Program 1

Create a simple class STUDENT containing data members roll no, name age & amp; display the contents using setdat() and Outdata()methods.

Test the program with

a) Member function inside the body of the student class.

```
#include<iostream.h>
class student
{
int rno;
char Name[20];
int age,M1,M2,M3;
public:
float avg;
void readdata()
{
cout<<"\nEnter the roll no.:";
cin>>rno;
cout<<"\nEnter the name:";
cin>>Name;
cout<<"\nEnter the age:";
cin>>age;
cout<<"Enter the marks of three subjects:";
cin>>M1>>M2>>M3;
}
void average()
{
avg=(M1+M2+M3)/3;
}
void display()
{
cout<<rno<<"\t\t"<<Name<<"\t"<<M1<<"\t"<<M2<<"\t"<<M3<<"\t"<<avg<<"\n";
}
};
```

```
int main()
{
student s[10];
int n,i;
float total =0;
cout<<"\nEnter the number of students:";
cin>>n;
for(i=0;i<n;i++)
{
s[i].readdata();
s[i].average();
total+=s[i].avg;
}
cout<<"Roll no.\tName\tM1\tM2\tM3\tAverage\n";
for(i=0;i<n;i++)
s[i].display();
cout<<"total average:"<<total/n;
return 0;
}
```

1 a) Output:

b) Member function outside the body of the student class (using::operator).

```
#include<iostream.h>
class student
{
int rno;
char name[20];
int age,m1,m2,m3;
public:
float avg;
void readdata();
void average();
void display();
};
void student :: readdata()
{
cout<<"Enter the roll no:";
cin>>rno;
cout<<"\n Enter the name:";
cin>>name;
cout<<"\n Enter the age:";
cin>>age;
cout<<"Enter the marks of 3 subjects";
cin>>m1>>m2>>m3;
}
void student::average()
{
avg=(m1+m2+m3)/3;
 }
void student::display()
{
cout<<rno<<"\t"<<name<<"\t"<<m1<<"\t"<<m2<<"\t"<<m3<<"\t"<<avg<<"\n";
}
 int main()
{
 student s[10];
 int n,i;
float total=0;
 cout<<"Enter the numbers of students:";
 cin>>n;
```

```
 for(i=0; i<n; i++)
{
 s[i].readdata();
s[i].average();
 total+=s[i].avg;
}
 cout<<"Roll no\t Name\t M1\t M2\t M3\t Average\n";
 for(i=0;i<n;i++)
 s[i].display();
 cout<<"Total average:"<<total/n;
 return 0;
 }
```

1 b) Output:

Program 2

Write a C++ program to create class DATE and member function day, month, and year. Display age of the person by considering date of birth and current date using inline function.

```
#include   <iostream>
using namespace std;

void age(int pd, int pm, int py,int bd, int bm, int by)
{
  int d, m, y;
  int md[] = { 31,28,31,30,31,30,31,31,30,31,30,31 };
  y = py - by;
  if (pm < bm)
  {
    y--;
    m = 12 - (bm - pm);
  }
  else
  {m = pm - bm;}
  if (pd < bd)
  {
    m--;
    d = md[pm - 1] - (bd - pd);
  }
  else
  {d = pd - bd;}
  cout << "your age is : \n";
  cout << y << " years " << m << " months " << d << " days. ";
}
int main()
{
  int pd, pm, py, bd, bm, by;
  cout << " Enter the present date in the format dd mm yyyy : ";
  cin >> pd >> pm >> py;
  cout << " Enter the birth date in the format dd mm yyyy : ";
  cin >> bd >> bm >> by;
```

```
  age(pd, pm, py, bd, bm, by);

  return 0;
}
```

2) Output:

Program 3

Write a C++ program to create a class ACC with data members, accno, balance. Create objects ACC1, ACC2 andACC3. Write a member function to transfer money from ACC3 to ACC1.Display the balance in all accounts.

```
#include<iostream.h>
class acc
{
int  no;
int bal;
public:
void get()
{
cout<<"\nEnter the account number: ";
cin>>no;
cout<<"\nEnter the balance: ";
cin>>bal;
}
void set(int ain)
{
no=ain;
bal=0;
}
void set(int ain,float bin)
{
no=ain;
bal=bin;
}
void display()
{
cout<<"\n The Account number: "<<no;
cout<<"\nBalance: "<<bal;
}
void transfer(acc &a,float amt);
};
void acc::transfer(acc &a,float amt)
{
```

```
bal = bal - amt;
a.bal = a.bal + amt;
}
int main()
{
float amount;
acc  a1,a2,a3;
a1.get();
a2.set(2);
a3.set(3,100.5);
cout<<"\nAccount information_____\n";
a1.display();
a2.display();
a3.display();
cout<<"\n How much money is to be transfered from account 1 to account 2: ";
cin>>amount;
a1.transfer(a2,amount);
cout<<"\n Updated account information____\n";
a1.display();
a2.display();
a3.display();
return 0;
}
```

3) Output:

Program 4

Create a class called QUEUE perform insertion and deletion of elements from the queue using constructors and destructors.

```
#include <iostream.h>

#include<conio.h>

#include<stdlib.h>

#define MAX_SIZE 100

using namespace std;

class Queue {

private:

   int item, i;

   int arr_queue[MAX_SIZE];

   int rear;

   int front;


public:

   Queue() {

      rear = 0;

      front = 0;

   }


   void insert() {
```

```
    if (rear == MAX_SIZE)
        cout << "\n## Queue Reached Max!";
    else {
        cout << "\nEnter The Value to be Insert : ";
        cin>>item;
        cout << "\n## Position : " << rear + 1 << " , Insert Value  : " << item;
        arr_queue[rear++] = item;
    }
  }

  void removeData() {
    if (front == rear)
        cout << "\n## Queue is Empty!";
    else {
        cout << "\n## Position : " << front << " , Remove Value   :" << arr_queue[front];
        front++;
    }
  }

  void display() {
    cout << "\n## Queue Size : " << (rear - front);
```

```
        for (i = front; i < rear; i++)
         cout << "\n## Position : " << i << " , Value  : " << arr_queue[i];
    }
};

int main() {
    int choice, exit_p = 1;
    Queue obj;
    cout << "\nSimple Queue Example - Class and Memeber Functions in C++";
    do {
        cout << "\n\n Queue Main Menu";

        cout << "\n1.Insert \n2.Remove \n3.Display \nOthers to exit";
        cout << "\nEnter Your Choice : ";
        cin>>choice;
        switch (choice) {
            case 1:
                obj.insert();
                break;
            case 2:
                obj.removeData();
                break;
```

```
            case 3:
                obj.display();
                break;
            default:
                exit_p = 0;
                break;
        }
    } while (exit_p);


    return 0;
}
```

4) Output:

Program 5

Write a C++ program to sort N numbers using swap as friend function.

```
#include <iostream>
using namespace std;

class Swap {

   // Declare the variables of Swap Class
   int temp, a, b;

public:

   // Define the parameterized constructor, for inputs
   Swap(int a, int b)
   {
      this->a = a;
      this->b = b;
   }

   // Declare the friend function to swap, take arguments
   // as call by reference
   friend void swap(Swap&);
};

// Define the swap function outside class scope
void swap(Swap& s1)
{
   // Call by reference is used to passed object copy to
   // the function
   cout << "\nBefore Swapping: " << s1.a << " " << s1.b;

   // Swap operations with Swap Class variables
   s1.temp = s1.a;
   s1.a = s1.b;
   s1.b = s1.temp;
```

```
    cout << "\nAfter Swapping: " << s1.a << " " << s1.b;
}

// Driver Code
int main()
{
    // Declare and Initialize the Swap object
    Swap s(4, 6);
    swap(s);
    return 0;
}
```

5) Output:

