Mechi Multiple Campus

(Tribhuvan University)
Bhadrapur, Jhapa

Lab Report of

Data Structures and Algorithm (CACS-201)

Implementation of Recursion

Faculty of Humanities & Social Sciences

Tribhuvan University

Kritipur, Nepal

Submitted By

Name: Santosh Bhandari

Roll No: 58

Submitted To

Mechi Multiple Campus

Department of Bachelor in Computer Application

Bhadrapur, Jhapa, Nepal

Introduction to recursion

Definition

Recursion is a concept of repeating the execution of statements for multiple times by calling the function or procedure itself until some condition is satisfied. The process is used for repetitive computations in which action is stated in terms of a previous result.

Applications

i) factorial Calculation
ii) fibonacci Sequence Generation
iii) Tower of Hanoi (ToH)
iv) Binary Search
v) Quick Sort

Advantages

i) It is the most natural way to solve some problem like factorial, fibonacci, TOH.
ii) It reduces the size of program code.

Disadvantages

i) It requires longer processing time
ii) It has larger memory requirement
iii) It is not as efficient as iteration
iv) It is slower.

Algorithm to Create factorial

```
① START
② Input a number (n)
③ r = Factorial (n){
        if n=0 OR n=1
            then return 1
        else
            return (n * Factorial (n-1))
    }
④ Output r
⑤ EXIT
```

Program Code and Output to Calculate Factorial

```c
#include<stdio.h>
int fact(int n){
        if(n==1|| n==0)
                return 1;
        else
                return n*fact(n-1);
}
void main(){
        int num;
        printf("Enter the Number for Factorial: ");
        scanf("%d",&num);
        printf("Factorial = %d",fact(num));
}
```

```
Enter the Number for Factorial: 10
Factorial = 3628800
```

Algorithm to calculate term of a Fibonacci Series

```
① START
② Input a Number (n)
③ term = fib(n)
        if n=1
            return 0;
        else if n=2
            return 1
        else
            return (fib(n-1)+fib(n-2));
④ Output term
⑤ EXIT
```

Program code and output to calculate the term of a Fibonacci series

```
#include<stdio.h>
int fib(int n){
        if(n==1)
                return 0;
        else if(n==2)
                return 1;
        else
                return fib(n-1)+fib(n-2);
}
void main(){
        int num;
        printf("Which Term Fibonacci Number You Want: ");
        scanf("%d",&num);
        printf("Fibonacci Number = %d",fib(num));
}
```

```
Which Term Fibonacci Number You Want: 10
Fibonacci Number = 34
```

Algorithm to Calculate the reverse of Number

```
① START
② Input the Number (n)
③ r = Reverse (n, length)
       if n = 1
           return n
       else
           return ((n%10) * pow (10, length)) + Reverse (n/10, --length
④ Output r
⑤ EXIT
```

Program code and output to calculate the reverse of a number

```
#include<stdio.h>
#include<math.h>
int Reverse(int num, int l){
        if(l==1)
                return num;
        else
                return ((num%10)*pow(10,l-1))+Reverse(num/10,--l);
}
void main(){
        int num,length=0,temp;
        printf("Enter the Numeber: ");
        scanf("%d",&num);
        temp=num;
        while(temp!=0){
                length++;
                temp/=10;
        }
        printf("Reversed of Number = %d",Reverse(num,length));
}
```

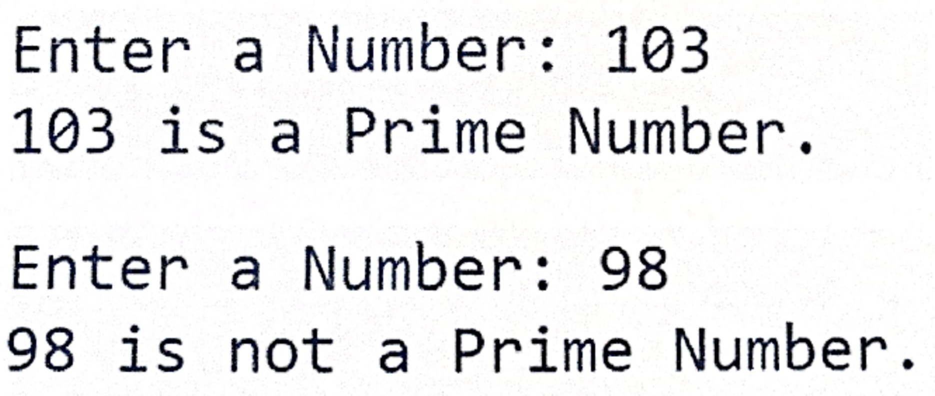
```
Enter the Numeber: 58984651
Reversed of Number = 15648985
```

Algorithm to check if a number is prime or not

```
(1) START
(2) INPUT the Number (Num)
(3) r = Check(num, n)
        if n=0 or n=1
                return 0
        else if num%n=0
                return 1
        else
                return Check(num,n-1)
(4) Output r
(5) EXIT
```

