LIST OF EXPERIMENTS

1. Simple C++ Programs to Implement Various Control Structures.
    a. If statement
    b. Switch case statement and do while loop
    c. For loop
    d. While loop
2. Programs to Understand Structure & Unions.
    a. Structure
    b. union
3. Programs to Understand Pointer Arithmetic.
4. Functions & Recursion.
    a. Recursion
    b. function
5. Inline Functions.
6. Programs to Understand Different Function Call Mechanism.
    a. Call by reference & Call by Value
7. Programs to Understand Storage Specifiers.
8. Constructors & Destructors.
9. Use of "this" Pointer. Using class
10. Programs to Implement Inheritance and Function Overriding.
    a. Multiple inheritance –Access Specifiers
    b. Hierarchical inheritance – Function Overriding /Virtual Function
11. Programs to Overload Unary & Binary Operators as Member Function & Non Member Function.
    a. Unary operator as member function
    b. Binary operator as non member function
12. Programs to Understand Friend Function & Friend Class.
    a. Friend Function
    b. Friend class
13. Programs on Class Templates

LIST OF QUESTIONS

Questions:

1. **Simple C++ Programs to Implement Various Control Structures.**
   a. **If statement**
   b. **Switch case statement and do while loop**
   c. **For loop**
   d. **While loop**

Ex 1A: if .. else statement

An electricity board charges the following rates to domestic users ti discourage large consumption of energy:

FOR the first 100 units	-	60P per unit
For next 200 units	-	80P per unit
Beyond 300 units	-	90P per unit

All users are charged a minimum of Rs.50.00. if the total amount is more than Rs.300.00 than an additional surcharge of 15% is added

Write a C++ program to read the names of users and number of units consumed and print out the charges with names

Ex 1B: switch.. case statements and do .. while loop

An election is contested by five candidates. The candidates are numbered 1 to 5 and a voting is done by marking the candidate number in a ballot paper. Write a C++ program to read the ballot and count the votes cast for each candidate using an array variable count. In case, a number read is outside the range 1 to 5 the ballot should be considered as a 'spoilt ballot', and the program should also count the number of spoilt ballots

Ex 1C: for loop

Write a C++ program to print the following by reading number of rows to be printed from the user

```
    *
   * *
  * * *
 * * * *
* * * * *
```

Ex 1D: while loop

Write a C++ program to print the Fibonacci series 0 1 1 2 3 5 8 13 …. By getting number of number to be displayed is given as input
Eg. 5 is input value means it should print first 5 numbers 0 1 1 2 3

Questions:

2. **Programs to Understand Structure & Unions.**
    a. **Structure**
    b. **union**

Ex 2A: Structure

Create a Structure called employee with the following details as variables within it.

1. Name of the employee
2. Age
3. Designation
4. Salary

Write a C++ program to create array of objects for the structure to access these and print the name, age, designation and salary

Ex2B: Union

Create a Union called student with the following details as variables within it.

1. Name of the student
2. Age
3. Year of study
4. Semester
5. 5 different subject marks in array

Write a C++ program to create object for the union to access these and print the Name, age, year, semester and grade according to their percentage of marks scored.
90 % and above – S grade
80% to 89% -- A grade
70% to 79% -- B grade
60% to 69% -- C grade

50% to 59% -- D grade
<50% -- F grade

Question

3. **Programs to Understand Pointer Arithmetic.**

Ex 3:
Write a C++ program to find the number of vowels present in the given character array using pointer arithmetic.

Question

4. **Functions & Recursion.**
    a. **Function**
    b. **Recursion**

Ex 4A:

Write a C++ program to print the given number in reverse order. Use functions with return type and without return type for reversing the number
Ex: given number is 2345 , output should be 5432

Ex 4B:

Write a C++ program to find the sum of factorial of a given number using recursive function

Question

5. **Inline Functions.**

Ex 5:

Write a C++ program to perform different arithmetic operation such as addition, subtraction, division, modulus and multiplication using inline function

Question

6. **Programs to Understand Different Function Call Mechanism.**
    a. **Call by reference and Call by value**

Ex 6:

Write a C++ program to swap two number by both call by value and call by reference mechanism, using two functions swap_value() and swap_reference respectively , by getting the choice from the user and executing the user's choice by switch-case.