Program 6

Write a C++ program to create a class NAME and implement the following operations. Display the result after every operation by overloading the<<.
i) NAME firstname =“Herbert”
ii) NAME lastname =“Schield”
iii) NAME fullname = firstname+lastname

```
#include<iostream.h>
#include<string.h>
class string
{
char str[50];
public:
string()
{
strcpy(str," ");
}
string(char *s)
{
strcpy(str,s);
}
string(string &s)
{
strcpy(str,s.str);
}
string operator+(string s2)
{
strcat(str," ");
strcat(str,s2.str);
return *this;
}
friend ostream& operator<<(ostream &print,string);
};
ostream& operator<<(ostream &print,string s)
{
print<<s.str;
```

```
return print;
}
int main()
{
string s1="HERBERT";
string s2(s1);
cout<<"First string is: "<<s2<<"\n";
string s3="SCHIELD";
string s4(s3);
cout<<"The second string is: "<<s4<<"\n";
string s5=s1+s3;
cout<<"Sum of two strings is: "<<s5<<"\n";
return 0;
}
```

6) Output:

Program 7

Write a C++ program to create a class called MATRIX using a two-dimensional array of integers. Implement the following operations by overloading the operator = = which checks the compatibility of two matrices m1and m2 to be added and subtracted. Perform the addition and subtraction by overloading the operators + and – respectively. Display the results (sum matrix m3 and difference matrix m4) by overloading the operator <<.

if (m1 = = m2)
{
m3 = m1 + m2;
m4 = m1 – m2;
}
Else
Display error.

```
#include<iostream.h>
class matrix
{
int a[10][10],m,n;
public:
void getdata();
int operator==(matrix);
matrix operator+(matrix);
matrix operator-(matrix);
friend ostream& operator<<(ostream&,matrix&)
};
void matrix::getdata()
{
cout<<"Enter the size of the matrix: ";
cin>>m>>n;
cout<<"Enter the element: ";
for(int i=0;i<m;i++)
for(int j=0;j<n;j++)
cin>>a[i][j];
}
int matrix::operator==(matrix cm)
```

```
{
if(m==cm.m && n==cm.n)
return 1;
else
return 0;
}
matrix matrix::operator+(matrix am)
{
matrix temp;
for(int i=0;i<m;i++)
{
for(int j=0;j<n;j++)
{
temp.a[i][j]=a[i][j]+am.a[i][j];
}
temp.m=m;
temp.n=n;
}
return temp;
}
matrix matrix::operator-(matrix sm)
{
matrix temp;
for(int i=0;i<m;i++)
{
for(int j=0;j<n;j++)
{
temp.a[i][j]=a[i][j]-sm.a[i][j];
}
temp.m=m;
temp.n=n;
}
return temp;
}
ostream& operator<<(ostream &doubt,matrix &d)
{
for(int i=0;i<d.m;i++)
```

```
{
for(int j=0;j<d.n;j++)
{
doubt<<d.a[i][j];
cout<<" ";
}
cout<<"\n";
}
return doubt;
}
int main()
{
matrix m1,m2,m3,m4;
m1.getdata();
m2.getdata();
if(m1==m2)
{
m3=m1+m2 ;
m4=m1-m2;
cout<<"The addition of thwo matrices: \n"<<m3;
cout<<"The substraction of two matrices: \n"<<m4;
}
else
cout<<"The two matrices are not compatible\n";
return 0;
}
```

7) Output:

Program 8

Write a C++ program to create a class called COMPLEX and implement the following overloading functions ADD that return a COMPLEX number.