Program Code and Output to Check if a Number is Prime or Nor

```
#include<stdio.h>
int Check(int num, int n){
        if(n==0 || n==1)
                return 0;
        else if(num%n==0)
                return 1;
        else
                return Check(num,n-1);
}
void main(){
        int n;
        printf("Enter a Number: ");
        scanf("%d",&n);
        if(Check(n,n-1)==0)
                printf("%d is a Prime Number.",n);
        if(Check(n,n-1)==1)
                printf("%d is not a Prime Number.",n);
}
```



```
Enter a Number: 103
103 is a Prime Number.

Enter a Number: 98
98 is not a Prime Number.
```

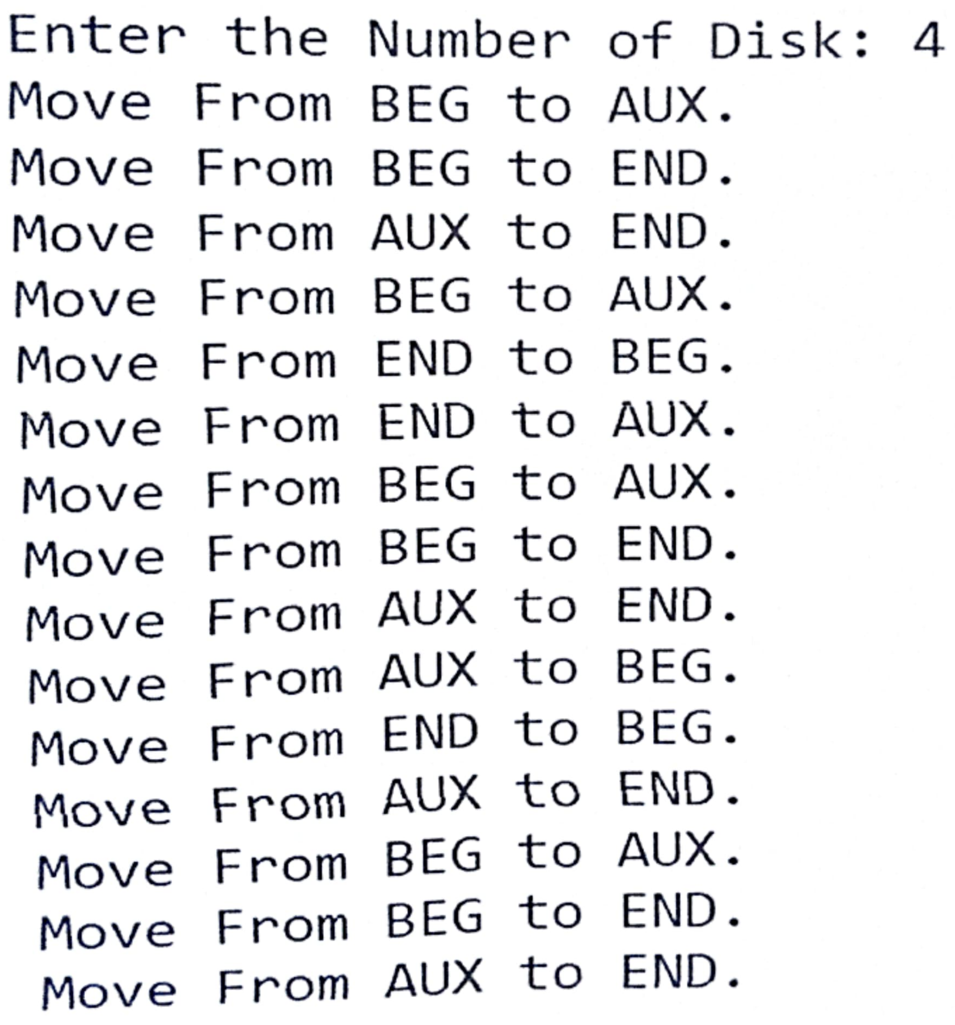
Algorithm to solve the Tower of Hanoi (TOH)

```
① START
② Input the Number of desk (N)
③ TOH (N, BEG, AUX, END)
        if(n>0)
                TOH (N-1, BEG, END, AUX)
                Write BEG → END
                TOH (N-1, AUX, BEG, END)
                Return

④ EXIT
```

Program code and output to solve the Tower of Hanoi

```
#include<stdio.h>
void TOH(int n, char BEG[3], char AUX[3],char END[3]){
        if(n>0){
                TOH(n-1,BEG,END,AUX);
                printf("Move From %s to %s.\n",BEG,END);
                TOH(n-1,AUX,BEG,END);
        }
}
void main(){
        int n;
        printf("Enter the Number of Disk: ");
        scanf("%d",&n);
        TOH(n,"BEG","AUX","END");
}
```



```
Enter the Number of Disk: 4
Move From BEG to AUX.
Move From BEG to END.
Move From AUX to END.
Move From BEG to AUX.
Move From END to BEG.
Move From END to AUX.
Move From BEG to AUX.
Move From BEG to END.
Move From AUX to END.
Move From AUX to BEG.
Move From END to BEG.
Move From AUX to END.
Move From BEG to AUX.
Move From BEG to END.
Move From AUX to END.
```

Conclusion.

Hence, Recursion is the process in which a function calls itself repeatedly, Until some specified condition has been satisfied. To be recursion, the function should call itself and there must be stopping condition.