JAN – 2008 Question Paper

PART – A

2 Marks

1. **Define Algorithm**

   An algorithm consists of a set of explicit and unambiguous finite steps which when carried out for a given set of initial conditions produce the corresponding output and terminate in a finite state.

2. **Define Array**

   An array is a group of similar data types stored under a common name.

   Features:

   - Array elements are stored in continuous memory location
   - Array elements can be accessed with the help of index or subscript.

   Syntax :

   Data type Arrayname[Size]

   Example:

   int num[10];
   char name[10];
   float average[10];

3. **How does Switch statement differ from Nested if?**

Switch	Nested if
Switch can test only for equality	Nested if can evaluate any type of relational or logical expression
Character constants used in switch statement are automatically converted to integers	No conversion is required

Syntax:
Switch(expression){
Case constant1:
Statement sequence
Break;
Case constant2:
Statement sequence
Break;
.
.
.
Default
Statement sequence
}
	if (condition1)
	{
	if (condition2)
	{
	True statement-2;
	}
	else
	{
	False statement-2;
	}
	}
	else
	{
	False statement-1;
	}

4. **Distinguish between Break and continue.**

Break	continue
Purpose: • Can use it to terminate a case in switch statement • Used to force immediate termination of a loop, bypassing the normal loop conditional test	**Purpose:** • Forces the next iteration of the loop to take place skipping any code in between
Break statement is encountered inside a loop , the loop is immediately terminated	Continue statement is used to continue the loop
Syntax: break;	**Syntax:** continue;

5. **How is array of strings represented in C**

Array of strings are represented using two dimensional character arrays in C.

Syntax:

char Array name[Size of Left Dimension][Size of Right Dimension]

where Size of Left dimension determines number of strings
Size of Right dimension determines maximum length of each string

Example:

char str[30][80]

declares an array of 30 strings each with a maximum length of 79 characters. Each string is terminated by null character ‘/0’.

6. **How is function definition distinguished for function call?Give an example**

A number of statements grouped into a single logical unit are called a function.

Function definition contains actual logic or function body. It comprises of return type and formal parameters. Function definition is defined outside main function

Function call is written inside main function. It consist of actual parameters that is passed as arguments to function

Syntax:

```
void main()
{
        // Function prototype
        <return_type><function_name>([<argu_list>]);

        // Function Call
        <function_name>([<arguments>]);
}
// Function definition
<return_type><function_name>([<argu_list>]);
{
        <function_body>;
}
```

Program :

```
#include <stdio.h>

#include <conio.h>
```

```
void add()
{
        int a, b, c;
        clrscr();
        printf("\n Enter Any 2 Numbers : ");
        scanf("%d %d",&a,&b);
        c = a + b;
        printf("\n Addition is : %d",c);
}
void main()
{
        add();
        getch();
}
```

7. **What are File I/O functions**
    - **fopen()-**opening a file
    - **fclose()-**closing a file
    - **putc(),fputc()-** write a character to a file
    - **getc(),fgetc()-**read a character to a file
    - **fputs()-**write string to a file
    - **fgets()-**read string to a file
    - **fprintf()-**write mixed data types to a file
    - **fscanf()-**read mixed data types to a file
    - **fwrite()-**write blocks of any data type
    - **fread()-**read blocks of any data type

8. **List all Dynamic Memory allocation functions**

    Allocating memory at runtime is called **Dynamic Memory Allocation**
    - malloc() is used to allocate memory space in bytes
    - calloc() is sued to allocate multiple blocks of memory dynamically during the execution (run-time) of the program
    - realloc() to increase the size of a block of dynamically allocated memory.
    - free() to free the dynamically allocated memory

PART – B

16 Marks

11. a) i)Explain steps involved in analyzing algorithm (8)

ii) Write detailed notes on Top down design (8)

OR

11.b) i)Explain in detail about problem solving aspects (8)

ii) With appropriate examples Explain how an algorithm is analyzed for best, worst, average cases for time and space. (8)

13.a) i)Explain with example different looping statements (10)

ii) Write a C Program to print prime numbers between 1 and 100. (6)

```
#include<stdio.h>

#include<conio.h>

void main()

{

        int x,i;

        clrscr();

        printf("1\n");

        for(i=1;i<=100;i++)

        {

                for(x=2;x<=i;x++)

                {

                        if(i%x==0)

                        {

                                break;

                        }

                        else

                                continue;

                }

                if(i==x)

                printf("%d\n",i);

        }

        getch();

}
```

OR

i) Explain the difference between While and Do While with an example program(8)

ii) Develop a program that will generate the following tables with the result right justified as shown: (8)

```
Number         Number Cubed

  1                  1
  2                  8
  -                 -

 10               1000
```

Program:

```
#include <stdio.h>

void main()

{

     int i,a;

     clrscr();

     printf("\n\tNumber\t cubed\n");

     for(i=1;i<=10;i++)

    {

           a=i*i*i;

           printf("\n\t%d\t %d",i,a);

    }

  getch();

}
```

14)a) i) Explain Recursive function with an example program (8)

ii) Write a C Program that would accept a string of any number of characters. Provide the function to count the number of vowels in the given string (8)

Program :

```
#include <stdio.h>

void main()

{

     int i=0,v=0;

     char str[50];

     clrscr();
```

```
    printf("\nEnter the string........");

    scanf("%s",&str);

    while(str[i]!='.')

        {

                if(str[i]=='a'||str[i]=='e'||str[i]=='i'||str[i]=='o'||str[i]=='u'||str[i]=='A'
        ||str[i]=='E'||str[i]=='I'||str[i]=='O'||str[i]=='U')

           {

                    v=v++;

           }


                i++;

           }

           printf("\nThe Number of Vowels are %d",v);

           getch();

}
```

OR

b) Define a structure Student that consist of the following members : Reg-no, name, marks in 3 subjects (16)

i) To print maximum mark in each subject along with name of the student

ii) To calculate total and average and print Rankwise list

```
#include<stdio.h>

#include<conio.h>

struct student1

{

        int regno;

        char name[25];

        int mark1;

        int mark2;
```

```
    int mark3;
    int total;
    float average;
    char grade;
}stud[50],*pt;
void main()
{
    int n,i,j,large,large1,large2,reg,reg1,reg2,reg3,a,b,rank;
    float avg,large3,t;
    clrscr();
    printf("Enter the Number of Students:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter the Name:");
        scanf("%s",&stud[i].name);
        printf("Enter the Register Number:");
        scanf("%d",&stud[i].regno);
        printf("Enter the Mark1:");
        scanf("%d",&stud[i].mark1);
        printf("Enter the Mark2:");
        scanf("%d",&stud[i].mark2);
        printf("Enter the Mark3:");
        scanf("%d",&stud[i].mark3);
        stud[i].total=stud[i].mark1+stud[i].mark2+stud[i].mark3;
        stud[i].average=stud[i].total/3;
    }
```

```
printf("\n\tDETAILS OF STUDENTS WHO SECURED MAXIMUM MARKS IN SUBJECTS");
printf("\n\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");
        large=stud[0].mark1;
        for(i=0;i<n;i++)
        {
                if(stud[i].mark1>large)
                {
                        large=stud[i].mark1;
                        reg=stud[i].regno;
                 }
        }
        printf("\n\t\tSUBJECT 1:");
        printf("\n\t__________________________________________________");
        printf("\n\tMaximum Marks\t\tRegister Number of Canditate");
        printf("\n\t__________________________________________________");
        printf("\n\n\t%d",large);
        printf("\t\t\t%d",reg);
        large1=stud[0].mark2;
        for(a=0;a<n;a++)
        {
                if(stud[a].mark2>large1)
                {
                        large1=stud[a].mark2;
                        reg1=stud[a].regno;
                 }
        }
        printf("\n\t\tSUBJECT 2:");
```

```
printf("\n\t_____________________________________________________");
printf("\n\tMaximum Marks\t\tRegister Number of Canditate");
printf("\n\t_____________________________________________________");
printf("\n\n\t%d",large1);
printf("\t\t\t%d",reg1);
large2=stud[0].mark3;
for(b=0;b<n;b++)
{
        if(stud[b].mark3>large2)
        {
                large2=stud[b].mark3;
                reg2=stud[b].regno;
        }
}
printf("\n\t\tSUBJECT 3:");
printf("\n\t_____________________________________________________");
printf("\n\tMaximum Marks\t\tRegister Number of Canditate");
printf("\n\t_____________________________________________________");
printf("\n\n\t%d",large2);
printf("\t\t\t%d",reg2);
pt=stud;
for(pt=stud;pt<stud+n;pt++)
{
        if(pt->average<50)
                pt->grade='F';
        else if((pt->average>60)&&(pt->average<70))
                pt->grade='C';
                else if((pt->average>70)&&(pt->average<80))
                        pt->grade='B';
                else if((pt->average>80)&&(pt->average<90))
                        pt->grade='A';
                else
                        pt->grade='O';
          }
        printf("\n\t\t\tSTUDENT MARK LIST");
        printf("\n\t\t\t~~~~~~~~~~~~~~~~~~~~");
printf("\n\t==================================================================");
printf("\n\tName\tRegisterNumber\tMark1\tMark2\tMark3\tTotal\tAverage\tGrade");
printf("\=======================================================================");
        for(i=0;i<n;i++)
        {
          printf("\n\t %s",stud[i].name);
          printf("\t%d",stud[i].regno);
          printf("\t\t%d",stud[i].mark1);
          printf("\t%d",stud[i].mark2);
          printf("\t%d",stud[i].mark3);
          printf("\t%d",stud[i].total);
          printf("\t%.2f",stud[i].average);
          printf("\t%c",stud[i].grade);
        }
   getch();
}
```