i. ADD (a, s2) – where s1 is an integer (real part) and s2 is a complex number.
ii. ADD (s1, s2) – where s1 and s2 are complex numbers.

```
#include <iostream>
#include <iostream.h>
#include <conio.h>
#include <math.h>
class complex
{
int real;
int img;
public:
 void input()
 {
 cout << "enter real and img part" << '\n';
cin >> real >> img;
}
 void output()
 {
 if(img<0)
 cout << real<< img << "i" << '\n';
else
 cout << real << "+"<< "i" << img << '\n';
}
 friend complex add(int,complex)
 friend complex add(complex,complex)
 };
 complex add(int a , complex s2)
 {
complex temp;
temp.real = s2.real + a;
temp.img = s2.img;
return temp;
```

```
 }
 complex add(complex s1, complex s2)
{
footer
 complex s3;
s3.real = s1.real + s2.real;
s3.img = s1.img + s2.img;
return s3;
 }
void main()
{
 complex s1,s2,sum1,sum2;
int a;
clrscr();
s1.input();
s2.input();
cout << "first complex number is:"<< '\n';
s1.output();
cout << "second complex no is:"<< '\n';
 s2.output();
cout << "enter the value of a :";
 cin >> a;
sum1 = add(a,s2);
sum2 = add(s1,s2);
cout << "output of integer and complex no is:"<< '\n';
sum1.output();
cout << "output of complex and complex no is:"<< '\n';
sum2.output();
getch();
}
```

8) Output:

Program 9

Write a C++ program to exchange two numbers using function overloading.

```
#include<iostream.h>
void swap(int &ix, int &iy);
void swap(float &fx,float &fy);
void swap(char &cx,char &cy);
int main()
{
int ix,iy;
float fx,fy;
char cx,cy;
cout<<"Enter 2 integers";
cin>>ix>>iy;
cout<<"Enter 2 floating numbers:";
cin>>fx>>fy;
cout<<"Enter 2  characters:";
cin>>cx>>cy;
cout<<"\n Integers";
cout<<"\n ix= "<<ix<<"\n iy= "<<iy;
swap(ix,iy);
cout<<"\n After swapping:";
cout<<"\nix="<<ix<<"\n iy= "<<iy;
cout<<"\n Floating point number:";
cout<<"\nfx ="<<fx<<"\nfy= "<<fy;
swap(fx,fy);
cout<<"\n After swapping ";
cout<<"\nfx ="<<fx<<"\nfy= "<<fy;
cout<<"\n Characters:";
cout<<"\n ncx= "<<cx<<"\ncy ="<<cy;
swap(cx,cy);
cout<<"\nAfter swapping";
cout<<"\n ncx= "<<cx<<"\ncy= "<<cy;
return 0;
}
void swap(int &a,int &b)
{
int temp;
temp=a;
```

```
a=b;
b=temp;
}
void swap(float &a,float &b)
{
float temp;
temp=a;
a=b;
b=temp;
}
void swap(char &a,char &b )
{
char temp;
temp=a;
a=b;
b=temp;
}
```

9) Output:

Program 10

Design three classes called STUDENT, EXAM and RESULT. The student class has data members Such as those represent Rollno, Name and Branch etc. Create the class EXAM by inheriting the STUDENT class. The EXAM class adds data members representing the marks scored in six subjects. Derive the RESULT class from the EXAM class and it has its own data members. Such as total_marks. Write an interactive program to model this inheritance relationship.

```
#include<iostream.h>

class student

{

int rollno;

char name[30],branch[30];

public:

void getdata()

{

cout<<"Enter the roll no: ";

cin>>rollno;

cout<<"Enter the name of the student: ";

cin>>name;

cout<<"Enter the branch: ";

cin>>branch;

}
```

```
void displaydata()
{
cout<<"\nRoll number: "<<rollno;
cout<<"\nName: "<<name;
cout<<"\nBranch: "<<branch;
}
};
class exam:public student
{
protected:
int mark[6];
public:
void getdata();
void displaydata();
};
void exam::getdata()
{
student::getdata();
cout<<"Enter the marks\n";
for(int i=0;i<6;i++)
cin>>mark[i];
}
```

```
void exam::displaydata()
{
student::displaydata();
cout<<"\nThe marks are:\n";
for(int i=0;i<6;i++)
cout<<mark[i]<<" "<<"\n";
}
class result:public exam
{
int total;
public:
result()
{
total=0;
}
void calculate();
void displaydata()
{
exam::displaydata();
cout<<"\nTotal: "<<total;
}
};
```

```
void result::calculate()
{
exam::getdata();
{
for(int i=0;i<6;i++)
total=total+mark[i];
}
}
int main()
{
result r;
cout<<"Enter the data of students: ";
r.calculate();
cout<<"Student Details\n";
r.displaydata();
return 0;
}
```

