6/06/

1. Write a program to print fibonacci series using recursion

Aim: To print fibonacci series using recursion method in a language

Algorithm:

Add integer value to num.

Then run a for loop (i<0; i<num; i++)

To each iteration print and call the fibonacci series function with i as a parameter

in the recursive function (i-1) + fibonacci(i-2)

Code:

```
#include <stdio.h>
int fibonacci series(i){
    if (i<=1)
        return i;
else
    return (fibonacci (i-1)+fibonacci series (i-2));
    }
int main(void) {
        int num=10,i;
        for(i=0; i< num; i++){
        print ("%d", fibonacciseries (i));
        }
    return 0;
    }
```

2. write a program to check the given no is Armstrong or Not using recursive function.

Aim:

To print Armstrong Number checking using recursive function.

Algorithm:

1. start
2. define function 'power' and input base and experiment
3. if experiment is 0, return 1 (base case), otherwise return 'base multiplied by' power(base, experiment-1) (recursive ok)

Define function 'order'

Input 'Num' 'initialize' count to 0

write 'Num' is not 0;

define 'main' function

declare the integer 'Number'.

Call the Armstrong (Number)

If the result is '1' print the Number is an Armstrong Number.

Otherwise, print the Number Not an Armstrong Number

Code:

```
#include<stdio.h>
#include<math.h>
int fun(int num) {
```

```
    int count = 0;
    while (num) {
        num = num/10;
        count ++;
    return count;
    }
int func (int num) {
    int original = num;
    int temp = func (num);
    int sum = 0;
    while (num != 0) {
    int temp2 = num % 10;
    sum = sum + pow (temp2, temp);
    num = num/10;
    }
    return sum == original
}
int main () {
    int num
    scanf ("%d", &num);
    if (func2 (num)) {
    printf ("%d is armstrong", num);
    }
    else {
    printf ("%d is not armstrong", num);
    }
    return 0;
    }
```

3 Write a program to find the GCD of two numbers using recursive factorization.

Aim: To print the GCD of the Number using recursive factorization

Algorithm:

start

Take function integer name gcd int a and b

if b=0 return to a else make gcd(b, a%b)

int main()

Take the Num1, Num2 Assignations and printgcd (Num1, Num2)

stop

```
# include<stdio.h>
    int gcd(int a, int b)
    {
      if(b==0)
      return a;
    else
      return gcd(b, a%b);
      }
  int main() {
      int num1=45, num2=18;
  printf("%d", gcd(Num1, Num2));
      return 0;
    }
```

4. write a program to get the largest element in Array

Aim: to print the largest element in a array using c language

Algorithm:

start

Take int main()

Take int num()

assign the array in the program

a[] = {1, 2, 3, 4, 5}

4. using the forloop to find the max element

5. print the max element

6. stop

program

```
#include<stdio.h>
int main()
{
int arr[] = {1,2,3,4,5}
int n= size of (arr)/ size of (arr[0]);
    int max = arr[0];
for (int i=1; i<n ; i++) {
 if ( arr[i] > max) {
    max = arr[i];
  }
}
print ("%d \n", max);
  return 0;
```

```
printf("%d\n", max);
return 0;
}
```

5. write a program to find the factorial of the number using recursion

Aim: To find (print) the factorial of the number using recursion

Algorithm:

1. start
2. void fact function with argument (int num)
3. if num = 0 return 1 else return sum (num) to fun(num-1)
4. new loop int main()
5. Assign value 5
6. print the factorial
7. stop

Code:

```
#include<stdio.h>
void fun(int num){
if(num==0){
return 1;
}
else
return func(num) * fun(num-1)
int num()
{
    int num 5;
    printf("%d", fun(num));
    return 0;
}
```

6. write a program for to copy one string to another using recursion

Aim: To print a program to copy one string to another string using recursion

Algorithm

1. start
2. use void function copystring Assign two files sources, destination to copy the string from source to destruction
3. Destination = source use the copystring()
4. int main() use not print the source and
5. destination file