Question

7. **Programs to Understand Storage Specifiers.**

Ex 7:

Write a C++ program to demonstrate the static and non static variable usage defining them within a function.

Question

8. **Constructors & Destructors.**

Ex 8:

Create a class for counting the number of objects created and destroyed within various block using constructor and destructors.

Question

9. **Use of "this" Pointer. Using class**

Ex 9:

Write a C++ program to create three objects for a class named pntr_obj with data members such as roll_no & name . Create a member function set_data() for setting the data values and print() member function to print which object has invoked it using 'this' pointer

Question

10. **Programs to Implement Inheritance and Function Overriding.**

    a. **Multiple inheritance –Access Specifier**
    b. **Hierarchical inheritance – Function Overriding /Virtual Function**

Ex 10A:

Write a C++ program with different class related through multiple inheritance and demonstrate the use of different access specifiers by means of member variables and member functions.

Ex 10B:

Write a C++ program to explain virtual function (polymorphism) by creating a base class c_polygon which has virtual function area(). Two classes c_rectangle and c_traingle derived from c_polygon and they have area() to calculate and return the area of rectangle and triangle respectively.

Question

11. **Programs to Overload Unary & Binary Operators as Member Function & Non Member Function.**
    a. **Unary operator as member function**
    b. **Binary operator as non member function**

Ex 11 A:

Write a C++ program to count the number of persons inside a bank, by increasing count whenever a person enters a bank, using an increment(++) operator overloading function, and decrease the count whenever a person leaves the bank using a decrement(--) operator overloading function inside a class

Ex 11 B:

Write a C++ program to create two objects of a class called company and add their data members using an operator overloaded function for '+' operator and '-'operator

Question

12. **Programs to Understand Friend Function & Friend Class.**
    a. **Friend function**
    b. **Friend class**

Ex 12 A:

Write a program to accept five different numbers by creating a class called friendfunc1 and friendfunc2 taking 2 and 3 arg respectivly and calculate the average of these numbers by passing object of the class to friend function.

Ex 12 B:

Write a program to accept the student detail such as name and 3 different marks by get_data() method and display the name and average of marks using display() method. Define a friend class for calculating the average of marks using the method marrk_avg().

Question:

13. Programs on Class Templates

Ex 13:

Write a program to explain class template by creating a template T for a class named pair having two data members of type T which are inputted by a constructor and a member function get-max() return the greatest of two numbers to main. Note: the value of T depends upon the data type specified during object creation.

//EX1A: if..else statement

```
#include<iostream.h>
#include<conio.h>
int main()
{
int no_unit;
float charge,scharge;
char name[20];
cout<<"\n enter name and number of units consumed";
cin>>name;
cin>>no_unit;
if(no_unit<=100)
   charge=(0.60*no_unit);
elseif(no_unit>100&&no_unit<=300)
  {
    charge=(100*0.60);
    charge+=((no_unit-100)*0.80);
  }
elseif(no_unit>=300)
  {
   charge=(100*0.60);
   charge+=(200*0.80);
   charge+=((no_unit-300)*0.90);
  }
if(charge<50)
   charge=50;
if(charge>300)
 {
  scharge=(0.15*charge);
    charge+=scharge;
 }
cout<<"electricity bill \n";
cout<<name<<" : : rs"<<charge;
getch();
return(0);
}
```