15 a) i) Write a program using pointers to read in an array of integers and print its elements in reverse order (8)

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[10];
int *ptr;
int i,n;
ptr = a;
clrscr();
printf("\n Enter the Number of elements in array:");
scanf("%d",&n);
for(i=0;i<n;i++) {
 scanf("%d",&a[i]);
}
for(i=n-1;i>=0;i--) {
 printf("\n%d",*(ptr+i));
}
getch();
}
```

ii) Explain about Preprocessors (8)

OR

b) i) Describe different types of Linked List (6)

ii) Write a Program to copy content of one file to another file. Explain (10)

```
#include<stdio.h>
#include<conio.h>
void main()
{
```

```
FILE *fp,*fp1;
char c,ch;
fp=fopen("data.txt","r");
fp1=fopen("data1.txt","w");
rewind(fp);
while(!feof (fp))
{
        ch=getc(fp);
        fputc(ch,fp1);
}
fclose(fp);
fclose(fp1);
getch();
}
```

JAN – 2009 Question Paper

PART - A

2 Marks

1. **Define the term Algorithm**
   An algorithm consists of a set of explicit and unambiguous finite steps which when carried out for a given set of initial conditions produce the corresponding output and terminate in a finite state.
2. **What is meant by Program verification**

Program verification is to ensure the correctness of a program. **Verification is** set of activities that ensure that the software correctly implements a specific function.

Testing is a technique used to verify the program correctness.

3. **Write an algorithm to swap two integer values without using temporary variable**
   Step 1: Start the process
   Step 2: Get two integer values namely a, b as input.
   Step 3: Swap two numbers using the following logic:

```
        a = a + b;
        b = a - b;
        a = a - b;
```

   Step 4: Display the swapped contents
   Step 5: Terminate the process
4. **What will be the output of the program?**

```
Void main()

{

        int ch = 65;

        printf("%d %c",ch+1,ch);

}
```

OUTPUT:

```
66      A
```

Explanation:

ch is declared as int that is equal to 65.so in printf statement ch+1 is evaluated as 65+1=66 and %c represents character so ASCII equivalent of 65 is A.

5. **Write a single line conditional statement for the following:**

```
           10     if x<0
  y   =    0      if x=0
           20     if x>0
```

6. What will be the output of the program?

```
void main()
{
        int i;
        for(i=1;i<=5;i++)
        {
                if(i%2==0) continue;
                printf("%d",i);
        }
}
```

Output:

```
        1       3       5
```

Explanation:

Loop iterates for 5 times.

1st Iteration: 1%2! =0 because it has remainder then condition is false executes printf and displays 1

2nd Iteration: 2%2 ==0 because it has no remainder then continue statement skips printf and goes to next iteration

3rd Iteration : 3%2 !=0 because it has remainder then condition is false executes printf and displays 3

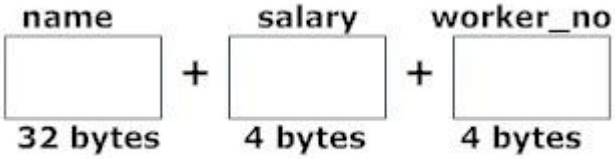
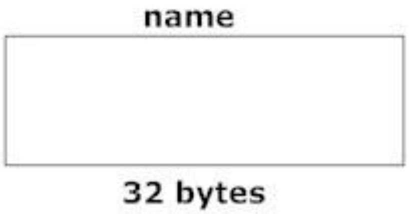
4th Iteration : 4%2 !=0 because it has no remainder , then continue statement skips printf and goes to next iteration

5th Iteration : 5%2 !=0 because it has remainder then condition is false executes printf and displays 5

7. State the difference between structure and union

Structure	Union
Structure is the collection of variables of different types under a single name that occupies different memory location.	Union is a collection of variables of different types under a single name that shares same memory location
The size of the structure corresponds to the sum of length of all members	The size of the union corresponds to the length of the largest member
The members do not share a common location so they have the different starting address.	The members share a common location so they have the same starting address.

struct job1 {
  char name[32];
  float salary;
  int worker_no;
}s;
	union job { //defining a union
	  char name[32];
	  float salary;
	  int worker_no;
	}u;

Memory Allocation:  name salary worker_no + + 32 bytes 4 bytes 4 bytes Fig: Memory allocation in case of structure **Total Bytes = 32+4+4= 40 bytes**	**Memory Allocation:**  Fig: Memory allocation in case of union

8. **What is indirection operator?**

Pointer is a variable which stores the address of another variable.one of the Pointer operator is **'*'** is called as **'Value at address'** Operator that is also called indirection operator.

Program :

```
#include<stdio.h>

int main()
{
int n = 20;
printf("\nThe address of n is %u",&n);
printf("\nThe Value of n is %d",*(&n));
}
```

Output:

```
The address of n is 1002

The Value of n is 20
```

9. **How do you allocate memory dynamically?**

Two ways to dynamically allocate memory:

- Using pointers
- Using Dynamic memory allocation functions:
  - malloc() is used to allocate memory space in bytes
  - calloc() is sued to allocate multiple blocks of memory dynamically during the execution (run-time) of the program
  - realloc() to increase the size of a block of dynamically allocated memory.

PART – B

16 MARKS

11.a) Explain the steps involved in solving problem using Top down strategy (16)

OR

b) i) List out the qualities and capabilities of good algorithm (8)

ii) Explain worst, average cases complexity of an algorithm with an example (8)

13.a) i) Explain the bitwise operators with suitable examples. (8)

ii) Admission to a professional course in subject to the following conditions (10)

a) Marks in Maths > =60
b) Marks in Physics >=50
c) Marks in chemistry >=40
d) Total in all the three subjects >=200 (or) Total in Maths and Physics >=150

Given the marks of three subjects, write a program to process the applications to list the eligible candidates

```
#include<stdio.h>

#include<conio.h>

struct student1

        {

                int regno;

                char name[25];

                int maths;

                int physics;

                int chemistry;

                int total,mptot;

                char *eligible;

        }stud[50],*pt;

void main()

{

        int n,i,j,large,large1,large2,reg,reg1,reg2,reg3,a,b,rank;
```

```
float avg,large3,t;
clrscr();
printf("Enter the Number of Students:");
scanf("%d",&n);
for(i=0;i<n;i++)
{
  printf("Enter the Name:");
  scanf("%s",&stud[i].name);
  printf("Enter the Register Number:");
  scanf("%d",&stud[i].regno);
  printf("Enter the Mark1:");
  scanf("%d",&stud[i].maths);
  printf("Enter the Mark2:");
  scanf("%d",&stud[i].physics);
  printf("Enter the Mark3:");
  scanf("%d",&stud[i].chemistry);
  stud[i].total=stud[i].maths+stud[i].physics+stud[i].chemistry;
  stud[i].mptot=stud[i].maths+stud[i].physics;
}
pt=stud;
for(pt=stud;pt<stud+n;pt++)
{
        if((pt->maths>=60)&&(pt->physics>=50)&&(pt->chemistry>=40))
        {
                if((pt->total>=200)||(pt->mptot>=150))
                        pt->eligible="ELIGIBLE";
                else
```

```
                                            pt->eligible="NOTELIGIBLE";
                              }
                    }
                    printf("\n\t\t\tSTUDENT ELIGIBILITY LIST");
                    printf("\n\t\t\t~~~~~~~~~~~~~~~~~~~~~~~~");
printf("\n\t =====================================================");
printf("\n\tName\tRegisterNumber\tMaths\tphysics\tChemistry\tTotal\tTotal(Maths+Physics)\tEligibility");
printf("\n\t ====================================================");
                    for(i=0;i<n;i++)
                    {
                      printf("\n\t %s",stud[i].name);
                      printf("\t%d",stud[i].regno);
                      printf("\t\t%d",stud[i].maths);
                      printf("\t%d",stud[i].physics);
                      printf("\t%d",stud[i].chemistry);
                      printf("\t%d",stud[i].total);
                      printf("\t%.d",stud[i].mptot);
                      printf("\t%s",stud[i].eligible);
                    }
              getch();
          }
```

OR

b) Discuss the formatted I/O statements with two examples each (16)

14.a) i) what are the categories of function? Explain (10)

ii) Write a recursive function to find factorial of a given number (4)

