<=rows-i;++space)

{

printf(" ");

++c;

}

while(k!=2\*i-1)

{

if(c<=rows-1)

{

printf("%d",i+k);

++c;

}

else

{

++c1;

printf("%d",(i+k-2\*c1));

}

++k;

}

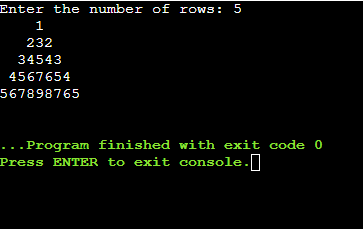
c1=c=k=0;

printf("\n");

}

}

OUTPUT c)



CONCLUSION: We learned about the different types of loops used in C and their usage. They provides the code reusability. Using loops we don’t need to write the same code again and again according to the given condition. It saves the time of user by reducing the code.