//Ex1B: switch case, do.. while

```
#include<iostream.h>
int main()
{
    int i,can[5],ballot,count[5];
    char ch;
    for(i=0;i<=5;i++)
    {
        count[i]=0;
    }
    do
    {
    cout<<"\nEnter the ballot number :";
    cin>>ballot;
    switch(ballot){
    case 1: count[1]++;
        break;
    case 2: count[2]++;
        break;
    case 3: count[3]++;
        break;
    case 4: count[4]++;
        break;
    case 5: count[5]++;
        break;
    default: count[0]++;
    }
    cout<<" \nWant to vote again\n";
    cin>> ch;
    }
    while(ch=='y');
    for(i=1;i<=5;i++)
    {
        cout<<"\nNo:of votes for candidate "<< i <<"="<<count[i];
    }
    cout<<"\n Sploit ballots are "<<count[0];
    return 0;
}
```

//Ex 1C: Pattern printing

```
#include<iostream.h>
int main()
{
    int n,i,j,star,space,count;
    cout<<"\nEnter the number of lines: ";
    cin>>n;
    space=n-1;
    count=0;
    star=1;
    do
    {
        for(i=1;i<=space;i++)
        {
                cout<<" ";
        }
        for(j=1;j<=star;j++)
        {
                cout<<"*"<<" ";
        }
        cout<<"\n";
        space--;
        star++;
        count++;
    }while(count<n);
    return 0;
}
```

//Ex 1D:Fibonacci series

```
#include<iostream.h>
int main()
{
    int i,j,n,f0,f,f1;
    cout<<"\nEnter the value of n\n";
    cin>>n;
    f0=-1;
    f1=1;
    i=1;
    while(i<=n)
    {
        f=f0+f1;
        cout<<" " <<f;
        f0=f1;
        f1=f;
        i++;
    }
}
```

//Ex 2A: Structure:

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
struct bio
    {
        char name[20];
        int age;
        float sal;
        char designation[20];
    };
void main()
{
    struct bio info[2];
    for(int i=0;i<2;i++)
    {
    cout<<"Enter the data : name, age, salary, designation"<<endl;
    cin>>info[i].name>>info[i].age>>info[i].sal>>info[i].designation;
    }
    for(i=0;i<2;i++)
    {
    cout<<"Given Data for obj["<<i<<"]:- "<<endl;
    cout<<"++++++++++++++++++++++++++++\n";
    cout<<"Name                 : "<<info[i].name<<endl;
    cout<<"Age                  : "<<info[i].age<<endl;
    cout<<"Salary       : "<<info[i].sal<<endl;
    cout<<"Address    : "<<info[i].designation<<endl;
    }
    cout<<"Size of the structure Student is :"<<sizeof(info[0]);
    getch();
}
```

//EX 2B: Union

```
#include <iostream.h>
union Student
{
    char name[25],grade;
    int age,year,semester;
    float m[5];
}s1;
int main()
{
    int i,j;
    int total=0,average;
    char grade;
    cout<<"\nEnter the student's name : ";
    cin>>s1.name;
    cout<<"\nName : "<<s1.name;
    cout<<"\n----------------------\n";
    cout<<"\nEnter the age";
    cin>>s1.age;
    cout<<"\nAge : "<<s1.age;
    cout<<"\n----------------------\n";
    cout<<"\nEnter the year";
    cin>>s1.year;
    cout<<"\nYear : "<<s1.year;
    cout<<"\n----------------------\n";
    cout<<"Enter semester ";
    cin>>s1.semester;
    cout<<"\nSemester : "<<s1.semester;
    cout<<"\n----------------------\n";
    cout<<"Enter the five different marks for the student :";
    for(i=0;i<5;i++)
    {
        cin>>s1.m[i];
        if(s1.m[i]>=50)
        total=total+s1.m[i];
        else
        j++;
    }
    average=total/5;
    if(average>=90)
     {   grade='S';}
    else if(average>=80 && average<90)
```

