No of Pages : 4

4915

Roll No: 22Mx109
(To be filled in by the candidate)

Course Code : 20MX12

PSG COLLEGE OF TECHNOLOGY, COIMBATORE 641 004

SEMESTER EXAMINATIONS, JANUARY 2023

MCA Semester : 1

20MX12 STRUCTURED PROGRAMMING CONCEPTS

Time : 3 Hours **Maximum Marks : 100**

INSTRUCTIONS:
1. Answer **ALL** questions. Each question carries 25 Marks.
2. Subdivision (a) carries 3 marks each, subdivision (b) carries 10 marks each and subdivision (c) carries 12 marks each.
3. Course Outcome : Table

Qn.1	CO1.	Qn.2	CO2.	Qn.3	CO3.	Qn.4	CO4.

1. a) Compare and Contrast Bottom-Up and Top–Down approaches for Structured Programming.

   b) Explain the essences of 'Structured Programming Concepts' and its benefits in applications development.

   c) i) Discuss the various criteria for classification of Programming Languages. Based on these criteria fit the position of Programming Language C in the classification chart. Mention when to use and when not use C for software development. [6]

   ii) Discuss the various properties of good programming language in the perspective of programmer and language developer. Enumerate reasons for the popularity of a programming language. Give simple illustrations in C for the above said points [6]

2. a) Distinguish Iteration from Recursion with an example.

   b) Compare and contrast the following programming language constructs:

   i. Expressions and Statements [4]

   ii. Iterative structure Statements and its sequence control [6]

   c) i) An element of a 2D array is a saddle point if it is the "maximum" in its column and the "minimum" in its row. Assume all array elements are having distinct values. For example the element with index *(1, 1)* is a saddle point for the matrix:

$$\begin{matrix} 1 & 2 & 3 \\ 7 & 5 & 6 \\ 7 & 4 & 9 \end{matrix}$$

Complete the function below to check if the element *(r,c)* is a saddle point for the matrix *M* of size *n x n*.

```
int isSaddlePt(int M[   100 ][100] , int n, int r, int c   ){
int i, j, flag =    1;
for(i  = 0; i < n; i  ++)
      if ( __________   M[r][c] ) {flag = 0;   ______;}
for(j = 0; j < n; j++)
      if ( __________   M[r][c] ) {flag = 0;   ______;}
return flag;
}
```

Complete the program fragment below to print all the saddle points of the matrix *M* having size *n x n*. Assume that value of *n*, and elements of the matrix *M*, are already available.

```
int M[100][100],    n,  i = 0, j  = 0, flag =   0;
for (i = 0; i < n; i++) {
      for (j = 0; j < n; j++) {
            if ( ______________________________){
                  printf( "M[%d][%d] is a saddle point.     \n",  _____);
                  flag =  1;  ______;      /* there can be only one */
            }                              /* saddle point in a row */
      }
      if ( flag == 1  )  {
            ______;               /* there can be only one */
            ________;             /* saddle point in a column */
      }
}
```

[OR]

ii) Write a C program that accepts an array A of n integers and rearrange the elements of A such that it satisfy the following inequalities A[0] < A[1] < · · · < A[m − 1] < A[m] > A[m + 1] > A[m + 2] > · · · > A[n − 1] for some (unknown) index m in the valid range. Let us call such an array a *hill-valued* array. The sequence A[0],A[1], . . . ,A[m−1],A[m] is called the ascending part of the hill, and the remaining part A[m],A[m+1], . . . ,A[n −1] is called the descending part of the hill. The element A[m] is the peak of the hill and is the largest element in the array

3. a) In the following C code snippet, fill in the blanks so that the program will print the values as mentioned in the comment section

```
#include<stdio.h>
int main()
{
    int a[] = {10,5,21,51,3,2,19}, *p;
    p =        (1)____;
    printf("%d\n",*p);  // It will print 10
    p=p+*(p+4);
    printf("%d\n",*p);  // What will be printed line? (2) ____
    p = (3)____         ;
    printf("%d\n",*p);  // It will print 3
    p= (4)____          ;
    printf("%d\n",*p);  // It will print 19
    p=p-*(a+1);
    printf("%d\n",*p);  // What will be printed above line? (5)____
    return 0;
}
```

b) i) In this C declaration "long *(*(*(*z)(void))[7])(void); ", What is z? Give syntax for the use of 'z'. [3]

