Ten

Recursion

[A] What will be the output of the following programs:

(a)
```
# include <stdio.h>
int main( )
{
    printf ( "C to it that C survives\n" ) ;
    main( ) ;
    return 0 ;
}
```

Output:

The message will get printed indefinitely.

(b)
```
# include <stdio.h>
int main( )
{
    int  i = 0 ;
    i++ ;
    if ( i <= 5 )
    {
        printf ( "C adds wings to your thoughts\n" ) ;
        exit ( 0 ) ;
        main( ) ;
    }
    return 0 ;
```

```
}
```

Output:

```
C adds wings to your thoughts
```

[B] Attempt the following:

(a) A 5-digit positive integer is entered through the keybo write a recursive and a non-recursive function to calcu sum of digits of the 5-digit number.

Program:

```
/* Calculate sum of digits of a five-digit number with/without recursio
# include <stdio.h>

int sum ( int ) ; /* Function without recursion */
int rsum ( int ) ; /* Function with recursion */
int main( )
{
    int s, rs ;
    int n ;

    printf ( "Enter number" ) ;
    scanf ( "%d", &n ) ;

    s = sum ( n ) ; /* Function call without recursion */
    printf ( "Sum digits without using recursion is %d\n", s ) ;
    rs = rsum ( n ) ; /* Function call with recursion */
    printf ( "Sum of digits using recursion is %d\n", rs ) ;

    return 0 ;
}

int sum ( int num ) /* Function without recursion */
{
    int remainder, sum = 0 ;
```

```
    /* This time our code is very short because we can now use a while"
       clause which was not used in the earlier instance of this program */

    while ( num > 0 )
    {
        remainder = num % 10 ;  /* Calculate remainder */
        sum = sum + remainder ;  /* update sum */
        num = num / 10 ;  /* Remove last digit */
    }

    return ( sum ) ;
}

int rsum ( int num )  /* Function with recursion */
{
    int  sum = 0 ;
    int  remainder ;

    if ( num != 0 )
    {
        remainder = num % 10 ;  /* Calculate remainder */
        sum = remainder + rsum( num / 10 ) ;  /* Recursive call */
    }

    return sum ;
}
```

(b) A positive integer is entered through the keyboard, write a program to obtain the prime factors of the number. Modify the function suitably to obtain the prime factors recursively.

Program:

```
/* Find Prime Factors of a number recursively */
# include <stdio.h>
```

```
void factor ( int ) ;
int main( )
{
    int  num ;

    printf ( "Enter a number" ) ;
    scanf ( "%d", &num ) ;

    printf ( "Prime factors are:" ) ;
    factor ( num ) ;
    return 0 ;
}

void factor ( int n )
{
    static int i = 2 ;

    if ( i <= n )
    {
        if ( n % i == 0 )
        {
            printf ( "%d ", i ) ;
            n = n / i ;
        }
        else
            i++ ;

        factor ( n ) ; /* Recursive call */
    }
    return ;
}
```

(c) Write a recursive function to obtain the first 25 numbers of a Fibonacci sequence. In a Fibonacci sequence the sum of two successive terms gives the third term. Following are the first few terms of the Fibonacci sequence:

1 1 2 3 5 8 13 21 34 55 89....

Program:

```
/* Generate first 25 terms of a fibonacci sequence */
# include <stdio.h>

void fibo ( int, int ) ;
int main( )
{
    int i, t, old = 0, current = 1, new ;

    printf ( "%d\t%d\t", old, current ) ;
    fibo ( old, current ) ;

    return 0 ;
}

void fibo ( int old, int current )
{
    static int  terms = 2 ;
    int  new ;

    if ( terms < 20 )
    {
        new = old + current ;

        printf ( "%d\t", new ) ;
        terms = terms + 1 ;
        fibo ( current, new ) ;
    }
    else
        return ;
}
```

(d) A positive integer is entered through the keyboard, write a function to find the binary equivalent of this number :

(1) Without using recursion
(2) Using recursion

Program:

```
/* Binary equivalent of a decimal number */
# include <stdio.h>

int binary ( int ) ;
int main( )
{
    int num ;

    printf ( "\nEnter the number: " ) ;
    scanf ( "%d", &num ) ;

    binary ( num ) ; /* Function call */

    return 0 ;
}

/* function to convert decimal to binary */
int binary ( int n )
{
    int r ;

    r = n % 2 ;
    n = n / 2 ;
    if ( n == 0 )
    {
        printf ( "\nThe binary equivalent is %d", r ) ;
        return ( r ) ;
    }
    else
        binary ( n ) ; /* Recursive call */
    printf ( "%d", r ) ;
}
```

(e) Write a recursive function to obtain the running sum of first 25 natural numbers.

Program:

```
/* Program to obtain running sum of natural numbers */
# include <stdio.h>

int getsum ( int ) ;
int main( )
{
    int  s ;

    s = getsum ( 0 ) ;
    printf ( "The sum of first 25 natural numbers is %d\n", s ) ;

    return 0 ;
}

int getsum ( int n )
{
    int sum = 0 ;
    if ( n == 25 )
        return sum ;

    sum = n + getsum ( ++n ) ;
    return ( sum ) ;
}
```