**Problems**

**Q1 - Write a program to create student class with data members rollno, marks1, mark2, mark3. Accept data (acceptInfo()) and display using display member function.Also display total, percentage and grade.**

#include <iostream>

using namespace std;

class student

{

private:

int rollNo;

int total;

int mark1;

int mark2;

int mark3;

float perc;

public:

void acceptinfo()

{

cout << "Enter roll number: ";

cin >> rollNo;

cout << "Enter marks of subject 1 : ";

cin >> mark1;

cout << "Enter marks of subject 2 : ";

cin >> mark2;

cout << "Enter marks of subject 3 : ";

cin >> mark3;

total=(mark1+mark2+mark3);

perc=(float)total/300\*100;

}

void displaydata()

{

cout << "Student details:";

cout<<endl;

cout<< "Roll Number of student :" << rollNo;

cout<<endl;

cout<< "Total marks of student :" << total;

cout<<endl;

cout<< "Percentage of student :" << perc;

}

};

int main()

{

student obj;

obj.acceptinfo();

obj.displaydata();

return 0;

}

**Output**

Enter roll number: 1

Enter marks of subject 1 : 78

Enter marks of subject 2 : 98

Enter marks of subject 3 : 85

Student details:

Roll Number of student :1

Total marks of student :261

Percentage of student :87

**Q2 - Create a class Person with data members as name, age, city. Write getters and setters for all the data members. Also add the display function. Create Default and Parameterized constructors.Create the object of this class in main method and invoke all the methods in that class.**

#include <iostream>

using namespace std;

class person

{

private:

char name[100];

int age;

char city[100];

public:

void acceptinfo()

{

cout << "Enter name of person: ";

cin >> name;

cout << "Enter age of person : ";

cin >> age;

cout << "Enter city of person : ";

cin >> city;

}

void displaydata()

{

cout << "Person details:";

cout<<endl;

cout<< "Name of person is :" << name;

cout<<endl;

cout<< "Age of person is :" << age;

cout<<endl;

cout<< "City of person is :" <<city;

}

};

int main()

{

person obj;

obj.acceptinfo();

obj.displaydata();

return 0; }

**Output**

Enter age of person : 26

Enter city of person : pune

Person details:

Name of person is :mahi

Age of person is :26

City of person is :pune

**Q3 - Create a class Date with data members as dd, mm, yy. Write getters and setters for all the data members. Also add the display function. Create Default and Parameterized constructors. Create the object of this class in main method and invoke all the methods in that class.**

#include<iostream>

using namespace std;

class date

{

public:

int dd;

int mm;

int yy;

int getdd()

{

return dd;

}

void setdd(int obj)

{

dd=obj;

}

int getmm()

{

return mm;

}

int setmm(int obj)

{

mm=obj;

}

int getyy()

{

return yy;

}

int setyy(int obj)

{

yy=obj;

}

date(){}

date(int d , int m , int y)

{

dd=d;

mm=m;

yy=y;

}

};

int main()

{

date x(23,05,96);

cout<<x.dd<<" "<<x.mm<<" "<<x.yy <<endl;

return 0;

}

**Output**

23 05 96

**Q4 - Create a class Book with data members as bname, id, author, price. Write getters and setters for all the data members. Also add the display function. Create Default and Parameterized constructors. Create the object of this class in main method and invoke all the methods in that class.**

#include<iostream>

using namespace std;

class book

{

public:

char bname[100];

int id;

char author[100];

int price;

char getbname()

{

return bname;

}

void setbname(char x)

{

bname=x;

}

int getid()

{

return id;

}

void setid(int y)

{

id=y;

}

char getauthor()

{

return author;

}

void setauthor(char a)

{

author=a;

}

char getprice()

{

return price;

}

void setprice(int p)

{

price=p;

}

book(){}

book(char x , int y, char a, int p)

{

bname=x;

id=y;

author=a;

price=p

}

};

int main()

{

book b("C Book" ,12345, "Kanetkar",300);

cout<<b.bname<<" "<<b.id<<" "<<b.author<<" "<<b.price;

return 0;

}

**Output**

C Book 12345 Kanetkar 300

**Q5 - Create a class Point with data members as x,y. Create Default and Parameterized constructors. Write getters and setters for all the data members. Also add the display function. Create the object of this class in main method and invoke all the methods in that class.**

#include <iostream>

using namespace std;

class Point

{

private:

int x, y;

public:

void setter(int d, int m)

{

x = d;

y = m;

}

int getX()

{

return x;

}

int getY()

{

return y;

}

Point()

{

cin >> x >> y;

}

Point(int d, int m)

{

x = d;

y = m;

}

void display()

{

cout << getX() << getY();

}

};

int main()

{

Point d;

d.display();

} **Output**

12 15

12 15

**Q6 - Create a class ComplexNumber with data members real, imaginary. Create Default and Parameterized constructors. Write getters and setters for all the data members. Also add the display function. Create the object of this class in main method and invoke all the methods in that class.**

#include<iostream>

using namespace std;

class complexnumber

{

int real;

float imaginary;

void setter(int x , float y)

{

real = x;

imaginary = y;

}

int getX()

{

return x;

}

float getY()

{

return y;

}

complexnumber()

{

}

complexnumber(int a , float b)

{

real=a;

imaginary=b;

}

int accept()

{

cin>>x>>y;

}

void display()

{

cout << getX() << getY();

}

};

int main()

{

complexnumber n;

n.accept();

n.display();

return 0;

}

**Output**

Input 1 2.5

Output 1 2.5

**Q7 - Create Date class with members day, month, year. Write no argument and parameterized constructor. Create two object s and initialize them using no argument and parameterized constructor respectively. Print date using display function.**

#include<iostream>

using namespace std;

class Date {

private:

int dd,mm,yy;

public:

void setter(int d, int m, int y){

dd = d;

mm = m;

yy = y;

}

int getDay(){

return dd;

}

int getMonth(){

return mm;

}

int getYear(){

return yy;

}

Date(){

cin>>dd>>mm>>yy;

}

Date(int d, int m ,int y){

dd =d;

mm= m;

yy=y;

}

void display(){

cout<<getDay()<<getMonth()<<getYear();

}

};

int main(){

Date d1;

Date d2(3,4,2015);

d1.display();

d2.display();

}

**Output**

23 05 1996

23 05 1996

**Q8 - Create Employee class with members id(int),name(string),dob(Date).Use above created Date class. Write default and parameterized constructor in Employee Class.Write accept () function to accept information and display () to display emp information.**

#include <iostream>

using namespace std;

class Employee

{

public:

int id;

char name[50];

int Day;

int Month;

int Year;

Employee(int i, char n[], int d, int m, int y) : dob(d, m, y)

{

id = i;

name=n;

Day= d;

Month=m;

Year=y;

}

int getDay()

{

return Day;

}

int getMonth()

{

return Month;

}

int getYear()

{

return year;

}

void accept()

{

cin >> name >> id;

int d, m, y;

cin >> d >> m >> y;

dob.setter(d, m, y);

}

void display()

{

cout << name << id << dob.getDay() << dob.getMonth() << dob.getYear();

}

};

int main()

{

Employee E;

E.accept();

E.display();

return 0;

}

**Output**

Mahi 12345 23 05 1996

Mahi 12345 23 05 1996

**Q9 - Consider that payroll software needs to be developed for computerization of operations of an ABC organization.**

**The organization has employees.**

#include <iostream>

using namespace std;

class Employee

{

int id;

char name[24];

double salary;

double hra, medical = 1000, pf, pt;

double net\_salary, gross\_salary;

public:

Employee()

{

cin >> name >> id >> salary;

hra = 0.5 \* salary;

pf = 0.12 \* salary;

pt = 200;

}

double getGross()

{

return salary + hra + medical;

}

double getNet()

{

return getGross() - (pf + pt);

}

void display()

{

cout << name<<" "<< id<<" " << salary<<" ";

cout << getGross()<<" " << getNet();

}

};

int main()

{

Employee emp;

emp.display();

}

**Out put**

Mahi 1563 25000

Mahi 1563 25000 38500 35300

**Q10 - Fresh business scenario to apply inheritance , polymorphism to emp based organization scenario. Create Emp based organization structure --- Emp , Mgr , Worker**

#include <iostream>

using namespace std;

class Employee

{

int id;

char name[100];

int deptId;

double basicS;

public:

Employee() {}

Employee(int i, int d, char n[], double s)

{

id = i;

deptId = d;

name = n;

basicS = s;

}

double getbasicS()

{

return basicS;

}

virtual double netsalary()

{

return 0;

}

};

class manager : public Employee

{

int perBonus;

public:

manager() {}

manager(int i, int d, char n[], double s, int perf) : Employee(i, d, n, s)

{

perBonus = perf;

}

double netsalary()

{

return getbasicS() + perBonus;

}

};

class worker : public Employee

{

int hoursqworked;

int hourlyrate;

public:

worker() {}

worker(int i, int d, char n[], double s, int h, int hr) : Employee(i, d, n, s)

{

hourlyrate = hr;

hoursqworked = h;

}

double netsalary()

{

return getbasicS() + hourlyrate \* hoursqworked;

}

int gethourlyrate()

{

return hourlyrate;

}

};

int main()

{

int noOfMgr, noOfworker;

cin >> noOfMgr >> noOfworker;

manager \*M = new manager[noOfMgr];

worker \*W = new worker[noOfworker];

int id;

char name[100];

int deptId;

double basicS;

int perBouns;

int hourlyrate;

int hoursworked;

for (int i = 0; i < noOfworker; i++)

{

cin >> name >> id >> deptId >> basicS >> hourlyrate >> hoursworked;

W[i] = worker(id, deptId, basicS, hourlyrate, hoursworked);

}

double sum = 0;

for (int i = 0; i < noOfworker; i++)

{

sum += W[i].getbasicS();

}

for (int i = 0; i < noOfMgr; i++)

{

sum += M[i].getbasicS();

}

cout << sum;

return 0;

}

**Output**

1 M 2 25000 25\*8

1 M 2 25000 200

**Q11 - Create cpp application for bank account handling.**

#include <string.h>

using namespace std;

class account

{

public:

char name[30];

int acc\_num;

int balance;

int amount;

void getData()

{

cout << "\nEnter the following details\nCustomer Name :";

gets(name);

cout << "\nAccount number :";

cin >> acc\_num;

cout << "\nAccount balance:";

cin >> balance;

}

void display()

{

cout << "\nYour Account Balance :" << balance;

}

void withdraw()

{

cout << "\nEnter the amount you want to withdraw :";

cin >> amount;

if (amount > balance)

cout << "\nInsuficient balance";

else

balance = balance - amount;

display();

}

void deposit()

{

cout << "\n Enter the amount you want to deposit :";

cin >> amount;

balance = balance + amount;

display();

}

};

int main()

{

account ac;

ac.getData();

ac.display();

ac.deposit();

ac.withdraw();

getch();

}

Output

Enter the following details

Customer Name :Mahi

Account number :3000

Account balance:5000

Your Account Balance :5000

Enter the amount you want to withdraw :4500

Your Account Balance :500

**Q12 - Create an abstract class Shape with pure virtual method area; Create Rectangle, Circle, Square class. Inherit them from Shape class. Override area method. Test these all classes by creating object of respective class. C++ program to calculate the area of a square and a circle**

#include <iostream>

using namespace std;

class Shape

{

protected:

float dimension;

float height;

public:

void getDimension()

{

cin >> dimension;

}

void getHeight()

{

cin >> height;

}

virtual float calculateArea() = 0;

};

class Square : public Shape

{

public:

float calculateArea()

{

return dimension \* dimension;

}

};

class Circle : public Shape

{

public:

float calculateArea()

{

return 3.14 \* dimension \* dimension;

}

};

class Rectangle : public Shape

{

public:

float calculateArea()

{

return dimension\*height;

}

};

int main()

{

Square square;

Circle circle;

Rectangle Rectangle;

cout << "Enter the length of the square: ";

square.getDimension();

cout << "Area of square: " << square.calculateArea() << endl;

cout << "\nEnter radius of the circle: ";

circle.getDimension();

cout << "Area of circle: " << circle.calculateArea() << endl;

cout << "Enter the length of the Rectangle: ";

Rectangle.getDimension();

cout << "Enter the height of the Rectangle: ";

Rectangle.getHeight();

cout << "Area of Rectangle: " << Rectangle.calculateArea() << endl;

return 0;

}

**Output**

Enter the length of the square: 10

Area of square: 100

Enter radius of the circle: 10

Area of circle: 314

Enter the length of the Rectangle: 10

Enter the height of the Rectangle: 23

Area of square: 230