Total No. of Questions—**12**] [Total No. of Printed Pages—**8**

[3862]-222

S.E. (I.T.) (First Semester) EXAMINATION, 2010

FUNDAMENTALS OF DATA STRUCTURE

(2008 COURSE))

Time : Three Hours **Maximum Marks : 100**

N.B. :— (*i*) Answer to the two Sections should be written in separate answer-books.

(*ii*) Neat diagrams must be drawn wherever necessary.

(*iii*) Figures to the right indicate full marks.

(*iv*) Assume suitable data, if necessary.

SECTION I

1. (*a*) (*i*) Compare while and do_while loop in C. [3×2]

(*ii*) Explain enumerated data type with example.

(*b*) Select *correct* choice for the output of the following code segments : [2×3]

(*i*)
```
# define M(x) x * x

main( )

  { printf ("%d", M(2 + 3));  }
```

(1) 12

(2) 11

(3) 25

(4) error

P.T.O.

(*ii*) main( )

```
{   int x;
    x = 4 + 2% –8;
    printf ("%d", x);
}
```

(1) –6

(2) 6

(3) 4

(4) None of (1), (2), (3)

(*iii*) An expression contains relational operators, assignment operators, and arithmetic operators. In the absence of parantheses; they will be evaluated in which of the following order ?

(1) assignment, relational, arithmetic

(2) arithmetic, relational, assignment

(3) relational, arithmetic, assignment

(4) assignment, arithmetic, relational

(*c*) Write a C program to find HCF and LCM of two nos. [6]

Or

2. (*a*) Compare macro and function. [4]

(*b*) Write different bitwise operators in C and explain their use. [6]

(*c*) Write output of the following statements : [8]

(*i*) printf ("%d", 3|0);

(*ii*) printf ("% 0 % x", 10, 20);

(*iii*) printf ("%d", 3 > 2 ? 1 : 0);

(*iv*) int x = 10; printf ("%d %d", ++x, x – –);

(*v*) if (–1)

printf ("Error");

else

printf ("No error");

(*vi*) printf ("%d", 1 < 2 && 3 || 0);

(*vii*) printf ("%c", 4["Param"]);

(*viii*) int A[3] [2] = {1, 2, 3, 4, 5, 6};

printf ("%d", *(*(a + 2) + 0));

3. (*a*) Describe the following declarations : [6]

(*i*) int *p[5];

(*ii*) int **q;

(*iii*) float (*p) (int no);

(*iv*) int (*q) [3];

(*v*) int * fun1 (int *x);

(*vi*) char s[10] [30] [80];

(*b*) Differentiate between call by value and call by reference parameter passing methods. [4]

(*c*) Write a C function to compare two strings. [4]

(*d*) Compare malloc and calloc functions in C language. [2]

Or

4. (*a*) Write output of the following C code : [3]

(*i*)

```
void fun (int val)
    {
      if (val == 0)
        return;
      else
        {
            fun(val –1);
                printf("%d", val);
        }
    }

main( )
{
   fun(5);
}
```

(*ii*) void main (void) [3]

```
{
  int A[4] [3] = { {2, 4, 3},
                   {6, 8, 5},
                   {3, 5, 1}
                 };
  printf("%d %d %d", *n, n[2] [2], n[3] [2]);
}
```

(*iii*) [2]

```
void print (void)
{ static int x = 1;
  printf ("%d", x);
  x ++;
}
```

```
main( )
{ print( );
  print( );
  print( );
}
```

(*b*) Write a C program to accept, display and find topper from a list of n students, using functions. [8]

5. (*a*) Classify data structures and give one example of each type. [8]

(*b*) Analyze time complexity of the following code segments : [8]

(*i*)

```
for (i = 1; i <= n; i++)
  for (j = 1; j <= m; j++)
    for(k = 1; k <= p; k++)
      x = x + 1;
```

(*ii*)
```
i = 1
while (i <= n)
{    x++;
     i++;
}
```

(*iii*)
```
int process (int no)
{
  if (no <= 0)
      return (0);
  else
      return (no + process (no – 1));
}
```

Or

6. (*a*) What do you mean by frequency count of a statement ? Explain its importance in analysis of algorithm with suitable examples. [6]

(*b*) What is space complexity of an algorithm ? Explain its importance with example. [4]

(*c*) Write time complexity of the following algorithm using θ and Ω notations : [2]

```
void disp (Node * temp)
 {
   while (temp)
  {
     temp = temp → link;
  }
}
```

(*d*) Explain the following terms with example : [4]

(*i*) Data object

(*ii*) Data type

SECTION II

7. (*a*) Show output of each pass using bubble sort to arrange the following nos in ascending order. Write pseudo C code for bubble sort : [10]

10, 9, 8, 7, 6, 5, 4, 3, 2, 1

(*b*) Explain binary search with suitable example. [6]

Or

8. (*a*) Write output of each pass of merge sort for the following list : [6]

26, 5, 77, 1, 61, 11, 59, 15, 48, 19

(*b*) Write pseudo C code of quick sort and write average and worst case time complexity. [10]

9. (*a*) Represent sparse matrix using suitable data structure. Write pseudo C algorithm to find transpose of a sparse matrix using simple/slow transpose algorithm. Analyze its time complexity. [10]

(*b*) Explain sequential memory organization with example. [6]

Or

10. (*a*) Represent sparse matrix using suitable data structure. Write pseudo C algorithm for addition of two sparse matrices. Analyze its time complexity. [12]

(*b*) Compare array and linked list. [4]

11. (*a*) Represent the following lists using generalized linked list : [4]

(*i*) ((a, b), c)

(*ii*) (a, b, c, (d, e))

(*b*) Write a C function to reverse a singly linked list by changing link pointers. [6]

(*c*) Write a C program to create doubly linked list and print the list forward and reverse using functions. [8]

Or

12. (*a*) Write a C function to add two sorted circular linked list of polynomials to form a third sorted list. Write time complexity. [12]

(*b*) Write recursive functions for : [4]

(*i*) Display SLL forward

(*ii*) Display SLL reverse

(*c*) Compare SLL and DLL. [2]