```
#include<stdio.h>
#include<conio.h>
```

```
void main()
{
        int num , f;
        clrscr();
        printf("Enter a number to find factorial\n");
        scanf("%d", &num);
        f = factorial(num);
        printf( "The factorial of %d is %d\n", num, f);
        getch();
 }
 int factorial(int n)
{
        if (n == 0)
          return 1;
        else
          return(n * factorial(n-1));
}
```

OR

b) i) Explain the various string handling functions with example (8)

ii) Write a program to find whether the given string is palindrome or not (8)

Program :

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int len=0,i,j;
        char str[25];
```

```
    clrscr();
    printf("Enter the String ...");
    scanf("%s",str);
    while(str[len]!='\0')
        len++;
    printf("\nThe Length of String is %d",len);
    for(i=0,j=len-1;i<len/2;i++,j--)
    {
        if(str[i]!=str[j])
        {
            printf("\nThe String %s is Not Palindrome",str);
            exit(0);
        }
    }
        printf("\n The String is %s is Palindrome",str);
        getch();
}
```

15.a) i) What is self-referential structure ? Give an Example (4)

ii) write a C program to create and display the contents of a linked list (12)

```
//Single linked list operations
# include<stdio.h>
# include<conio.h>
# include<alloc.h>
# include<stdlib.h>
struct node
{
    int number;
```

```
        struct node *next;
};
void main()
{
        struct node *p,*head;  /* p can be said as the head or a start ptr */
        void create(struct node *p);
        void display(struct node *p);
        head=(struct node *)malloc(sizeof(struct node));
        clrscr();
        create(head);
        printf("\n");
        display(head);
}
void create(struct node *list)
{
        printf("Enter the Number:");
        printf("Type 0 to Stop");
        scanf("\n%d",&list->number);
        if(list->number==0)
        {
                list->next=NULL;
        }
        else
        {
                list->next=(struct node *)malloc(sizeof(struct node));
                create(list->next);
        }
        return;
}
 void display(struct node *q)
 {
    if(q==NULL)
    {
        printf("\n\n\tEmpty Link List.Can't Display The Data");
        getch();
        goto last;
    }
    while(q!=NULL)
    {
        printf("\n%d",q->number);
        q=q->next;
    }
    last:
 }
```

OR

b) i) What is preprocessor ? How are they categorized and explain each with suitable example? (2+8)

ii) Write short notes on random access file functions. (6)

JUNE / JULY 2008 Question Paper

PART - A

2 Marks

1. What is Worst case behavior of an algorithm? How it is determined?
2. Write steps to exchange the contents of two variable x and y without using third temporary variable.

   Step 1: Start the process

   Step 2: Get two integer values namely x, y as input.

   Step 3: Swap two numbers using the following logic:

```
            x = x + y;
            y = x - y;
            x = x - y;
```

   Step 4: Display the swapped contents

   Step 5: Terminate the process

3. What are escape sequences?

   Escape sequences are typically used to specify actions such as carriage returns and tab movements on terminals and printers.

   They are also used to provide literal representations of nonprinting characters and characters that usually have special meanings, such as the double quotation mark (").

   \b – backspace

   \f – form feed

   \n – new line

   \r – carriage return

   \′ – single quote

   \″ – Double quotes

   \\ - backslash

   \v – vertical tab

   \a – alert (bell)

   \t – horizontal tab

   \? – question mark

   \0 – null

4. The straight line method of computing the yearly depreciation of the value of an item is given by

$$\text{Depreciation} = \frac{\text{Purchase Value} \quad - \quad \text{Salvage Value}}{\text{Years of service}}$$

   Write a C program to determine the salvage value of an item when the purchase price, years of service, and the annual depreciation are given

```
#include<stdio.h>
```

```
#include<conio.h>

void main()

{

            int deprice,purchaseval,salvageval,years_of_service;

            clrscr();

            printf("\n Enter the Depreciation value:");

            scanf("%d",&deprice);

            printf("\n Enter the Purchase value:");

            scanf("%d",&purchaseval);

            printf("\n Enter the Years of service:");

            scanf("%d",&years_of_service);

            salvageval=purchaseval-(deprice*years_of_service);

            printf("\n The Salvage Value is :%d",salvageval);

            getch();

 }
```

5. Assume that s1,s2,and s3 are declared as follows:
   Char s1[10]=”he”,s2[20]=”she”,s3[30],s4[40];
   What will be the output of the following statements which are executed in sequence?
   Printf(“%s”,strcpy(s3,s1));
   Printf(“%s”,strcat(strcat(strcpy(s4,s1),”or”),s2)));
   Output:
   he
   heorshe
   Explanation:
   strcpy(s3,s1)) --- s1 = he is copied to s3 and he is displayed
   strcat(strcat(strcpy(s4,s1),”or”),s2))) – strcpy(s4,s1) – he is copied to s4, strcat(strcpy(s4,s1),”or”) = he concatenated with or i.e heor, strcat(strcat(strcpy(s4,s1),”or”),s2))) – heor concatenated with she i.e heorshe
6. Difference between Structure and union ?Give an example

7. Give two benefits of Pointers to a Programmer
   - Used for dynamic memory allocation
   - Efficiently handles data in arrays
   - Reduces length and complexity of program
   - Increases execution speed
   - Saves data storage space in memory

8. What is the purpose of malloc() function? Give an example

malloc() is used to allocate memory space in bytes for variables of any valid C data type.

Syntax :

```
pointer= (data_type*)malloc(user_defined_size);
```

Example :

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
void main()
{
int a,*ptr;
a=10;
ptr=(int*)malloc(a*sizeof(int));
ptr=a;
printf("%d",ptr);
free(ptr);
getch();
}
```

PART – B

16 MARKS

13.a) i) with the help of examples explain the relational and logical operators in C (6)

ii) Write a Program in C to read a set of n single digits and converts them into single decimal integer. For example the program should convert the set of 5 digits [1,2,3,4,5] to integer 12345 (10)

```
#include<stdio.h>

#include<conio.h>

#include<math.h>

void main()

{

        int a[10],sum=0,r,i;

        int n,d,j;

        clrscr();
        printf("\n Enter the number of digits:");
        scanf("%d",&n);
        printf("\n Enter The digits:");
        for(i=0;i<n;i++)
        {
                scanf("%d",&a[i]);
        }
        for(i=0;i<n;i++)
        {

                for(j=n;j>0;j--)
                {
                        d=j-1;
                        r=a[i]*pow(10,d);
                        sum=sum+r;
                        i++;
                 }
        }
        printf("\n The Integer is %d",sum);
        getch();
}
```

OR

b) i) Write the general syntax and working of Switch statement in C with an example(6)

ii) write a program to read a list of n random numbers and count the number of positive, zero, negative numbers in the list and also print the sum of all positive numbers (10)

```
#include<stdio.h>
#include<conio.h>
void main()
```

```
{
int a[10];
int i,n,cz=0,cp=0,cn=0,sum=0;
clrscr();
printf("\n Enter the Number of elements in array:");
scanf("%d",&n);
printf("\n Enter the Numbers:");
for(i=0;i<n;i++) {
 scanf("%d",&a[i]);
}
for(i=0;i<n;i++) {
        if(a[i]==0)
           cz++;
        else if(a[i]>0)
        {
                cp++;
                sum=sum+a[i];
        }
        else if(a[i]<0)
        {
                cn++;
        }
}
printf("\n Total number of Zeros:%d",cz);
printf("\n Total number of Positive Numbers:%d",cp);
printf("\n Total number of Negative Numbers:%d",cn);
printf("\n Sum of all Positive numbers:%d",sum);
```

```
    getch();}
```

14.a) i)What is function ?Write the general format of function definition. Write a C Program with a function (6)

ii) Write a program to read array of n random numbers and sort and print the numbers both in ascending and descending order (10)

```
# include<stdio.h>

# include<conio.h>

void main()

{

    int num[25];

    int i,n,j,a;

    clrscr();

    printf("\n Enter the Number of elements to be Sorted:");

    scanf("%d",&n);

    for(i=0;i<n;i++)

    {

        scanf("%d",&num[i]);

    }

    for(i=0;i<n;i++)

    {

        for(j=i+1;j<n;j++)

        {

            if(num[i]>num[j])

            {

                a=num[i];

                num[i]=num[j];

                num[j]=a;

            }
```

```
                }
        }
        printf("\n The Ascending order of elements are :");
        for(i=0;i<n;i++)
        {
                printf("\t%d",num[i]);
        }
        printf("\n The Descending order of elements are :");
        for(i=n-1;i>=0;i--)
        {
                printf("\t%d",num[i]);
        }
        getch();
}
```

OR

b) i)What is Recursion? What are the advantages and limitations of recursion? (4)

ii)Write a program to read two matrices A and B of varying sizes and print the product of the two matrices. The program should also check the condition for multiplication. (12)