```
    {   grade='A';}
   else if(average>=70 && average<80)
    {   grade='B';}
   else if(average>=60 && average<70)
    {   grade='C';}
   else if(average>=50 && average<60)
    {   grade='D';}
    else if(j>=1)
    {grade='F';}
   cout<<"\nGrade : "<<grade<<"\n";
   cout<<"Size of the union is :"<<sizeof(s1);
   return 0;
}
```

//Ex 3: Pointer Arithmetic

```
 #include<iostream.h>
void main()
{
const int nArraySize=20;
char szName[nArraySize];
cout<<"\nEnther the word:";
cin>>szName;
int nVowels = 0;
for (char *pnPtr = szName; pnPtr <= szName + nArraySize; pnPtr++)
{
   switch (*pnPtr)
   {
     case 'A':
     case 'a':
     case 'E':
     case 'e':
     case 'I':
     case 'i':
     case 'O':
     case 'o':
     case 'U':
     case 'u':
        nVowels++;
        break;
   }
}
cout << szName << " has " << nVowels << " vowels" << endl;
}
```

//EX 4AFunction

```
#include<iostream.h>
#include<math.h>
void reverse(int n)
{
    int a;
    cout<<"\nReverse output without return type is =";
    while(n!=0)
    {
        a=n%10;
        cout<<a;
        n=n/10;
    }
}
int reverse1(int n)
{
    int a,c=0,x,z,b[100],i=0;
    while(n!=0)
    {
        b[i]=n%10;
        i++;
        x=i;
        n=n/10;
    }
    int y=x-1;
    for(i=0; i<=x-1;i++)
    {
      z=b[i]*pow(10,y);
      c=c+z;
      y--;
    }
    return c;
}
int main()
{
    int n;
    cout<<"\nEnter the number to be reversed \n";
    cin>>n;
    reverse(n);
    cout<<"\nReverse output with return type is ="<<reverse1(n);
    return 0;
}
```

//EX 4B: Recursion

```
#include<iostream.h>
int factorial(int n)
{
 int fact;
 if(n==1 || n==0)
 {
     return 1;
 }
 fact=n*factorial(n-1); //recursive function
 return fact;
}
int main()
{
     int n,i,sum=0;
     cout<<"\nEnter the number of terms\n";
     cin >>n;
     for(i=1;i<=n;i++)
     {
          sum+=factorial(i);
     }
     cout<<"Sum of factorial of first "<<n<<" number is "<<sum;
     return 0;
}
```

//EX 5: Inline function

```
#include<iostream.h>
inline int add(int a, int b)
{
    return a+b;
}
inline double division(double x, double y)
{
    return x/y;
}
inline float difference(float a, float b)
{
    return a-b;
}
inline int modulus(int x, int y)
{
    return x%y;
}
inline long int multiplication(long int a, long int b)
{
    return a*b;
}
int main()
{
    int a,b;
    double x,y;
    float i,j;
    long int e,f;
    cout<<"\nEnter 2 integer values for calculating there sum:\n";
    cin>>a>>b;
    cout<<"\nSum of "<<a<<"and "<<b<<" = "<<add(a,b)<<"\n";
    cout<<"\nEnter 2 double values for performing division:\n";
    cin>>x>>y;
    cout<<"\nValue of "<<x<<" divided by "<<y<<" = "<<division(x,y)<<"\n";
    cout<<"\nEnter 2 float values for calculating there difference :\n";
    cin>>i>>j;
    cout<<"\nDifference between "<<i<<" and "<<j<<" = "<<difference(i,j)<<"\n";
    cout<<"Enter 2 integer values for calculating there modulus:\n";
    cin>>a>>b;
    cout<<"\n"<<a<<" modulus "<<b<<" = "<<modulus(a,b)<<"\n";
    cout<<"\nEnter 2 long integer values for calculating there product:\n";
    cin>>e>>f;
```

```
    cout<<"\nProduct of "<<e<<" and "<<f<<" = "<<multiplication(e,f)<<"\n";
    return 0;
}
```

//EX 6: Call by reference & Call by value

