May - June - 2012

Total No. of Questions—12] [Total No. of Printed Pages—4+2

Seat No.	

[4162]-212

S.E. (I.T.) (First Semester) EXAMINATION, 2012

FUNDAMENTAL OF DATA STRUCTURE

(2008 PATTERN)

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

N.B. :— (*i*) Answers 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 wherever necessary.

SECTION I

1. (*a*) What is similarity between structure, union and enumeration ? [6]

(*b*) What do you understand by precedence of arithmetic operators ? How does it work in C program ? Give suitable example. [6]

(*c*) Differentiate while, do-while and for loop structures. [6]

Or

2. (*a*) Define pointers. How do we declare pointers ? Give its advantages. [6]

P.T.O.

(*b*) Explain the use of break and continue statements in C with suitable example. [6]

(*c*) Compare macro and function. [6]

3. (*a*) What do you mean by type definition ? Explain with suitable example. [4]

(*b*) Write a C program to interchange two variables without using third variable. [6]

(*c*) Write output of the following C code : [6]

(*i*)
```
# include <stdio.h>
main( )
{
    int x=3;
    float y=3.0;
    if(x==y)
        printf ("\nx and y are equal");
    else
        printf ("\nx and y are not equal");
}
```

(*ii*)
```
# include <stdio.h>
void main( )
{
    int i=0;
    for (; i ;)
        printf ("\nHere is some mail for you");
}
```

(*iii*)
```
# include <stdio.h>
main( )
{
    int x=4;
    while (x==1)
    {
        x = x–1;
        printf ("\n%d", x);
        – – x;
    }
}
```

Or

4. (*a*) Differentiate between malloc and calloc. [4]

(*b*) Explain various bitwise operators in C. Give example of each operator. [6]

(c) Explain the following with example :

(i) call by value

(ii) call by reference. [6]

5. (a) What do you mean by frequency count ? Explain its importance in the analysis of algorithm. [6]

(b) Define the following terms :

data object, data type, data structure. [6]

(c) What is an abstract data type ? Explain with an example. [4]

Or

6. (a) Explain different Asymptotic notations. [4]

(b) Explain primitive and non-primitive, linear and non-linear, static and dynamic, persistent and ephermal data structure. [8]

(c) Write a non-recursive C function to generate Fibonacci series. [4]

SECTION II

7. (a) Compare the Selection sort and Insertion sort with respect to :

(i) Time complexity

(ii) Passes

(iii) Storage requirement

(iv) Sort stability. [8]

(*b*) Write a non-recursive pseudo C routine to sort the numbers using Merge sort. Show all passes to sort the values using Merge Sort in ascending order : [8]

21, 5, 7 11, 35, 76, 0, 9, 27, 45.

Or

8. (*a*) What is Bubble sort ? Explain with example. [4]

(*b*) Sort the following data to ascending order using Quick sort. Show all passes with pivot : [6]

17, 8, –9, 2, 0, –5, 7, 20, 11, 15.

(*c*) Write pseudo code for Binary search with recursion. [6]

9. (*a*) What is sparse matrix. List the applications. [4]

(*b*) Explain with example simple and fast transpose. [8]

(*c*) Write a pseudo C code for performing the following string operations without using library functions :

(*i*) Reverse of a string

(*ii*) Concatenation of two strings. [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*) Explain sequential memory organization with example. [6]

11. (*a*) Write an algorithm for polynomial addition, where polynomials are represented using linked lists. [6]

(*b*) Write function that removes all duplicate elements from a linear singly linked list. [6]

(*c*) Write C pseudo code to insert and delete an element from singly linked list. [4]

Or

12. (*a*) Compare linked list to the arrays with reference to the following aspects :

(*i*) Accessing any element randomly

(*ii*) Insertion and deletion of an element

(*iii*) Utilization of computer memory. [6]

(*b*) Discuss the application of circular linked list in detail. [6]

(*c*) Represent the following GLL :

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

(*ii*) (*a*, (*b*, *c*), *d*). [4]