```
# include<stdio.h>
# include<conio.h>
void main()
{
        int a[25][25],b[25][25],c[25][25];
        int m,n,r,c1,i,j,k;
        clrscr();
        printf("\n Enter the Number of Rows and Columns of Matrix A:");
        scanf("%d %d",&m,&n);
```

```
printf("\n Enter the Number of Rows and Columns of Matrix B:");
scanf("%d %d",&r,&c1);
if(m!=r)
{
    printf("\nThe Matrix cant be multiplied");
}
else
{
    printf("\nEnter the values of Matrix A:");
    for(i=0;i<m;i++)
    {
        for(j=0;j<n;j++)
        {
            scanf("%d",&a[i][j]);
         }
     }
     printf("\nEnter the values of Matrix B:");
     for(i=0;i<r;i++)
     {
        for(j=0;j<c1;j++)
        {
            scanf("%d",&b[i][j]);
         }
     }
    printf("\The Multiplication of Two Matrices are");
    for(i=0;i<m;i++)
    {
```

```
                for(j=0;j<n;j++)
                {
                        c[i][j]=0;
                        for(k=0;k<m;k++)
                        c[i][j]=c[i][j]+a[i][k]*b[k][j];
                 }
            }
            printf("\n");
            for(i=0;i<m;i++)
           {    printf("\n");
                for(j=0;j<n;j++)
                {
                        printf("\t%d",c[i][j]);
                 }
            }
        }
    getch();
}
```

15.a) i) Write a function to calculate the roots of quadratic equation. The function must be passed with the value of the coefficients a, b and c as input parameters and return the roots of the equation x1,x2 as output parameters using pointers. Use the function in a main program to calculate the roots of quadratic equation (10)

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float r1,r2;
float calroot1(int *,int *,int *);
float calroot2(int *,int *,int *);
```

```
void main()
{
        int a,b,c,d;
        float root1,root2;
        clrscr();
        printf("Enter the values of a,b,c\n");
        scanf("%d %d %d",&a,&b,&c);
        root1=calroot1(&a,&b,&c);
        printf("\nThe Root1 value is:%.2f",root1);
        root2=calroot2(&a,&b,&c);
        printf("\nThe Root2 value is:%.2f",root2);
        getch();
}
float calroot1(int *i,int *j,int *k)
{
        int d;
        d=(*j)*(*j)-4*(*i)*(*k);
        printf("The value of d is %d",d);
        if(d>=0)
        {
                r1=(-(*j)+sqrt(d))/(2*(*i));
        }
         else
                printf("The roots are Imaginary");
        return r1;
}
float calroot2(int *a1,int *b1,int *c1)
```

```
{
        int d;
        d=(*b1)*(*b1)-4*(*a1)*(*c1);
        if(d>=0)
        {
                r2=(+(*b1)+sqrt(d))/(2*(*a1));
        }
         else
                printf("The roots are Imaginary");
        return r2;
}
```

ii) What are the commands available in C for random accessing a file? Explain any one with example (6)

OR

b) i) write a C program to copy the contents of one file to another (8)

ii) Why a linked list is called a dynamic data structure? What are the advantages of using linked list over arrays? Describe different types of linked lists. (8)

APRIL/MAY 2008

PART – A

2 MARKS

1. Define Algorithm
   An algorithm consists of a set of explicit and unambiguous finite steps which when carried out for a given set of initial conditions produce the corresponding output and terminate in a finite state.

2. What is O-notation?

3. In what ways does a switch statement differ from an if statement

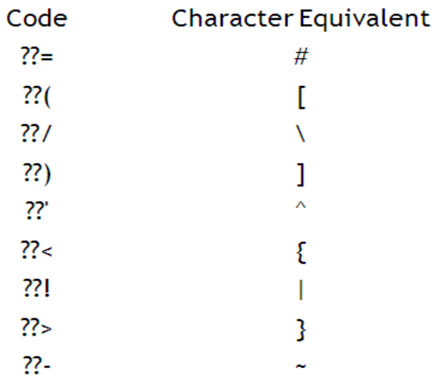
Switch	**if**
Switch can test only for equality	if can evaluate any type of relational or logical expression
Character constants used in switch statement are automatically converted to integers	No conversion is required

Syntax:
Switch(expression){
Case constant1:
    Statement sequence
    Break;
Case constant2:
    Statement sequence
    Break;
.
.
.
Default
    Statement sequence
}
	if (condition1)
	{
	    True statement-1;
	}
	else
	   {
	      False statement-1;
	    }

4. What are trigraph? How are they useful?

<u>Trigraph sequence:</u>

- To provide way to enter certain characters that are not available in keyboards
- It consists of three characters two question mark followed by another character



Code	Character Equivalent
??=	#
??(	[
??/	\
??)	]
??'	^
??<	{
??!	|
??>	}
??-	~

5. How does a structure differ from an array?

Array	Structure
Array is a collection of similar data types under common name	Structure is a collection of different data types under common name
An Array is derived data type	An structure is user data type
Syntax: Data type Arrayname[Size] **Example:** int num[10]; char name[10]; float average[10];	**Syntax:** **struct** structure_name { <data-type> element 1; <data-type> element 2; - <data-type> element n; }; **Example** struct person { char name[50]; int id; float salary; };

6. Write the limitations of using getchar and scanf function for reading strings
The major limitation is with the getchar () function is that it can only take a single character. No Format strings need to be specified.

7. Distinguish between malloc() and calloc()

malloc()	calloc()
malloc() is used to allocate memory space in bytes	calloc() is sued to allocate multiple blocks of memory dynamically during the execution (run-time) of the program
Syntax: pointer= (data_type*)malloc(user_defined_size);	**Syntax:** pointer=(data_type*)calloc(no of memory blocks, size of each block in bytes);

8. What are the advantages of using macro definition in a program?

PART – B

16 MARKS

11.a) i) Explain the concept of Program verification (10)

ii) Write short notes on Documentation of programs, debugging program and program testing (6)

OR

b) i) Explain top down design approach (8)

ii) How do you estimate the efficiency of given two algorithms (8)

13a) i) Write a program to find the number of and sum of all integers greater than 100 and less than 200 that are divisible by 7. (8)

```
#include<stdio.h>

#include<conio.h>

void main()

{

int a[10];

int i,n,count=0,sum=0;

clrscr();

printf("\nThe Integers greater than 100 and less than 200 that are divisible by 7:");

for(i=100;i<200;i++)

{

        if(i%7==0)

        {

                count++;

                sum=sum+i;

                printf("\n%d",i);

        }

 }
printf("\n Total number of Integers :%d",count);
printf("\n Sum of all Integers :%d",sum);
getch();
}
```

ii) Explain various Input and Output operations with an example. (8)

b) i) Compare in terms of their function the following pairs of statements. While and do….while and for, break and goto and break and continue (8)

ii) Write a program that will read a positive integer and determine and print its binary equivalent (8)

```
#include <stdio.h>
void main()
{
    long num, dnum, rem, base = 1, bin = 0;
    clrscr();
    printf("\nEnter a decimal integer\n");
    scanf("%ld", &num);
    dnum = num;
    while( num > 0 )
    {
        rem = num % 2;
        bin = bin + rem * base;
        num = num / 2 ;
        base = base * 10;
    }
    printf("\nInput number is               = %d", dnum);
    printf("\nIts Binary equivalent is       = %ld", bin);
    getch();
}
```

14.a) i) Write a program which will read a string and rewrite it in the alphabetical order. For example the word STRING should be written as GINRST. (10)

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

void sort_string(char*);

void main()

{

        char string[100];

        clrscr();

        printf("Enter some text\n");

        gets(string);

        sort_string(string);

        printf("%s\n", string);

        getch();

}

void sort_string(char *s)

{

        int c, d = 0, length;

        char *pointer, *result, ch;

        length = strlen(s);

        result = (char*)malloc(length+1);

        pointer = s;

        for ( ch = 'a' ; ch <= 'z' ; ch++ )

       {

          for ( c = 0 ; c < length ; c++ )

          {

                  if ( *pointer == ch )
```

```
                {
                  *(result+d) = *pointer;
                  d++;
                }
                pointer++;
          }
          pointer = s;
        }
        *(result+d) = '\0';
        strcpy(s, result);
        free(result);
}
```

ii) Explain the various string handling functions (6)

OR

b) i) Write a program to illustrate the comparison of structure variable (8)

ii) Explain the concept of Recursion with a sample recursion program (8)

15.a) i) Explain the concept of addition and deletion in a doubly linked list with an example (12)

ii) Write a nested macro that gives the minimum of 3 values (4)