Code:

```
#include <stdio.h>
  void copystring (char *source, char *destruction)
   {
    if (*source=='\0') {
      *destination='\0';
       return i;
    }
    *destination= *source;
    copystring (source+1, destination+1);
   }
    int main()
   char source [] = 'Hello'
   char destination [50];
    copystring (source, destination);
    printf ("%s", source);
```

```
printf("%s", destination)
return 0;
}
```

7. write a program to reverse the string using recursion

Aim :- To Print the Reverse of a string using recursion method in c language

Algorithm :-

1. start
2. Data the void (Null) function name reverse Input int, int
3. if start s = and return else char temp = str[start]; str[start] str[end]; str[end] = temp;
4. Take the int main() {
5. and call the function Reverse (char, int, int)
6. print the str
7. return the function
8. stop

Code :-

```
#include<stdio.h>
#include<string.h>
void reverse(char *str, int start, int end){
    if(start >= end){
        return;
    }
    char temp = str[start];
    str[start] = str[end];
```

```
        str[end] = temp;
    reverse(str, start + 1, end - 1); }
int main() {
    char str[] = "hello world";
    reverse(str, 0, strlen(str) - 1);
printf("%s", str);
    return 0;
    }
```

8. Write the program to generate all the prime numbers using recursion.

Aim: The program to print to generate the all the prime numbers using recursion.

Algorithm:
==========

1. Define a recursive number function that accepts an integer num.
2. initialize a variable "i" to 2
3. if num is equal to 0 or 1, then RETURN false
4. if num is equal to "i", then RETURN true.
5. if num is divisible by "i", the RETURN false
6. increment "i"
7. Recursively call the function and pass num as an argument

Code:

```
#include <stdio.h>
#include<stdio.h>
    bool find prime (int num)
    { static int i=2;
```

```
    if (num == 0 || num == 1)
    {
      return false;
    }
  if (num == 2)
  return true;
  if(num%i == 0)
  {
    return false;
   }
    i++;
  return find prime(num);
   }
   int main()
  {
   int num = 20;
 if (find. prime (num))
  {
   printf("%d" is a prime Number \n", num);
    }
  else
  {
   printf(" %d is not a prime Number", num)
  }
  return 0;
  }
```

9 Write a program to check a number is a prime number or not using recursion.

Ans: To check a number is a prime number or not using recursion.

Algorithm

1. start
2. define the recursive function which accepts as integer as parameter, say num,
3. initialize the value of i with '2'
4. Now, decide the base condition of the function when the Num value is equal to 0 or 1, the return the false
5. if the Num value is equal to 1 then return true
6. if the number is divisible by i then return false return false and increment i;
7. keep calling the function recursively until it reaches the base value

code:

```
#include<stdio.h>
#include<stdio.h>
bool is_prime(int num)
{
  static int i=2;
  if(num==0||num==1)
  {
    return false;
  }
  if(num==i)
    return false;
  }
  else
  return is_prime(num);
  }
```

```
int main()
{
  int num;
  if(is_prime(num))
  printf("%d is a prime number\n", num);
  }
 else
  {
  printf("%d is not a prime number\n", num);
  }
 }
 return 0;
}
```

10. Given string a palindrome or not using recursion

Aim:- To print the program to check the whether string to palindrome or not using recursion

Algorithm:-

1. start
2. if the string has one or zero characters, it is palindrome
3. if the first and last characters of the string are unequal, it's not a palindrome
4. recursively check if the substring that excludes the first and last character is palindrome

Code:-

```
#include <stdio.h>
bool func(char *string start, int end) {
```

```
    if (start >= end) {
      return true;
    }
  if (str[start] != str[end]) {
     return false; }
  return func (str, start+1, end-1); }

int main() {
    char str[100];
    scanf ("%s", str);
  int length = strlen(str);
  if (func (str, 0, length-1)){
  printf("%s", str);
}
  printf("%s is not palindrome", str);
  } return 0;
    }
```