ii) The following C function is used to compute the product of two matrices, with each matrix represented as a 2-d array. The function takes as parameters a 2-d array A with n_r_A rows and n_c_A columns and a 2-d array B with n_r_B rows and n_c_B columns. The function returns a pointer to the first element of the product matrix C. Fill in the missing lines using pointer notations. Find the output and describe the output generated by the following program [7]

```
int ** mult(int **a, int n_r_a,int n_c_a,int **b,int n_r_b, int n_c_b)
{
  int  ------ ; // declare suitable variable  for returning results
  int sum,i,j,k;
  c= ________________________________; // dynamic memory allocation

  for(i=0;i<n_r_a;i++)
    {
      c[i]= ----------------------------------------; //dynamic memory allocation
      for(j=0;j<n_c_b;j++)
      {
        c[i][j]=0;
        for(k=0;k< -------;k++)
                 (*(*(c+i)+j)= ----------------------------------------;
      }
    }
  return -----------;
}
```

c) i) Discuss the following aspects of structured programming with illustrations from C and compare them

a. Call by Value and Call by Reference [3]

b. Storage Classes [3]

c. Arrays and Structures [3]

d. Binary Files and Text Files. [3]

(OR)

ii) a. Give the uses of Name Spaces, 'volatile' and 'const' keywords [4]

b. Trace the following program and find the final output [8]

```
# include <stdio.h>
int funct1 (int );
int funct2(int);
int main() {
                int a=0,count;
                static int b=1;
                          for ( count =1; count < = 5 ; ++count)
                 {   b+=funct1(a++) + funct2(a++);
                    printf("%d",b++);
                 }
              return 0;
          }
int funct1( int a)
 {      int b;
        b= funct2(a++);
       return b++;
 }
int funct2(int a)
 {      static int b= 1;
       b+=1;
       return(b++  +  a++);
 }
```

4. a) How to decide when to use and when not use Scripting and System Programming language for application development?

b) Compare and contrast the features of system programming languages and scripting languages in detail.

c) i) Explain the features offered by Static library, Dynamic library and low level programming in C for application development. [8]

ii) Discuss the role of Markup languages in Web applications. [4]

/END/

FD/RL

22MX609

PSG COLLEGE OF TECHNOLOGY, COIMBATORE -641004
DEPARTMENT OF COMPUTER APPLICATIONS
20MX12 STRUCTURED PROGRAMMING CONCEPTS SPC - CA I

I Semester MCA **Test – I**
Duration: 90 minutes 30-11-2022 **Max. Marks: 50**

1A. Give tips to take decisions when to use and when not use Interpreter and Compiled form of programming Languages for software development? [3]

1. B. Explain the essences of 'Structured Programming Concepts' and its benefits in applications development. [7]

1.C. i. Discuss the various criteria for classification of Programming Languages. Based on these criteria fit the position of Programming Language C in the classification chart. [7]

ii. Briefly discuss when to use and when not use C Language for software development. [8]

2.A.
i. Consider the two variables x and y with values 1 and 2 respectively. What is the value of x after executing the expression x=(x=(x=2,y=3+x)); 3
ii. Consider the following programs. Write the values of count, sum and n at the end of these programs?

```
#include < stdio.h > int
main()
{
     int n = 10, count = 0, sum = 0;
     while(n-- > 0)
     {
          n /= 2;
          count++; sum += n;
     }
     return 0 ;
}
```

count = 3
sum = 5
n = -1

3

2.B . The following program is supposed to insert a new integer value x into an already sorted (in ascending order) array A containing n distinct integers. You can assume that x does not already exist in A, and there is space available to insert x in A. For example, assume that n is 10, and A has the elements 10, 20, 30, 40, 60, 70, 80, 90, 100, 110, and x is 56. After insertion of x, the array would become 10, 20, 30, 40, 56, 60, 70, 80, 90, 100, 110, and n would be 11. Fill up the missing lines in the program so that the program inserts x in the sorted array A.

```
#include <stdio.h>
int main()
{
  int x, i = 0, j, n, A[100];
  scanf("%d%d", &n, &x);
  for (j = 0; j < n; j++)
     scanf("%d", &A[j]);
```

① while (A [i] < x && i

② for (j = n ; j > i ; j--)

③ A [j] = A [j-1]

④ A [i]

⑤ return 0

```
(1) while ( A[i] <x ) (1)
        i++; /* find position after which to insert */
(2) for ( j=n ; ____________ ; j--) /* make space for inserting x */
(3)     A[j] = __________;
    n++;
(4) A[i] = x; /* insert the element at the required place */
    for (i = 0; i < n; i++)
        printf("%d ", A[i]);
(5) return  0 ;  }
```

[7]

2.C. i.

1. Write a C program to find all Saddle points of a given matrix along with their position details. [7]

- For the matrix M1, the (3, 1) cell is a saddle point because 7 is the smallest value in row 3 and the largest value in column 1.
- For the matrix M2, the (2, 2) cell is a saddle point because 5 is the smallest value in row 2 and the largest value in column 2.
- The matrix M3 does not have a saddle point. No cell is simultaneously the smallest value in its row and the largest value in its column.

M1

1	2	3
4	5	6
7	8	9

M2

9	1	2
8	5	7
3	4	6

M3

8	1	9
7	2	6
3	4	5

2. Write an in-place C program that accepts a two dimensional array A of integers represented as int A[m][n], rotate it 90 degrees **anti-clockwise**.. (In-place means minimal extra memory to be used, i.e., doesn't make a new array to copy into) [8]

Input Matrix

1	2	3
4	5	6
7	8	9

Rotating by 90 degree in Anticlockwise $\Longrightarrow$

Output Matrix

3	6	9
2	5	8
1	4	7

Input Matrix

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Rotating by 90 degree in Anticlockwise $\Longrightarrow$

Output Matrix

4	8	12	16
3	7	11	15
2	6	10	14
1	5	9	13

[OR]

ii. Explain any **FIVE** of the following constructs with illustrations from C language, when to use and when not use while programming by discussing the principles behind the constructs. [5 X 3 =15]

1. Expressions vs. statements ✓2. Literals, constants and variables
✓3. Header files and Library files ✓4. printf ( ) and scanf ( ).
✓5. Sequence controls and Iterative statements ✓6. String Handling Functions

### END##

PSG COLLEGE OF TECHNOLOGY, COIMBATORE -641004
DEPARTMENT OF COMPUTER APPLICATIONS
20MX12 STRUCTURED PROGRAMMING CONCEPTS

21MX222

I Semester MCA **Test – I**
Duration: 90 minutes **Max. Marks: 50**

1 A.
Give tips to take decisions when to use and when not use Interpreter and Compiled form of programming Languages for software development? [3]

1. B.
Explain the essences of 'Structured Programming Concepts' and its benefits in applications development. [7]

1.C. i. Discuss the various criteria for classification of Programming Languages. Based on these criteria fit the position of Programming Language C in the classification chart. [7]

ii. Briefly discuss when to use and when not use C Language for software development. language. [8]

2.A. Consider the following C program. Select the correct option(s) for the output.

```
 #include <stdio.h>
main()
{ int m, n, temp;
 printf("Give integers m and n: ");
 scanf("%d, %d", &m, &n);
do
{
   if (m < n) n = n - m;
   if (n < m) {temp = m; m = n; n = temp;}
   if (m > n) printf( "!");
} while (m < n); }
```

Which of the following is true for the number of "!"s printed by the program?

A. For some values of 'm' and 'n', exactly one "!" is printed.
B. For all values of 'm' and 'n', no "!" is printed.
C. For all values of 'm' and 'n', the number of "!"s printed is either 1 or a positive even number.
D. For all values of 'm' and 'n', the number of "!"s printed is a positive odd number. [3]

2.B . A number is Armstrong number if the sum of its digits raised to the third power is equal to the number itself. For example, 371 is an Armstrong number, Since 33 + 73 + 13 = 371. In the following program user is supposed to enter a limit and the program prints all the Armstrong numbers from 1 to the user specified limit. Complete the program.

```
 #include <stdio.h>
int main()
 { int number=1, originalNumber, indicator, result = 0, limit;

 printf("Enter the limit: "); /* Read the limit from the user*/
scanf("%d", &limit);
 while(_original number != 0_)
  {           originalNumber = number; result = 0;
/* Loop for updating the result */
while (_original number > 0_)
    { indicator = _original number % 10_;
     result +=_indicator * indicator * indicator_;
    originalNumber = _original num/10_;      }
```

PSG COLLEGE OF TECHNOLOGY, COIMBATORE -641004
DEPARTMENT OF COMPUTER APPLICATIONS
20MX12 STRUCTURED PROGRAMMING CONCEPTS

I Semester MCA
Duration: 90 minutes **Test – I**
Max. Marks: 50

```
if( rsult== number )
        printf("%d is an Armstrong number.",number);
    else print not an arm number
}
return 0;
}
```

2.C. i. [7]

1. Write a C program to find all Saddle points of a given matrix along with their position details. [7]

- For the matrix M1, the (3, 1) cell is a saddle point because 7 is the smallest value in row 3 and the largest value in column 1.
- For the matrix M2, the (2, 2) cell is a saddle point because 5 is the smallest value in row 2 and the largest value in column 2.
- The matrix M3 does not have a saddle point. No cell is simultaneously the smallest value in its row and the largest value in its column.