```
#include<stdio.h>
#include<conio.h>
#define min(a,b) ((a>b)?b:a)
#define minthree(a,b,c) (min(min(a,b),c))
void main()
{
        int x,y,z,w;
        clrscr();
        printf("enter three numbers :\n");
        scanf("%d%d%d",&x,&y,&w);

        z=minthree(x,y,w);

        printf("Minimum of three value is %d",z);

        getch();

}
```

OR

i) Two files DATA 1 and DATA 2 contain sorted list of integers. write a program to produce a third file DATA which hold a single sorted, merged list of these two lists (12)

ii) Explain the difference between call by reference and by value (4)

JUNE 2010

PART – A

2 MARKS

1. **Define Algorithm**
   An algorithm consists of a set of explicit and unambiguous finite steps which when carried out for a given set of initial conditions produce the corresponding output and terminate in a finite state.
2. **Justify the need for Problem analysis.**
   After specifying the overall problem it is an important step to analyze the problem, when there are several different ways to organize data and devise algorithm it becomes an important step to develop criteria to recommend a choice. Hence we need to analyze the behavior of algorithm under various conditions.
3. **What do you understand by programming paradigm**

   Paradigm *is serves as pattern or model or example* that shows how something works
   A **programming paradigm** is a fundamental style of computer programming.

   Programming paradigm is framework that defines how the user conceptualized and interprets complex problems. It is also a fundamental style or approach used in Software engineering to implement a programming language.

   A **Programming Paradigm** is a model for a class of Programming Languages that share a set of common characteristics.

4. **Write an algorithm for summing up two integer values**

   Step 1: Start the process

   Step 2: Get two integer values namely a, b as input.

   Step 3: Add two numbers using the following logic:

   c = a + b;
   Step 4: Display the sum stored in C
   Step 5: Terminate the process

5. **What are various phases in program development**
   - Specification and analysis of the problem
   - Prepare algorithm, flow chart and pseudo code
   - Coding the algorithm
   - Debugging
   - Testing
   - Documentation
   - Maintenance

6. **Write short notes on Pseudo code**

- A pseudocode is semiformal English like language with a limited vocabulary that can be used to design and describe an algorithm.
- The main purpose of a pseudocode language is to define the procedural logic of an algorithm in a simple, easy to understand manner for its readers, who may or may not proficient in computer programming languages.

7. **Write short notes on relational operators**

Relational Operators support logical relationships. They are all binary operators that accept two operands and compare them. The result is logical data – the result is always true (1) or false (0).

The various relational operators in C are:

<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	equal to
!=	not equal to

8. **What is a pointer and state one application of a pointer**

Pointer is a variable which stores the address of another variable.

Application of Pointers:

- Passing Parameter by Reference
- Accessing Array element
- Dynamic Memory Allocation
- Passing Parameter to the function
- Some of the applications:
    - Passing Strings to function
    - Provides effective way of implementing the different data structures such as tree , graph, linked list

9. **Compare Union and a structure**

10. **Write a note on enumeration**

An enumeration consists of a set of named integer constants.

Syntax:

enum identifier { enumerator-list }

Example :

```
#include< stdio.h>
void main()
{
int i;
enum week{SUN,MON,TUE,WED,THUR,FRI,SAT};
clrscr();
 for(i=SUN;i<=SAT;i++)
     printf("\n%d",i);
}
```

Output :

```
0    1    2     3     4     5     6
```

PART – B

16 MARKS

11.a) Discuss various decision structures used in problem solving with suitable examples(16)

OR

b) Explain in detail various phases in top down design of programming. (16)

12.a) Write short notes on the following:

i) Rule based Programming (8)

ii) Structured Programming (8)

OR

b) What is meant by Recursion? Write a recursive algorithm to find factorial of a number and explain its working (16)

13.a) Explain in detail the various phases in the development of programming (16)

OR

b) Explain in detail how an algorithm is analyzed for best, worst, and average case of time and space complexities (16)

14.a) i) Write a C Program to arrange the given list of numbers in descending order (8)

```
# include<stdio.h>

# include<conio.h>

void main()

{

        int num[25];

        int i,n,j,a;

        clrscr();
```

```
    printf("\n Enter the Number of elements to be Sorted:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&num[i]);
    }
    for(i=0;i<n;i++)
    {
        for(j=i+1;j<n;j++)

        {
            if(num[i]>num[j])
            {
                a=num[i];
                num[i]=num[j];
                num[j]=a;
             }
         }
    }
    printf("\n The Descending order of elements are :");
    for(i=0;i<n;i++)
    {
        printf("\t%d",num[i]);
    }
    getch();
}
```

ii) Write a C Program to find the roots of a quadratic equation (8)

```
#include<stdio.h>

#include<conio.h>

#include<math.h>

void main()

{

            int a,b,c,d;

            float root1,root2;

            printf("Enter the values of a,b,c\n");

            scanf("%d %d %d",&a,&b,&c);

            d=b*b-4*a*c;

            if(d>=0)

            {

                        root1=(-b+sqrt(d))/(2*a);

                        root2=(+b+sqrt(d))/(2*a);

printf("The Roots of the values a=%d,b=%d,c=%d are \n %f %f",a,b,c,root1,root2);

             }

             else

                        printf("The roots are Imaginary");

             getch();

}
```

OR

b) What is a Structure? Define a structure called salary and another structure called allowance. Use the structure variable allowance in salary structure and write program to read data into the structure variables. (16)

```
#include  <stdio.h>

#include  <conio.h>

int main()

      {
```

```
struct allowance
        {
        int hra;
        int da;
        int ta;
        int pf;
        int gradutity;
        int esi;
        };
struct employee
        {
        struct allowance a;
        int empid;
        char empname[25];
        int salary;
        float grosspay,netpay;
        }emp[25];
int n,i;
clrscr();
printf("Enter total number of Employees\n\n");
scanf("%d",&n);
for(i=0;i<n;i++)
        {
     printf("\nEnter details of %d-th Employee\n\n",i+1);
     printf("\nEmployee Name:\n");
     scanf("%s",&emp[i].empname);
     printf("\nEmployee ID:\n");
```

```
        scanf("%d",&emp[i].empid);
        printf("\nBasic Pay:\n");
        scanf("%d",&emp[i].salary);
        printf("\nAllowances:\n");
        printf("\nHRA:\n");
        scanf("%d",&emp[i].a.hra);
        printf("\nDA:\n");
        scanf("%d",&emp[i].a.da);
        printf("\nTA:\n");
        scanf("%d",&emp[i].a.ta);
        printf("\nPF:\n");
        scanf("%d",&emp[i].a.pf);
        printf("\nGRADUTITY:\n");
        scanf("%d",&emp[i].a.gradutity);
        printf("\nESI:\n");
        scanf("%d",&emp[i].a.esi);
    emp[i].netpay=emp[i].salary+emp[i].a.hra+emp[i].a.da+emp[i].a.ta+emp[i].a.pf+emp[i].
    a.gradutity+emp[i].a.esi;
        emp[i].grosspay=emp[i].netpay-emp[i].a.pf-emp[i].a.gradutity-emp[i].a.esi;
        }
printf("EMPLOYEE DETAILS:\n");
for(i=0;i<n;i++)
        {
        printf("\n\nEMPLOYEE ID:%d\n\n",emp[i].empid);
        printf("Employee Name:%s\n\n",emp[i].empname);
        printf("Netpay:%.2f\n\n",emp[i].netpay);
        printf("Grosspay:%.2f\n\n",emp[i].grosspay);
     }
    getch();
    return 0;
}
```

15. a) Write a program using pointers to read an array of strings and print in reverse order (16)

```
#include<stdio.h>
int string_length(char*);
void reverse(char*);
void main()
{
            char string[100];
            clrscr();
            printf("Enter a string\n");
            gets(string);
            reverse(string);
            printf("Reverse of entered string is \"%s\".\n", string);
            getch();
}
void reverse(char *string)
{
            int length, c;
            char *begin, *end, temp;
            length = strlen(string);
            begin = string;
            end = string;
            for ( c = 0 ; c < ( length - 1 ) ; c++ )
              end++;
            for ( c = 0 ; c < length/2 ; c++ )
```

```
                {
                   temp = *end;
                   *end = *begin;
                   *begin = temp;
                   begin++;
                   end--;
               }
}
```

OR

b) Expian in detail various functions for basic file manipulations and operations (16)

APRIL/MAY 2011

PART – A

2 MARKS

1. What is a variable?

Variable is a data name which is used to store data and may change during program execution. Variable name is a name given to memory cells location of a computer where data is stored.

Rules for variables:

- First character should be letter or alphabet.
- Keywords are not allowed to use as a variable name.
- White space is not allowed.
- C is case sensitive i.e. UPPER and lower case are significant.
- Only underscore, special symbol is allowed between two characters.
- The length of identifier may be up to 31 characters but only the first 8 characters are significant by compiler.

Syntax : data type variable name

Example : int id;

char name[10];

float average;

2. What is the purpose of documentation?

3. What are different programming paradigms?

- Imperative / Procedural Paradigm
- Functional Paradigm
- Object based Paradigm
- Logic Paradigm

4. Define rule-based programming.

5. How do you find the time complexity of a given algorithm?

6. What is the difference between Pseudo code and Flowchart?

7. Write the different user defined data type in C

Types of User defined data types:

- Structure
- Union
- Enumeration
- typedef
- Bit Field Manipulation

8.Write a C Program to swap two numbers