```
#include<iostream.h>
#include<conio.h>
void main(void)
{
clrscr();
int x,y,n;
void swap1(int,int);
void swap2(int*x,int*y);
cout<<"enter two values to be swapped\n";
cin>>x>>y;
cout<<"\nfor CALL BY VALUE press1";
cout<<"\nfor CALL BY REFERENCE press2";
cin>>n;
switch(n)
{
case 1:
cout<<"\nCALL BY VALUE";
cout<<"\nvalues before swap()"<<x<<"\t"<<y;
swap1(x,y);
cout<<"\nafter swap outside of swap1()";
cout<<"\nx="<<x<<"\ny="<<y;
break;
case 2:
cout<<"\nCALL BY REFERENCE";
cout<<"\nvalue before swap"<<x<<y;
swap2(&x,&y);
cout<<"\nafter swap(outside of swap2)"<<x<<y;
break;
}
getch();
}
void swap1(int x,int y)
{
int temp;
temp=x;
x=y;
y=temp;
cout<<"\nswapped values(inside swap1())"<<x<<"\t"<<y;
}
void swap2(int *x,int *y)
{
```

```
int temp;
temp=*x;
*x=*y;
*y=temp;
cout<<"\nswapped value(inside swap2())"<<*x<<"\t"<<*y;
}
```

//EX 7: Programs to Understand Storage Specifiers.

```
#include<iostream.h>
#include<conio.h>
void main()
{
clrscr ();
int funct1(int x);
int funct2(int x);
int i,value;
cout<<"WITH STATIC\n";
for(i=1;i<=10;i++)
{
value=funct1(i);
cout<<i<<"\t"<<value<<endl;
}
cout<<"WITHOUT STATIC\n";
for(i=1;i<=10;i++)
{
value=funct2(i);
cout<<i<<"\t"<<value<<endl;
}
getch();
}
int funct1(int x)
{
static int sum=100;
sum+=x;
return sum;
}
int funct2(int x)
{
  int sum=100;
sum+=x;
return sum;
}
```

//Ex 8: Constructors & Destructors.

```
#include<iostream.h>
int count=0;
class contdest
{
    public:
    contdest()
    {
            cout<<"\nObject Created ="<<count++;
    }
    ~contdest()
    {
            cout<<"\nObject Destroyed = "<<--count;
    }
};
void main()
{
    cout<<"\nMain Block\n";
    contdest a,b,c;
    {
            cout<<"\nBlock 1\n";
            contdest d;
    }
    cout<<"\nAgain main Block\n";
    contdest e;
    }
```

//EX 9: Using 'this' pointer

```
#include<iostream.h>
class prn_obj // class declaration
{
//private data members
     int rno;
     char *name;
     public:
          void set_data(char *n, int r) //set data member values
          {
                    name=n;
                    rno=r;
          }
          void print()  // prints data member using 'this' pointer
          {
                    cout<<this->name<<" has invoked print() function"<<endl;
                    cout<<"The roll number is "<<this->rno<<endl;
          }
};
int main()
{
     prn_obj ob1,ob2,ob3; // object declaration
     ob1.set_data("Suba",1);
     ob2.set_data("kayal",2);
     ob3.set_data("Jeysree",3);
     // calling print function using objects
     ob1.print();
     ob2.print();
     ob3.print();
     return 0; //denotes successfull termination
}
```

//EX 10A: Multiple inheritance –Access Specifiers

```
#include<iostream.h>
#include<conio.h>
class base1
{
    protected:
        int z;
    private:
        int y;
    public:
        int x;
        void set_value(int xx,int yy, int zz)
        {
                x=xx;
                y=yy;
                z=zz;
        }
        void display1()
        {
                cout<<"\nX="<<x;
                cout<<"\ny="<<y;
                cout<<"\nZ="<<z;
        }
};
class base2
{
    protected:
        int a;
    private:
        int b;
    public:
        int c;
        void set_values(int aa,int bb, int cc)
        {
                a=aa;
                b=bb;
                c=cc;
        }
        void display2()
        {
                cout<<"\nA="<<a;
                cout<<"\nB="<<b;
```

