RECURSION

A function is called recursive if the body of a function calls the same function. if else statement can be used where one branch makes the recursive call and the other doesn't.

Syntax

```
void recurse() //invoked function
{
   ... .. ...
   recurse(); //again continues the execution of the function
   ... .. ...
}

int main()
{
   ... .. ...
   recurse(); //this calls the recurse function
   ... .. ...
}
```

The recursive function is controlled by using if else statement by putting in some condition to satisfy the recursion.

Advantages and Disadvantages of Recursion

Recursion makes program elegant. However, if performance is vital, use loops instead as recursion is usually much slower. That being said, recursion is an important concept. It is frequently used in data structure and algorithms.

IMPORTANT QUESTIONS

1. What is recursion ?

   A function is called recursive if the body of a function calls the same function.
2. What is the advantage of recursive function?

   Its easy for one to call the function to execute it again. It reduces the size of the programme. Easy to solve out problems.

3. What is the necessity of exit condition in recursion?

   Exit condition is needed in recursion because if exit condition is not there the programme will continue to be executed based on the memory of the computer and sometimes we may not be getting the result as well.

Github Link for all the below programs :- https://github.com/Amarjith-c-k/recursion-factorial-and-fibonacci

4. Write a programme to find factorial of a given number using recursion and without using recursion?

   Without recursion

   https://github.com/Amarjith-c-k/recursion-factorial-and-fibonacci/blob/master/factorial.c

```
//factorial of a number without using recursion
#include<stdio.h>
int fct(int n);
int main()
{
   int n,f;
   printf("enter the number to find factorial \n");
   scanf("%d",&n);
   f=fct(n);
   printf("factorial of %d is %d",n,f);
   return 0;
}
int fct(int n)
{
   int f=1,i;
      for(i=1;i<=n;++i)
         {
        f *=i;
         }
   return f;
}
```

   With recursion

   https://github.com/Amarjith-c-k/recursion-factorial-and-fibonacci/blob/master/factorialr.c

```
//factorial of a number using recursion
#include<stdio.h>
int fct(int n);
int main()
{
   int n,f;
   printf("enter the number to find factorial \n");
   scanf("%d",&n);
   f=fct(n);
   printf("factorial of %d is %d",n,f);
   return 0;
}
int fct(int n)
{
   if(n>1)
      return n*fct(n-1);
   else
      return n;
}
```

5. Write a programme on Fibonacci series using recursion and not

Using recursion

https://github.com/Amarjith-c-k/recursion-factorial-and-fibonacci/blob/master/fibonacci.c

```
//fibonacci series using recursion
#include <stdio.h>
int fb(int n)
{
   if ( n == 0 )
     return 0;
  else if ( n == 1 )
     return 1;
  else
     return ( fb(n-1) + fb(n-2) );
}
int main() {
   int c,i=0, n;
   printf("Enter the number of terms: ");
   scanf("%d", &n);
   printf("Fibonacci series\n");
   for ( c = 1 ; c <= n ; c++ )
   {
      printf("%d, ",fb(i));
      i++;
   }
   return 0;
}
```

Without recursion
https://github.com/Amarjith-c-k/recursion-factorial-and-fibonacci/blob/master/fibonaccir.c

```
//fibonacci without usng recursion
#include <stdio.h>
int fb(int n)
{
      int i,t1 = 0, t2 = 1, nextTerm;
      printf("Fibonacci Series: ");
      for (i = 1; i <= n; ++i)
   {
      printf("%d, ", t1);
      nextTerm = t1 + t2;
      t1 = t2;
      t2 = nextTerm;
   }
    return 0;
}
int main() {
   int i, n;
   printf("Enter the number of terms: ");
   scanf("%d", &n);
   fb(n);
   return 0;
}
```