M1

1	2	3
4	5	6
7	8	9

M2

9	1	2
8	5	7
3	4	6

M3

[illegible]	1	9
7	2	6
3	4	5

2. Write an in-place C program that accepts a two dimensional array A of integers represented as int A[m][n], rotate it 90 degrees **anti-clockwise**.. (In-place means minimal extra memory to be used, i.e., doesn't make a new array to copy into) [8]

Input Matrix

1	2	3
4	5	6
7	8	9

Rotating by 90 degree in Anticlockwise

Output Matrix

3	6	9
2	5	8
1	4	7

Input Matrix

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Rotating by 90 degree in Anticlockwise

Output Matrix

4	8	12	16
3	7	11	15
2	6	10	14
1	5	9	13

[OR]

ii. Explain the following constructs with illustrations from C language, when to use and when not use while programming by discussing the principles behind the constructs.

1. Expressions vs. statements
2. Literals, constants and variables
3. Header files and Library files
4. printf ( ) and scanf ( ).
5. Sequence controls and Iterative statements

[5 X 3 =15]

21mx222

PSG COLLEGE OF TECHNOLOGY, COIMBATORE -641004
DEPARTMENT OF COMPUTER APPLICATIONS
20MX12 STRUCTURED PROGRAMMING CONCEPTS

I Semester MCA — **Test –II**
Duration: 90 minutes — **Max. Marks: 50**

1. A. Compare and contrast Client-side scripting and Server-side scripting for the development of dynamic Web Applications [3]

   B. Discuss the features of Scripting languages and Mark Up languages and their fitness for Web Apps development. [10]

   C.

   I. Discuss the various criteria for the comparison of Script Languages and System programming Languages with their applications and domains. [7]

   II. Why C is considered as one of the suitable programming languages for developing System software and tools. [5]

2. A. In this C declaration **"long *(*(*(*z)(void))[7])(void);** ", What is z? Give syntax for the use of 'z'. [3]

z is a pointer to a function taking no arguments returning to an array of 7 pointers to a function taking no arguments returning a pointer to long

2. B

Explain the following constructs with illustrations from C language, when to use and when not use while programming by discussing the principles behind the constructs.

1. Call by Value vs. Call by References [3]
2. Storage Classes [4]
3. Pointers vs. Arrays [3]

2.C.

i. Write a C program to accept the two matrices of integers. Create a third matrix such that corresponding elements are concatenated and indicate with special symbol (∞) if the resultant number is outside of range. Use pointers and pointer arithmetic operations

```
e.g  11   33   44       22   44   5        1122   3344   445
     55  888   77   +   66  999  88    =   5566    ∞     7788
     22   66 1111       22   77 222        2222   6677    ∞
```

[OR]

PSG COLLEGE OF TECHNOLOGY, COIMBATORE -641004
DEPARTMENT OF COMPUTER APPLICATIONS
20MX12 STRUCTURED PROGRAMMING CONCEPTS

I Semester MCA	**Test –II**
Duration: 90 minutes	**Max. Marks: 50**

ii. You are given an m × n array A of integers, each column of which is a sorted list of size m. Your task is to merge the n sorted lists and store the merged list in a one-dimensional array B. It is given that each column does not contain repetition(s) of integers, that is, the m integers in each sorted list are distinct from one another. However, integers may be repeated in different rows. During the merging step, you must remove all these repetitions. Use dynamic memory allocated arrays and pointer arithmetic operations for processing. Print the arrays before and after operations

Input A

```
1   2   3   4
3   5   6   7
5   7   8   9
```

Output B

```
1   3   5   2   7   6   8   4   9
```

##END##

4702

No of Pages : 3

Course Code : 20MX12

Roll No: 21MX222
(To be filled in by the candidate)

PSG COLLEGE OF TECHNOLOGY, COIMBATORE - 641 004

SEMESTER EXAMINATIONS, MARCH 2022

MCA Semester: 1

20MX12 STRUCTURED PROGRAMMING CONCEPTS

Time: 3 Hours **Maximum Marks: 100**

INSTRUCTIONS:

1. Answer **ALL** questions. Each question carries 25 Marks.
2. Subdivision (a) carries 3 marks each, subdivision (b) carries 10 marks each and subdivision (c) carries 12 marks each
3. Course Outcome : Table

Qn.1	CO1.	Qn.2	CO2.	Qn.3	CO3.	Qn.4	CO4.

1. a) Give tips to take decisions when to use and when not use Interpreter based and Compiler based programming Languages for software development? (3)

   b) Discuss the various criteria for classification of Programming Languages. Based on these criterions fit the position of Programming Language C in the classification chart. Mention when to use and when not use C Language for software development. (10)

   c) Discuss the various properties of good programming language with respect to programmer and language developer perspective. Enumerate reasons for the popularity of a programming language. Give simple illustrations in C for above said points (12)

2. a) Distinguish Iteration from Recursion with an example. (3) [illegible]

   b) Compare and contrast the following programming language constructs:

      i. Expressions and Statements [6]

      ii. Else – If statements and Switch statements [4]

   c) i) A number is Armstrong number if the sum of its digits raised to the third power is equal to the number itself. For example, 371 is an Armstrong number, since $3^3 + 7^3 + 1^3 = 371$. In the following program user is supposed to enter a limit and the program prints all the Armstrong numbers from 1 to the user specified limit. Complete the program.

```
#include <stdio.h>
int main()
 {
 int number=1, originalNumber, indicator, result = 0, limit;
 printf("Enter the limit: "); /* Read the limit from the user*/
 scanf("%d", &limit);
 while(__orginal numberl! =0__)
                  ↳ no space
 {
 /* Loop for updating the result */
 originalNumber = number; result = 0;
 while (__original ●Number >0__)

 {
 indicator = __original number % 10__;
 result += __indicator * Indicator * indicator;__
 originalNumber = __original number / 10__;

 }
 if(__result == number__)
 printf("%d is an Armstrong number.",number);
  else  printf(" not a armstong number
 }
 return 0;
 }
```

[OR]

ii) Write a C program that accepts an array A of n integers and your program rearrange the elements of A such that it satisfy the following inequalities A[0] < A[1] < · · · < A[m − 1] < A[m] > A[m + 1] > A[m + 2] > · · · > A[n − 1] for some (unknown) index m in the valid range. Let us call such an array a *hill-valued* array. The sequence A[0],A[1], . . . ,A[m−1],A[m] is called the ascending part of the hill, and the remaining part A[m],A[m+1], . . . ,A[n −1] is called the descending part of the hill. The element A[m] is the peak of the hill and is the largest element in the array

3. a) Given an array, the reverse Array function tries to reverse it using pointers. Fill it up the blanks in the below code to achieve the functionality.

```
Example:
Input Array is {-1, 5, -8, -4, 5, 3} and Output is {3, 5, -4, -8, 5, -1}
#include <stdio.h>
// Function to swap two memory contents
void swap(int* a, int* b)
{ int temp = *a; *a = *b;*b= temp;}                       temp= *a

void reverseArray(int array[], int array_size)
{                                                          *a= *b
// leftPointer pointing at the beginning of the array
 int *leftPointer =       (1) ;                             *b= temp
// rightPointer pointing at end of the array
int *rightPointer =    (2) ;
while (          (3)          )
{
swap(leftPointer, rightPointer);
         (4)        ;
         (5)        ;
}
}
```

b) i Compare and contrast Void Pointer and NULL Pointer with an example [3]

ii. Find the output and describe the output generated by the following program [7]

```
# include <stdio.h>
int  funct1 (int );
int  funct2(int);
main() {
                int a=0,b=1,count;
              for ( count =1; count < = 5 ; ++count)
                {   b+=funct1(a++) + funct2(a++);
                    printf("%d",b++);
                }
          }
int  funct1( int a)
  {         static  int b;
             b= funct2(a++);
            return b++;
   }
int  funct2(int a)
  {         static int b= 1;
             b+=1;
             return(b++  +  a++);
  }
```

0

b=

count=1	a=2	b=8
count=2	a=4	b=28
count=3	a=6	b=60
count=4	a=8	b'=104
count=5	a=10	b=160

c) i. Discuss the following aspects of structured programming with illustrations from C and compare the various alternatives

1. Call by Value and Call by Reference [4]
2. Storage Classes [5]
3. Functions and Macros [3]

[OR]

ii. Write a C program to sort two dimensional array of strings using pointers and dynamic memory. Identify the possible test cases for testing the program.

```
Input:
"AAAAAAA"    "ZZZ"     "YYY"
"KKKK"         "OOOO"    "PPP"
"GGG"           "HH"        "I"

Output
"AAAAAAAA"     "GGG"     "HH"
"I"                 "KKKK"  "OOOO"
"PPP"              "YYY"      "ZZZ"
```

4. a) What is the role of Scripting languages in Web Client and Server side applications development?

b) Compare and Contrast System programming Languages and Script languages in details.

c) i. Explain the features of C language suitable for System programming and C's relevancy in current programming world. [6]

ii. Discuss the role of Markup languages for Web programming. [6]

FD/RL /END/

22MX109

PSG COLLEGE OF TECHNOLOGY, COIMBATORE -641004
DEPARTMENT OF COMPUTER APPLICATIONS
20MX12 STRUCTURED PROGRAMMING CONCEPTS

I Semester MCA **Test –II**
Duration: 90 minutes **Max. Marks: 50**

1. A. Compare and contrast Client-side scripting and Server-side scripting for the development of dynamic Web Applications **[3]**

B. Discuss the various criteria for the comparison of Script Languages and System programming Languages with their applications and domains. **[10]**

C.

I. Discuss the useful features of Scripting languages and Mark Up languages and their fitness for Web Apps development **[6]**

II. Why C is considered as one of the best programming languages for developing System software and tools. **[6]**

2. A. In this C declaration **"long *(*(*(*z)(void))[7])(void);** ", What is z? Give syntax for the use of 'z'. [3] 3

2. . It is required to create a one dimensional array of strings and read a string one by one from an array element string containing only upper-case alphabetic characters and blank spaces from an array. Then, for a given value of an integer k, every alphabetic character is replaced by the k-th next alphabetic character. It is assumed that in counting the k-th next alphabet, the letter 'A' comes after 'Z'. (circular)

For example, with the value of k as 3, the string "HAPPY BIRTHDAY TO YOU" will be replaced by "KDSSB ELUWKGDB WR BRX.

Write a C program to do the above task for the given entire one dimensional array using pointers and dynamic array. [10] 5

2. C.

i. Explain the following constructs with illustrations from C language, when to use and when not use while programming by discussing the principles behind the constructs.

1. Call by Value vs. Call by References
2. Pointers vs. Arrays
3. Arrays vs. Structures
4. Storage Classes

C Program (X)
1) saddle point
2) matrix rotation 2
↳ clockwise
↳ anticlockwise

[OR]

PSG COLLEGE OF TECHNOLOGY, COIMBATORE -641004
DEPARTMENT OF COMPUTER APPLICATIONS
20MX12 STRUCTURED PROGRAMMING CONCEPTS

I Semester MCA **Test –II**
Duration: 90 minutes **Max. Marks: 50**

---

ii.a. The following function takes two null-terminated strings S, T as parameters. The function returns 0 if the two strings are equal, 1 if the first string (S) is a proper prefix of the other (T), 2 if the second string is a proper prefix of the other, -1 otherwise. Fill it the blanks. Each blank can have AT MOST one statement. Use Pointer Notation to fill the Blanks [6+3+3]

```
int compstr ( char *S, char *T )
{
  int i = 0 ;
 while (1)
 {
     if(_________________________) return 0 ;
    if (S[i] == \0) return 1 ;
    if (_________________________) return 2 ;
    if (_________________________) return -1 ;
    ++ i;
 }
}
```

b. What value does the call h(5) return, where h is defined as follows?

```
int h ( int n )  {
    if (n == 0) return 1 ;
     if (n%2 == 0) return 2 *h(n-1);
         else  return 3 *h(n-1);   }
```

c. Write the output for the successful execution.

```
#include <stdio.h>
struct abc     { int a;  int *b; int c[5]; };
void foo (struct abc *x, struct abc y)  {
    x->a = 25;
   *(x->b) = 50;
   x->c[0] = 30;
   y = *x;  }
int main ()
    {   struct abc x, y[5] ;
        int  n = 20;
        x.a = 5;
        x.b = &n;
        x.c[0] = 10;
        y[0] = x;
        foo (&x, y[0]) ;
        printf ("x: %d, %d \n", x.a, x.c[0]) ;
       printf ("y[0]: %d, %d \n", y[0].a, y[0].c[0]) ;
       printf ("n=%d\n", n) ;
}
```

## END##