```
                cout<<"\nC="<<c;
        }
};
class derived:public base1,private base2
{
    public:
    void display()
    {
        a=100;
        c=200;
        cout<<"\nValue of X in derived class="<<x;
        //cout<<"\nValue of Y in derived class="<<y; base1::y not accessible
        cout<<"\nValue of Z in derived class="<<z;
        cout<<"\nValue of A in derived class="<<a;
        //cout<<"\nValue of B in derived class="<<b;base2::b not accessible
        cout<<"\nValue of C in derived class="<<c;
    }
};
void main()
{
    clrscr();
    derived d;
    d.set_value(10,20,30);
    //d.set_values(100,200,300);not accessible when derived as private
    d.display1();
    //d.display2();  not accessible when derived as private
    d.display();
    cout<<"\nValue of X in main="<<d.x;
    //cout<<"\nValue of Y in main="<<d.y; base1::y not accessible
    //cout<<"\nValue of Z in main="<<d.z; base1::z not accessible
    //cout<<"\nValue of C in main="<<d.a; base2::a not accessible
    //cout<<"\nValue of B in main="<<d.b; base2::b not accessible
    //cout<<"\nValue of C in main="<<d.c; base2::c not accessible
    getch();
}
```

//EX 10B: Hierarchical inheritance – Function Overriding /Virtual Function

```
#include<iostream.h>
#include<conio.h>
class c_polygon
{
    protected:
        float a,b;
    public:
    void get_data()
    {
        cout<<"\nEnter any two floating values:\n";
        cin>>a>>b;
    }
    virtual float area()
    {
            return 0;
    }
};
class c_rectangle:public c_polygon
{
    public:
    float area()
    {
        return (a*b);
    }
};
class c_triangle:public c_polygon
{
    public:
    float area()
    {
        return (b*a/2);
    }
};
void main()
{
    clrscr();
    c_rectangle r;
    c_triangle t;
    c_polygon *p;
    p=&r;
    p->get_data();
```

```
    cout<<"\nArea of rectangle is "<<p->area();
    p=&t;
    p->get_data();
    cout<<"\nArea of triangle is "<<p->area();
    getch();
}
```

//Ex 11A: Unary operator as member function

```
#include<iostream.h>
#include<conio.h>
class counter
{
        int count;
        public:
                counter()
                {
                        count=0;
                }
                counter operator++(int)
                {
                        count++;
                }
                int get_count()
                {
                        return count;
                }
                counter operator--(int)
                {
                        count--;
                }
};
void main()
{
        clrscr();
        counter c1;
        cout<<"Initial No Of People "<<c1.get_count()<<endl;
        c1++;
        c1++;
        c1++;
        cout<<"Present No Of People "<<c1.get_count()<<endl;
        c1--;
        c1--;
        cout<<"Present No Of People "<<c1.get_count()<<endl;
        getch();
}
```

//EX 11B: Binary operator as non member function

```
#include<iostream.h>
#include<conio.h>
class company
{
        int sal;
        int nop;
        public:
                friend void print(company &);
                company()
                {
                        sal=0;
                        nop=0;
                }
                company operator+(company);
                company company:: operator-(company c)
                {
                        company temp;
                        temp.sal=sal-c.sal;
                        temp.nop=nop-c.nop;
                        return temp;
                }
                void set_data(int s, int n)
                {
                        sal=s;
                        nop=n;
                }
};
void print(company &sum)
{
        cout<<"Total Salary is :"<<sum.sal<<endl;
        cout<<"Total No: of Parts is:"<<sum.nop<<endl;
}
company company:: operator+(company c)
{
        company temp;
        temp.sal=sal+c.sal;
        temp.nop=nop+c.nop;
        return temp;
}
void main()
{
```

```
        clrscr();
        int s,n;
        company sum,ram,clif,sub;
        cout<<"Enter the salary for Ram";
        cin>>s;
        cout<<"Enter No:Of Parts for Ram:";
        cin>>n;
        ram.set_data(s,n);
        cout<<"Enter the salary for clif";
        cin>>s;
        cout<<"Enter No:Of Parts for clif:";
        cin>>n;
        clif.set_data(s,n);
        sum=ram+clif;
        print(sum);
        sub=sum-ram;
        print(sub);
        getch();
}
```

//EX 12A: Friend Function

```
#include <iostream.h>
#include<conio.h>
class friendfunc
{
      float a,b,c,d,e;
      friend void cal_avg(friendfunc &);
      public:
          void get_data()
          {
                    cout<<"\nEnter 5 different values:\n";
                    cin>>a>>b>>c>>d>>e;
          }

};
void cal_avg(friendfunc &f)
{
      float avg=(f.a+f.b+f.c+f.d+f.e)/5;
      cout<<"Average ="<<avg;
}
void main()
{
          clrscr();
          friendfunc ob;
          ob.get_data();
          cal_avg(ob);
          getch();
}
```

//EX 12B: Friend Class

```
#include<iostream.h>
#include<conio.h>
class student
{
      char name[20];
      int mark1,mark2,mark3;
      friend class friendclass;
      public:
      void get_data()
      {
                  cout<<"\nEnter name:";
                  cin>>name;
                  cout<<"\nEnter 3 marks:";
                  cin>>mark1>>mark2>>mark3;
      }
      void display(int a)
      {
                  cout<<"\nOutput\n ";
                  cout<<"============\n";
                  cout<<"\nName ="<<name;
                  cout<<"\nAverage ="<<a;
      }
};
class friendclass
{
          public:
          void mark_avg(student &stud)
          {
                  int avg=(stud.mark1+stud.mark2+stud.mark3)/3;
                  stud.display(avg);
          }
};
void main()
{
      clrscr();
      student s;
      s.get_data();
      friendclass obj;
      obj.mark_avg(s);
      getch();
}
```

//EX13A: Function Template

```
#include<iostream.h>
#include<conio.h>
template<typename T> void prn_arr(T *arr,int count)
{
	for(int i=0;i<count;i++)
	{
		cout<<arr[i]<<" ";
	}
	cout<<endl;
}
void main()
{
	clrscr();
	const int a_cnt=5;
	const int b_cnt=7;
	const int c_cnt=6;
	int a[a_cnt]={1,2,3,4,5};
	double b[b_cnt]={1.1,2.2,3.3,4.4,5.5,6.6,7.7};
	char c[c_cnt]="HELLO";
	cout<<"Integer Array\n";
	prn_arr(a,a_cnt);
	cout<<"\nDouble Array\n";
	prn_arr(b,b_cnt);
	cout<<"\nCharacter Array\n";
	prn_arr(c,c_cnt);
	getch();
}
```

//EX 13B: Class Template

```
#include<iostream.h>
#include<conio.h>
template<class T>
class pair
{
        T a;
        T b;
        public:
        pair()
        {
                cin>>a>>b;
        }
        T get_max();
};
template<class T>
T pair<T>::get_max()
{
        T ret;
        ret=a>b?a:b;
        return ret;
}
void main()
{
        clrscr();
        cout<<"\nEnter 2 Integer Numbers:\n";
        pair<int>obj1;
        cout<<"Greatest Integer is "<<obj1.get_max()<<endl;
        cout<<"\nEnter 2 Float Numbers:\n";
        pair<float>obj2;
        cout<<"Greatest Float is "<<obj2.get_max()<<endl;
        getch();
}
```