10) Output:

Program 11

Create classes RESERVATION, ADULT,SENIOR_CITIZEN,CHILD .The Reservation class containing data members,Name of passenger, age, date of journey, Source, Destination, Ticket charge. Write an interactive program to display the ticket charges depending upon the category of passenger. The classes ADULT, SENIOR_CITIZEN, CHILD are the derived class of RESERVATION. (Note: Category CHILDREN = ½ of adult ticket charge).

```
#include<iostream.h>

#include<process.h>

class reserve

{

char name[20],s[20],d[20];

int age,date;

public:

float chrg;

void getdata()

{

cout<<"Enter the name of the pasenger: ";

cin>>name;

cout<<"\nEnter the source: ";

cin>>s;

cout<<"\nEnter the destination: ";

cin>>d;
```

```
cout<<"\nEnter the age of the person: ";
cin>>age;
cout<<"\nEnter the date of travel(date): ";
cin>>date;
cout<<"\nEnter the ticket charge: ";
cin>>chrg;
}
void display()
{
cout<<"\nName:  "<<name<<"\nSource:  "<<s<<"\nDestination:  "<<d<<"\nAge: "<<age<<"\nDate of journey: "<<date;
}
};
class adult:public reserve
{
public:
adult()
{
getdata();
display();
cout<<"\nTicket charge: "<<chrg;
}
```

```
};
class child:public reserve
{
public:
child()
{
getdata();
display();
cout<<"\nTicket charge: "<<0.5*chrg;
}
};
class snr:public reserve
{
public:
snr()
{
getdata();
display();
cout<<"\nTicket charge: "<<0.25*chrg;
}
};
int main()
```

```
{
int ch;
cout<<"\nPassenger category\n1.Adult\n2.Child\n3.Senior citizen\n4.Exit\n";
while(ch!=4)
{
cout<<"\nEnter the choice: ";
cin>>ch;
switch(ch)
{
case 1:
adult a;
break;
case 2:
child c;
break;
case 3:
snr s;
break;
case 4:
exit(0);
break;
default:
```

```
cout<<"\nEnter a valid choice\n";
}
}
return 0;
}
```

11) Output:

Program 12

Write a C++ program to demonstrate how a pure virtual function is defined, declared and invoked from the object of a derived class through the pointer of base class.

```
#include<iostream.h>
using namespace std;
class B {
  public:
    virtual void s() = 0; // Pure Virtual Function
};

class D:public B {
  public:
    void s() {
      cout << "Virtual Function in Derived class\n";
    }
};

int main() {
  B *b;
  D dobj;
  b = &dobj;
  b->s();
}
```

12) Output:

Program 13

Write a C++ program to perform QUICKSORT for N numbers using template function. Demonstrate sorting of integers and doubles.

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
template<class t>
void quick(t a[],int low,int high)
 {
   t key;
   int i,j,flag=1;
   if(low<high)
   {
     key=a[low];
     i=low+1;
     j=high;
     while(flag)
     {
       while((a[i]<=key) && (i<j))
     i++;
       while(a[j]>key)
     j--;
       if(i<j)
     swap(a,i,j);
     else
     flag=0;
     }
     swap(a,low,j);
     quick(a,low,j-1);
     quick(a,j+1,high);
   }
 }

template<class t1>
```

```
void swap(t1 a[],int x,int y)
 {
  t1 temp;
  temp=a[x];
  a[x]=a[y];
  a[y]=temp;
 }
void main()
 {
  int i,n,a[20];
  clrscr();
  cout<<"Enter the number of elements to be sort::";
  cin>>n;
  cout<<"Enter elements:\n";
  for(i=0;i<n;i++)
    cin>>a[i];
  quick(a,0,n-1);
  cout<<"The sorted elements are:\n";
  for(i=0;i<n;i++)
    cout<< a[i]<<endl;;
  getch();
 }
```

13) Output: