BOARD PRACTICAL PROGRAMS 1-10

- **Please Follow Practical File Instructions Before You Start Writing Practical Programs**
- **Please Write One Sample output on your Own.**
- **Please Write any three (03) SQL Table based Questions & its Solution**

PROGRAM-1

Declare a structure to represent a complex number (a number having real part and an imaginary part).
Write functions to add and subtract two complex numbers. Write main () to test the functions**.**

SOLUTION-1

```
#include<iostream.h>
#include<conio.h>
struct complex
{
int real;
int img;
};
complex add(complex x, complex y)
{
complex c;
c.real=x.real+y.real;
c.img=x.img + y.img;
return c;
}
complex sub(complex x, complex y)
{
complex c;
c.real=x.real-y.real;
c.img=x.img - y.img;
return c;
}
void show(complex c)
{
cout<<c.real<<" +i"<<c.img<<endl;
}
```

```
void main()
{
clrscr();
complex a={50,60};
complex b={20,30};
complex c,d;
c=add(a,b);
d=sub(a,b);
show(a);
show(b);
cout<<"\n sum is \n";
show(c);
cout<<"\n difference is \n";
show(d);
getch();
```

PROGRAM-2

Define a structure Duration with members hours and minutes. Enter duration (no. of hours, no. of minutes, no. of seconds) of two events. Write a function that takes the duration of two events as arguments and returns the total duration of the events in the Duration type variable. Write main () to test the functions.

SOLUTION-2

```
#include<iostream.h>
#include<conio.h>
struct Duration
{
int hh;
int mm;
int ss;
};
void input(Duration &d)
{
cout<<"enter no. of hours:";
cin>>d.hh;
cout<<"enter no. of minutes:";
cin>>d.mm;
cout<<"enter no. of seconds:";
cin>>d.ss;
}
```

```
void display(Duration d)
{
cout<<"\n TIME : ";
cout<<d.hh<<":"<<d.mm<<":"<<d.ss;
}
Duration addTime(Duration d1, Duration d2)
{
Duration d;
int h,m,s;
s=d1.ss+d2.ss;
m=d1.mm+d2.mm+s/60;
h=d1.hh+d2.hh+m/60;
d.ss=s%60;
d.mm=m%60;
d.hh=h;
return d;
}
void main()
{
clrscr();
Duration d1,d2,d;
input(d1);
input(d2);
d=addTime(d1,d2);
display(d1);
display(d2);
display(d);
getch();
}
```

PROGRAM-3

Define a class **Report** with the following specifications:

Private members

adno 4 digit admission number
name 20 characters
marks an array of 5 integer values
average average marks obtained in five subjects
getavg() function to compute and return average of marks

public members

readinfo() function to accept values for adno, name, marks, and invoke the function getavg()
displayinfo function to display members
Complete the member function definitions and write main() to test the class.

SOLUTION-3

```
#include<iostream.h>
#include<stdio.h>
#include<conio.h>
class Report
{
private:
char adno[4];
char name[20];
int marks[5];
float average;
float getavg();
public:
void readinfo();
void displayinfo();
};
float Report::getavg()
{
float sum=0.0;
for(int i=0;i<5;i++)
sum+=marks[i];
return (sum/5.0);
}
void Report::readinfo()
{
cout<<"enter admission no:[4 digits]:";
cin>> adno;
cout<<"enter name:";
gets(name);
cout<<"enter marks of 5 subjects :";
for(int i=0;i<5;i++)
cin>>marks[i];
average=getavg();
}
void Report::displayinfo()
{
```

```
cout<<"\n admission no :"<<adno;
cout<<"\n name :"<<name;
cout<<"\n marks of 5 subjects :";
for(int i=0;i<5;i++)
cout<<marks[i]<<" ";
cout<<"\n average marks :"<<average;
}
void main()
{
clrscr();
Report r;
r.readinfo();
r.displayinfo();
getch();
}
```

PROGRAM-4

Declare a class to represent bank account of 10 customers with the following data members. Name of the

depositor, Account number, Type of account (S for savings and C for current), Balance amount. The class
also contains member functions to do the following:
(i) To initialize data members
(ii) To deposit money
(iii) To withdraw money after checking the balance(minimum balance is Rs 1000)
(iv) To display data members

You are required to give detailed function definitions. Write main() to test the class.

SOLUTION-4

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
class Account
{
private:
char name[30];
```

```
int accno;
char type;
float bal;
public:
Account();
void deposit(float amt);
void withdraw(float amt);
void display();
};
Account::Account()
{
cout<<"enter name :";
gets(name);
cout<<"enter account no:";
cin>>accno;
cout<<"type of account[S|C]:";
cin>>type;
cout<<"enter opening balance:";
cin>>bal;
}
void Account::deposit(float amt)
{
bal+=amt;
}
void Account::withdraw(float amt)
{
float temp;
temp=bal-amt;
if(temp<1000)
{
cout<<"\n transaction cannot be processed";
cout<<"\n minimum balance should be Rs 1000";
return;
}
bal-=amt;
}
void Account::display()
{
cout<<"\n account details";
cout<<"\n name :"<<name;
```

```
cout<<"\naccount no:"<<accno;
cout<<"\n type of account:"<<type;
cout<<"\ncurrent balance:"<<bal;
}
void main()
{
clrscr();
Account ob;
ob.display();
ob.deposit(50000);
ob.display();
ob.withdraw(10000);
ob.display();
getch();
}
```

PROGRAM-5

Define a class **Movie** in C++ with the description given below:

Private Members:
Name_of_movie of type character array(string)
Name_of_director of type character array(string)
Star of type int
Total_print_release of type int
Public Members:
A constructor to assign initial values as follows:
Name_of_movie = NULL,Name_of_director=NULL,Star= 2 and Total_print_release=100
A function calculate_star() which caculates and assigns the value of data member Star as follows:
Total Print Release Star
>= 1000 5
< 1000 & >=500 4
< 500 & >=300 3
< 300 & >=100 2
< 100 1
A function EnterMovie() to input the values of the data members Name_of_movie, Name_of_director and Total_print_release and invoke function calculate_star() to set value for Star
A function ShowMovie() which displays the contents of all the data

members for a movie.

SOLUTION-5

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
#include<stdio.h>
typedef char string[30];
class Movie
{
private:
string name_of_movie;
string name_of_director;
int star;
int total_print_release;
public:
Movie();
void calculate_star();
void entermovie();
void showmovie();
};
Movie::Movie()
{
strcpy(name_of_movie,"NULL");
strcpy(name_of_director,"NULL");
star=2;
total_print_release=100;
}
void Movie::calculate_star()
{
int tpr=total_print_release;
if(tpr>=1000)
star=5;
else if(tpr>=500)
star=4;
else if(tpr>=300)
star=3;
else if(tpr>=100)
star=2;
else
```

```
star=1;
}
void Movie::entermovie()
{
cout<<"enter movie name :";
gets(name_of_movie);
cout<<"enter director name :";
gets(name_of_director);
cout<<"enter no. of print realeases:";
cin>>total_print_release;
calculate_star();
}
void Movie::showmovie()
{
cout<<"\n movie name :"<<name_of_movie;
cout<<"\n director name :"<<name_of_director;
cout<<"\n no. of print realeases:"<<total_print_release;
cout<<"\n Star :"<<star;
}
void main()
{
clrscr();
Movie ob;
ob.showmovie();
cout<<endl;
ob.entermovie();
ob.showmovie();
getch();
}
```

PROGRAM-6

Write a C++ program to perform various operations on a string class with out using language supported built-in string functions.
The operations on a class are:

(a) Read a string (b) Display the string (c) Reverse the string
(d) Copy the string into an empty string (e) Concatenate two strings

SOLUTION-6

```
#include<iostream>
#include<stdio.h>
#include<string.h>
class cString
{
char *str;
public:
cString();
cString(char *s);

cString(cString &ob);
~cString();
void copy(cString &ob);
void concat(cString ob);
void show();
void reverse();
};

cString::cString()
{
str=NULL;
}
cString::cString(char *s)
{
str=new char[strlen(s)+1];
int i=0;
while(*s!='\0')
str[i++]=*s++;
str[i]='\0';
}
cString::~cString()
{
if(str!=NULL)
delete [] str;
}
void cString::copy(cString &ob)
{
ob.str=new char[strlen(str)+1];
for(int i=0;str[i]!='\0';i++)
ob.str[i]=str[i];
```

```
ob.str[i]='\0';
}
void cString::concat(cString ob)
{
int a=strlen(str);
int b=strlen(ob.str);
char *temp=new char[a+b+1];
for(int i=0;i<a;i++)
temp[i]=str[i];
for(int j=0;j<b;j++,i++)
temp[i]=ob.str[j];
temp[a+b]='\0';
cout<<endl<<temp;
delete []str;
str=temp;
}
cString::cString(cString &ob)
{
int l=strlen(ob.str);
str=new char[l+1];
for(int i=0;i<l;i++)
str[i]=ob.str[i];
str[i]='\0';
}
void cString::show()
{
cout<<"\n"<<str;
}
void cString::reverse()
{
char t;
int l=strlen(str);
for(int i=0,j=l-1;i<l/2;i++,j--)
{
t=str[i];
str[i]=str[j];
str[j]=t;
}
}
void main()
```

```
{
cString s1("My Country"), s2("India");
s1.show();
s2.show();
cString s3;
s1.copy(s3);
s3.show();
s3.reverse();
s3.show();
s1.concat(s2);
s1.show();
}
```

PROGRAM-7

A point on a 2D plane can be represented by its X coordinate and Y coordinate.
The sum of two points can be defined as a new point whose X coordinate is the sum of the X coordinate of the two points and whose Y coordinate is the sum of their y coordinates.
Declare and define the class Point with following specifications:

Private members
x, y float (x and y coordinates)
public members
Default constructor set point to origin (0,0)
Parameterised constructor takes coordinates as argument
Copy constructor creates copy of existing object
Sum(Point, Point) returns sum of the points as a new point object
Show(Point) displays x and y coordinates of passed point
Write function definitions and write main() to test the class.

SOLUTION-7

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
#include<stdio.h>
class Point
{
float x, y;
public:
Point(); //default constructor
```

```
Point(int x, int y); //parameterised constructor
Point(Point &p);//copy constructor
static Point sum(Point p1,Point p2); //static methods
static void show(Point p); //static methods
};
void Point::show(Point p)
{
cout<<"\n x="<<p.x<<"\t"<<p.y;
}
Point::Point()
{
x=y=0;
}
Point::Point(int x, int y)
{
this->x=x;
this->y=y;
}
Point::Point(Point &p)
{
x=p.x;
y=p.y;
}
Point Point::sum(Point p1, Point p2)
{
Point p;
p.x=p1.x+p2.x;
p.y=p2.y+p2.y;
return p;
}
void main()
{
clrscr();
Point p1;//default constructor
Point p2(10,20);//parameterised constructor
Point p3(p2);//copy constructor
Point::show(p1); //static show method
Point::show(p2);
Point::show(p3);
Point p4;
```

```
p4=Point::sum(p2,p3); //static sum method
Point::show(p4);
getch();
}
```

PROGRAM-8

Write a program in C++ to give an example of multilevel inheritance. Write main() to test the program.

SOLUTION-8

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
#include<string.h>
// Multilevel Inheritance
typedef char string[30];
class Person
{
private:
int age;
string name;
public:
Person(int a,string n);
void showp();
};
Person::Person(int a,string n)
{
age=a;
strcpy(name,n);
}
void Person::showp()
{
cout<<"\n age :"<<age;
cout<<"\n name :"<<name;
}
class Employee : public Person
{
private:
int ecode;
float salary;
public:
```

```
Employee(int ecode, float salary, int age,string name);
void showe();
};
Employee::Employee(int ecode, float salary, int age,string name) :
Person(age, name)
{
this->ecode=ecode;
this->salary=salary;
}
void Employee::showe()
{
showp();
cout<<"\n code :"<<ecode;
cout<<"\n salary :"<<salary;
}
class Manager :public Employee
{
private:
string dept;
public:
Manager(int age,string name,int code, float salary, string dep);
void showm();
};
Manager::Manager(int age,string name,int code, float salary, string dep):
Employee(code, salary,age,name)
{
strcpy(dept,dep);
}
void Manager::showm()
{
showe();
cout<<"\n department:"<<dept;
}
```

PROGRAM-9

Write a program in C++ to give an example of multiple inheritance. Write

main() to test the program.

SOLUTION-9

```
typedef char string[30];
class Person
{
private:
int age;
string name;
public:
Person(int a,string n);
void showp();
};
Person::Person(int a,string n)
{
age=a;
strcpy(name,n);
}
void Person::showp()
{
cout<<"\n age :"<<age;
cout<<"\n name :"<<name;
}
class Employee
{
private:
int ecode;
float salary;
public:
Employee(int ecode, float salary);
void showe();
};
Employee::Employee(int ecode, float salary)
{
this->ecode=ecode;
this->salary=salary;
}
void Employee::showe()
{
cout<<"\n code :"<<ecode;
cout<<"\n salary :"<<salary;
```

```
}
class Manager :public Person, public Employee
{
private:
string dept;
public:
Manager(int age,string name,int code, float salary, string dep);
void showm();
};
Manager::Manager(int age,string name,int code, float salary, string dep):
Person(age,name),Employee(code, salary)
{
strcpy(dept,dep);
}
void Manager::showm()
{
showp();
showe();
cout<<"\n department:"<<dept;
}
void main()
{
clrscr();
Manager ob(35,"Andrew Murray", 101, 55000.0,"Marketing");
cout<<"\n\n calling Person method:\n";
ob.showp();
cout<<"\n\n calling Employee method:\n";
ob.showe();
cout<<"\n\n calling Manager method:\n";
ob.showm();
getch();
}
```

PROGRAM-10

Write a program to keep count of the number of objects created at run-time using static data members and static member functions.

SOLUTION-10

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

```
class Object
{
int who;
static int count;
public:
Object(int w)
{
who=w;
count++;
cout<<"\n object created :"<<who;
}
~Object()
{
cout<<"\n object destroyed :"<<who;
count--;
}
static void objectcount()
{
cout<<"\n No. of Objects Created :"<<count;
}
};
int Object::count;
void main()
{
clrscr();
{
Object a(101), b(102), c(103);
Object::objectcount();
}
Object::objectcount();
getch();
}
```