```
#include <stdio.h>
 Void main()
{
                int x, y, temp;
                printf("Enter the value of x and y\n");
                scanf("%d%d", &x, &y);
                printf("Before Swapping\nx = %d\ny = %d\n",x,y);
                temp = x;
                x = y;
                y = temp;
                printf("After Swapping\nx = %d\ny = %d\n",x,y);
}
```

9. Define Union in C.
Union is a collection of variables of different types under a single name that shares same memory location
The size of the union corresponds to the length of the largest member

The members share a common location so they have the same starting address.

Syntax:

```
union union_name
{
                <data-type> element 1;
                <data-type> element 2;
                <data-type> element 3;
}union_variable;
```

Example :

```
union job
{         //defining a union
             char name[32];
             float salary;
             int worker_no;
}u;
```

Memory Allocation:

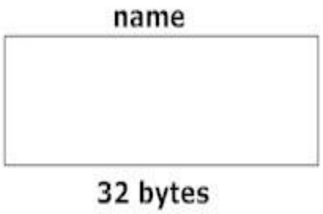


Fig: Memory allocation in case of union

10. Write the various string handling functions with their syntax.

strcat -	Concatenates two string. char *strcat(char *str1,const char *str2);

Strncat-	Concatenates not more than n characters of the string pointed to by str2 (string2).
	strcat(str1,str2,n)
	char *strncat(char *str1,const char *str2,size_t count);

strcmp -	Compares two strings. strcmp(str1,str2) compares str1 and str2 int *strcmp(const char *str1,const char *str2);
strrev -	Reverse a string. strrev(str)
strcpy -	Copies a string from source to destination. strcpy(str1,str2) copies the string pointed to by str2 to the string pointed to by str1 char *strncpy(char *str1,const char *str2);
strncpy-	strncpy(str1,str2,n) copies up to n characters from the string pointed to by str2 to the string pointed to by str1 char *strncpy(char *str1,const char *str2,size_t count);
Strcmpi-	strcmpi(str1,str2) compares str1 and str2 lexicographically without regards to case .
Strncmp-	strncmp(str1,str2,n) compares atmost n charactes of str2 to str1 lexicographically int *strcmp(const char *str1,const char *str2,size_t count);
Strchr-	strchr(str,ch) finds ch in str. Returns a pointer to the first occurrence of the character ch in str, if ch does not occur in str ,strchr returns NULL. char *strchr(const char *str1,int ch);
Strstr	strstr(str1,str2) finds the first occurrence of substring str2 in str1. char *strstr(const char *str1,const char *str2);
Strset	strset(str,ch) set all character in str to ch.
Strlen	Returns length of string Size_t strlen(const char *str)
strlwr -	Returns upper case letter to lower case. strlwr(str)

strupr -	Returns lower case letter to upper case. strupr(str)

PART – B

16 MARKS

11.a) Write short notes on the following :

i) Programming building blocks (8)

ii) Problem Analysis (8)

OR

b) Explain the various loop structures in detail (16)

12.a) Write short notes on the following:

i) Procedural Programming (8)

ii) Structured Programming (8)

OR

b) What is a Recursion? Explain in detail with an example (16)

13.a)Explain the various programming life cycle phases in detail (16)

OR

b) write an algorithm and flowchart to read 10 numbers and compute the average , max and minimum value (16)

14.a) Write a function using pointers in C that receive a sorted array of integers and an integer value and inserts the value in correct place (16)

```
# include<stdio.h>

# include<conio.h>

int pos,i;

void insert(int*,int*,int*);

void main()

{

        int number[25];

        int i,n,no,j,a,pos;
        clrscr();
        printf("\n Enter the Number of elements in Array:");
        scanf("%d",&n);
        for(i=0;i<n;i++)
        {
                scanf("%d",&number[i]);
        }
        printf("\n Enter the Number to be inserted:");
        scanf("%d",&no);
        insert(number,&no,&n);
        getch();
}
        void insert(int *num,int *no1,int *tot)
        {
                for(i=0;i<*tot;i++)
                {
                        if(*no1 < *(num+i))
                        {
                                pos=i;
                        }
                }
                pos=pos-1;
                *tot=*tot+1;
                for(i=*tot;i>=pos;i--)
                {
                        *(num+i)=*(num+(i-1));
                }
```

```
                    pos=pos-1;

                    *(num+pos)=*no1;

                    printf("\n");

                    printf("\n After insertion");

                    for(i=0;i<*tot;i++)

                    {

                              printf("\t%d",*(num+i));

                    }

          getch();

}
```

OR

b) Explain the different storage classes in C in detail (16)

15.a) Define a structure called student with name, register number and age as data members. Write a program to create an array of structures for 5 students. Read data into it and print (16)

```
#include<stdio.h>

#include<conio.h>

struct student

          {

                    int regno;

                    char name[25];

                    int age;

          }stud[50];

void main()

{

          int n,i,j;

          clrscr();

          printf("Enter the Number of Students:");
```

```
        scanf("%d",&n);
        for(i=0;i<n;i++)
        {
          printf("Enter the Name:");
          scanf("%s",&stud[i].name);
          printf("Enter the Register Number:");
          scanf("%d",&stud[i].regno);
          printf("Enter the Age:");
          scanf("%d",&stud[i].age);
        }
        printf("\n\t\t\tSTUDENT DETAILS");
        printf("\n\t\t\t~~~~~~~~~~~~~~~");
        printf("\n\t ==========================");
        printf("\n\t Name\tRegister Number\tAge");
        printf("\n\t ==========================");
        for(i=0;i<n;i++)
        {
          printf("\n\t %s",stud[i].name);
          printf("\t%d",stud[i].regno);
          printf("\t\t%d",stud[i].age);
        }
   getch();
}
```

OR

b) Explain the input and output operations on files in C with suitable example (16)

JUNE 2011

PART A

1. **What are the applications of decision structures?**

When we try to execute any program, the instructions were executed in the same order in which they appeared within the program. In practice we have number of situations

Where we may have to change the order of execution of statements based on certain conditions, or repeat a group of statements until certain specified condition are met.

By using control statement we can test condition. If condition satisfied, we can execute some set of statement, otherwise we can ignore the execution of those set of statements.

2. **When will you use a nested loop?**

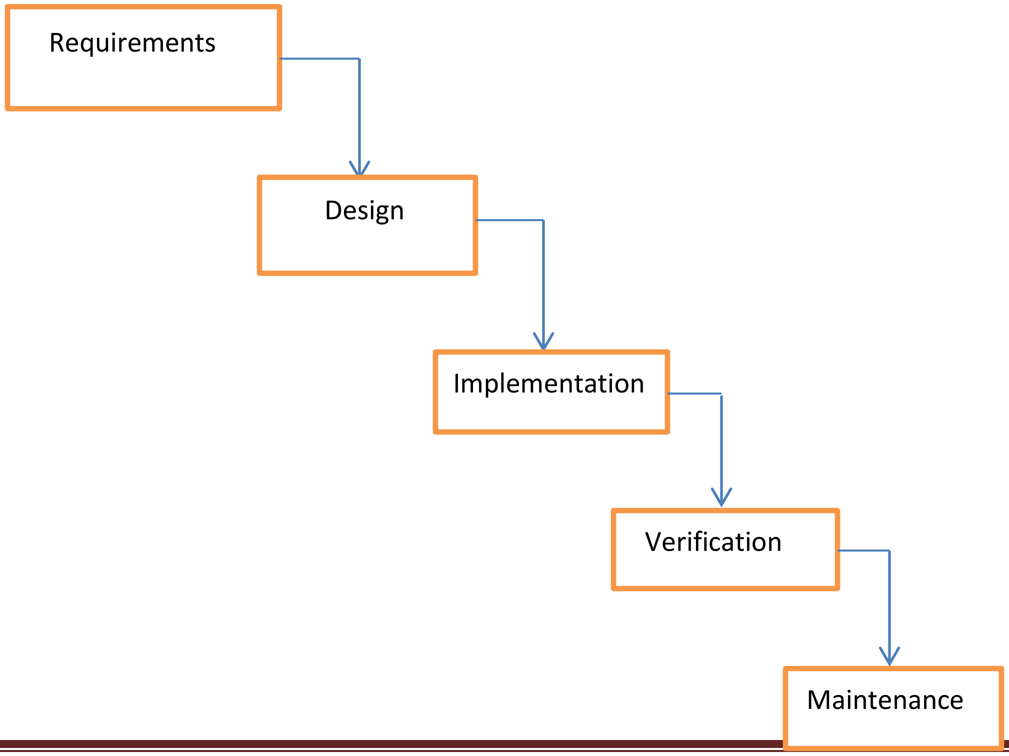
Nesting of one for loop with in another is allowed in c. ANSI C allows upto 15 Levels of nesting. However some compilers permit more.

The nesting may continue up any desired level. The loop should be properly indented so as to enable reader to easily determine which statements are contained within each for statement. For every single iteration of outer loop; the whole iteration of inner loop gets executed.

3. **Distinguish "iteration and recursion"?**
4. **What are rule-based systems?**

A logic programming consists of a series of axioms or fact, rules of inference, and a theorem or query to be proved. The output is true if facts support the query and false.

5. **Draw the programming life cycle?**



6. **Define time complexity?**

   The time complexity of a problem is the number of steps that to solve an
   Instance of the problem as a function of the size of the input, using most algorithm.

7. **Determine the output of the following code?**

```
int main()
{
        Char far*p1,p2;
        Printf(“”p1:%d\np2:”%d”,sizeof(p1),size(p2));
        Return 0;
}
```

Output:

```
P1=4;
P2=2;
```

8. **What is the type of temp below?**

```
typedef int(*test)(float*,float*)
test temp;
```

   here temp is a pointer variable to the function, that point to any function that accepts two float* as it’s arguments and return int as result.

9. **What is the benefit of using an enum rather than#define constant?**

   The use of an enumeration constant (enum) has many advantages over the traditional symbolic constant style of #define. These advantages include a lower maintenance requirement, improvement program readability, and better debugging capability.

   The first advantage is that enumerated constants are generated automatically by the compiler. Conversely, symbolic constants must be manually assigned values by the programmer.

10. **How will you find a circle in a linked list?**

PART-B

11. (a) (i) describe problem state and answer state with example. OR
    (b) Discuss the features of different programming paradigms with examples.

12. (a) write the non-recursive algorithm for finding the Fibonacci sequence and
    derive its time complexity. OR
    (b) Write the non-recursive algorithm for In order traversal of tree and derive its complexity.

13. (a) (i) Distinguish : priori analysis and posteriori analysis.
    (ii) Explain the terms: feasible solution, optimal solution and objective function.

OR

(b) (i) describe different notions used to represent algorithm complexity.
(ii) Distinguish : profiling and debugging.

14.(a) write a program to find and print the given number is odd or even. Using only one printf (output) statement, no conditional statement and no logical, relational and arithmetic operators.

```
#include<stdio.h>

#include<conio.h>

void main()
{
        char a[2][5]={"even","odd"};
        int i;
        clrscr();
        scanf("%d",&i);
        printf("%s",a[i%2]);
        getch();
}
```

(b). write a program to multiply two number without using any arithmetic operators. (hint: bitwise operations)

```
                #include<stdio.h>
int multiply(int x,int y)
{
        int product=0;
        clrscr();
        while(y!=0)
        {
                if(y&01)
                product=product+x;
                x<<=1;
                y>>=1;
        }
        return product;
        printf("the product is:%d",product);
}
void main()
{
        int a,b;
        printf("enter the two numbers:");
        scanf("%d",&a);
        scanf("%d",&b);
        printf("product is:%d",multiply(a,b));
        getch();
    }
```

15.(a) design a conventional iterative algorithm to traverse a binary tree represented in

Linked lists in post order.

(b) Write a pseudo-code for a time algorithm that generates the optimal binary search

Tree from the root table.

JANUARY 2011

PART A – 2 MARKS

1. Explain the structures of a switch-case statement?

The general syntax is:

```
Switch (expression / variable)

{

  case constant-1 :

        statement – sequence;

        break;

  case constant-2 :

        statement – sequence;

        break;

        ……

  default:

         statement- sequence;

         break;

}
```

2. Explain the need for loops in programming?

Sometime the program may need to execute group of instruction repeatedly, until some logical condition has been satisfied. This is known as looping. In looping, sequence of statements is executed until some conditions for termination of the loop are satisfied.

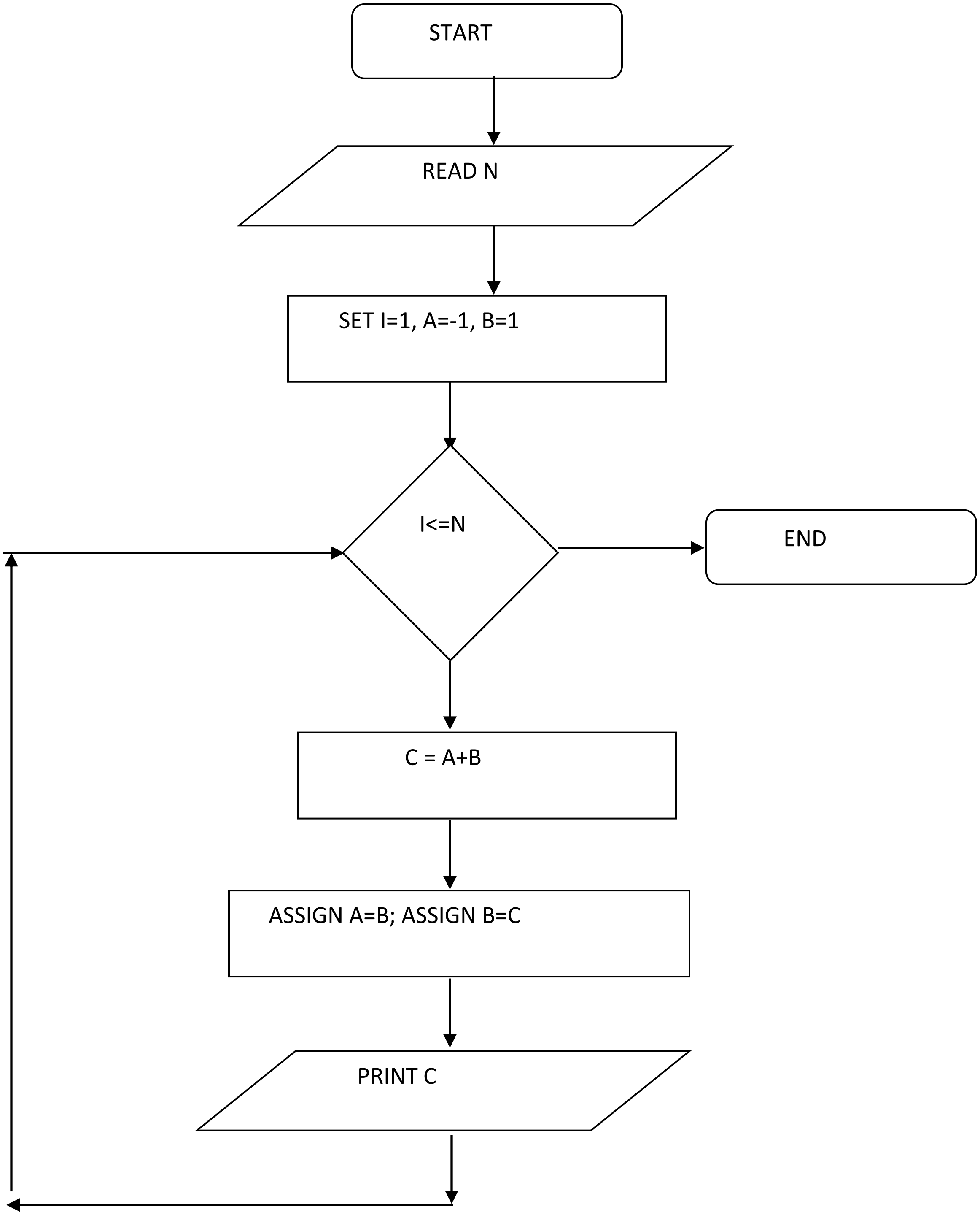
3. Differentiate logic and rule based programming?

- A logic programming consists of a series of axioms or facts, rules of inference, and a theorem or query to be proved. The output is true if the facts support the query and false otherwise. It is associated with artificial intelligence.
- A program in the rule based paradigm consists of functions. A program is a collection of functions, one of which is the top-level functions. There are no variables in functional languages, but names are allowed, no assignment, no iteration but recursion is used.

4. What is the necessary for writing a function using recursion?

Recursion is a process by which a function calls itself repeatedly until some specified condition has been satisfied. Recursion functions can be effectively used in applications in which the solution to a problem can be expressed in terms of successively applying the same solution to the subsets of the problem.

5. Draw a flow chart computing Fibonacci numbers?



6. Write a psecudo code to compute the sum of the cubes of the cubes of first ‘n’ numbers?

S = (1) + (2) +………… (N)

```
START
S=0, I=1
READ N
WHILE I<=N DO
    •  S= S+(I*I*I)
    •  I= I+1
 END MODULE
PRINT S
STOP
```

7. Write data type for representing:

(a) Class average marks.

(b) Weight of Hussains paintings.

- The data type for representing class average marks and weight of Hussians painting is float.

8. What is the need for a pointer?

- Pointers provide the means by which function can modify their calling arguments.
- Pointer support dynamic memory allocation.
- Pointer can improve the efficiency of certain routines.

9. Declare a structure that would store details of an employee and declare an instance of this structure as a pointer variable?

```
struct employee
    {
        int code;
        char name[50];
        char dept[50];
        float sal;
    };
```

10. What is stack?

A stack is a data structure in which the elements are added and removed from one end at the top. It follows last in first last in first out strategy. It has two basic operations PUSH (adding elements) and POP (removing elements).

PART B-16 MARKS

11. (a) (i) Explain the need for documentation and testing a program? (8)

11. (a) (ii) What are the building blocks of a program? (8)

(or)

11. (b) (i) Write a short notes on the various decision structures that are available in c programming? (8)

11. (b) (ii) Write short notes on the various loop structures and its application in a particular programming situation? (8)

12. (a) Write short notes on structure programming and mention some programming languages that are in this category. How do you justify that these languages satisfy the characteristics of structured programming? (16)

(or)

12. (b) (i) Write short notes on Functional programming. Name few languages that purely supports functional programming. Justify your examples? (8)

12. (b) (ii) Distinguish procedural and structured programming? (8)

13. (a) (i) Explain the need for modular programming? (4)

13. (a) (ii) What are the requirements for writing an algorithm and explain the need for analyzing an algorithm? Justify the need for expressing the time complexity using asymptotic notation explaining the asymptotic notations briefly? (12)

(or)

13. (b) (i) Write an algorithm that would check whether the elements in a given array are distinct or not. Analyze this algorithm for worst case time and space complexity? (8)

13. (b) (ii) Write an algorithm that computes the following function? f(x)=1!+2!+3!+ …+x?

14. (b) (i) Write a program in C that would add digits of a given integer. For example: input:-- 348, output = 3+4+8 =15? (8)

```
#include<stdio.h>

#include<conio.h>

void main()

{

        clrscr();

        int n,r,s=0;
```

```
                printf("Enter the value of n:\n");
                scanf("%d",&n);
                while(n>0)
               {
                        r=n%10;
                        s=s+r;
                        n=n/10;
                }
               printf("The sum of digits=%d",s);
               getch();
       }
```

14. (b) (i) Write a program in C that would convert a given decimal number to binary and octal using loop structure. Input: 10, Binary - 1010, Octal -12. (8)

```
#include<stdio.h>
#include<conio.h>
void main()
{
 clrscr();
 int i=0;int j=0,r,a[20],n1,n;
 printf("Enter the decimal number:\n");
 scanf("%d",&n);
 n1=n;
 while(n>0)
 {
  r=n%2;
  a[i]=r;
  n=n/2;
```

```
   i++;
  }
  printf("The binary number is:\n");
  for(j=i-1;j>=0;j--)
  printf("%d",a[j]);


i=0;
 while(n1>0)
 {
 r=n1%8;
 a[i]=r;
 n1=n1/8;
 i++;
 }
 printf("\nThe octal number is:\n");
 for(j=i-1;j>=0;j--)
 printf("%d",a[j]);
 getch();
 }
```

(or)

14. (b) (ii) Write a program in c that would check whether a given string is a palindrome or not. The output of this program should be a 1 if the string is a palindrome and 0 otherwise? (8)

```
#include<stdio.h>
#include<conio.h>
int stpal(char str[50]);
void main()
{
 char str[50];
 int pal;
 clrscr();
 printf("\n\n ENTER A STRING.....\n");
 gets(str);
 pal=stpal(str);
 if(pal)
 printf("\n\t THE ENTERED STRING IS A PALINDROME");
 else
 printf("\n\t THE ENTERED STRING IS A NOT PALINDROME");
 getch();
}
int stpal(char str[50])
{
int i=0,len=0,pal=1;
while(str[len]!=0)
len++;
len--;
for(i=0;i<len/2;i++)
{
 if(str[i]==str[len-i])
   pal=1;
 else
 {
   pal=0;
   break;
 }
}
return pal;
}
```

15. (a) Write functions in C using pointers that would do the following:

(i) Get an input string and computes its length? (4)

```
 #include<stdio.h>
#include<conio.h>
void main()
{
  char s[100];
  char *p;
  int length;
  clrscr();
  printf("Enter a string\n");
  scanf("%s",s);
  for(p=s;*p!='\0';p++);
  length=p-s;
  printf("Length of string is:%d",length);
  getch();
  }
```

(ii) Get an input string and find the occurrence of the character ‘c’ in the string and replace it with character ‘y’? (4)

(iii) Function takes two strings and concatenates the first string with the second?

(4)

```
 #include<stdio.h>
#include<conio.h>
void stcat(char *str1,char *str2);
void main()
{
 clrscr();
 char *str1,*str2;
 clrscr();
 printf("\n\n\t Enter the first string.....");
 gets(str1);
 printf("\n\n\t Enter the first string.....");
 gets(str2);
 stcat(str1,str2);
 printf("\n\t THE CONCATENATED STRING IS.....");
 puts(str1);
 getch();
 }
void stcat(char *str1,char *str2)
{
int i=0,len=0;
while(*(str1+len)!=0)
len++;
while(*(str1+i)!=0)
{
 *(str1+len)=*(str2+i);
 i++;
 len++;
 }
 *(str1+len)=0;
 }
```

(iv) Function takes two strings and checks whether the second string is a substring of the first string? (4)

```
  #include<stdio.h>
#include<conio.h>
int find_substr(char *listpointer,char *itempointer);
int main(void)
{
  if(find_substr("c is fun","is")!=-1)
  printf("Substring is found");
  return 0;
  }
int find_substr(char *listpointer,char *itempointer)
{
  int t;
  char *p,*p2;
  for(t=0;listpointer[t];t++)
  {
  p=&listpointer[t];
  p2=itempointer;
  while(*p2&&*p2==*p)
  {
  p++;
  p2++;
   }
  if(!*p2)return t;
}
return -1;
}
```

(or)

15. (b) Create a structures called “calendar” to store month, day, date and year (2)

Write functions in c to do the following operations in the created structure. Create an array of structures using dynamic memory allocation to hold 20 dates along with its, day and year, information and store the data randomly? (2)

(i) Given the date check whether the day is Friday and return ‘1’ if it so else return ‘0’? (4)

(ii) Given the date add ‘5’ to the date and correspondingly update the month, day and year if required? (4)

(iii) Write a recursive function that would scan the created structure array and print the complete date, day, month and year information if the date is an even number? (4)

```
#include<stdio.h>
#include<conio.h>
struct calender
{
  int date;
  int month;
  int year;
  char day[10];
}c[20];
void main()
{
int i;
clrscr();
printf("Enter the dates\n");
for(i=0;i<7;i++)
{
        scanf("%d%d%d%s", &c[i].date, &c[i].month, &c[i].year, &c[i].day);
}
```

```
printf("\n Check and Display Dates that has Friday");
for(i=0;i<7;i++)
{
        if(strcmp(c[i].day,"Fri")==0)
        {
                printf("\n%d/%d/%d  1",c[i].date,c[i].month,c[i].year);
        }
        else
                printf("\n%d/%d/%d  0",c[i].date,c[i].month,c[i].year);
}
printf("Enter the dates\n");
for(i=0;i<7;i++)
{
        scanf("%d%d%d%s", &c[i].date, &c[i].month, &c[i].year, &c[i].day);
        c[i].date = c[i].date+5;
        if(strcmp(c[i].day,"Mon")==0)
        {
        strcpy(c[i].day,"Fri");
        }
        else if(strcmp(c[i].day,"Tue")==0)
        {
        strcpy(c[i].day,"Sat");
        }
        else if(strcmp(c[i].day,"Wed")==0)
        {
        strcpy(c[i].day,"Sun");
        }
```

```
    else if(strcmp(c[i].day,"Thur")==0)
    {
    strcpy(c[i].day,"Mon");
    }
    else if(strcmp(c[i].day,"Fri")==0)
    {
    strcpy(c[i].day,"Tue");
    }
    else if(strcmp(c[i].day,"Sat")==0)
    {
    strcpy(c[i].day,"Wed");
    }
    else if(strcmp(c[i].day,"Sun")==0)
    {
    strcpy(c[i].day,"Thur");
    }
}
printf("\n Display Dates");
for(i=0;i<7;i++)
{
    printf("\n%d/ %d/ %d/ %s", c[i].date, c[i].month, c[i].year, c[i].day);
}
printf("\n Display Dates");
for(i=0;i<7;i++)
{
    if(c[i].date%2==0)
    printf("\n%d/ %d/ %d/ %s", c[i].date, c[i].month, c[i].year, c[i].day);
```

```
}
getch();
}
```