CEN 591 – Object Oriented Programming Languages Laboratory File

BTech Computer Engineering
Vth Semester

Submitted by:
Aiman Fatima (20BCS008)

Submitted to:
Dr. Wakar Ahmad

Department of Computer Engineering,
Faculty of Engineering & Technology,
Jamia Millia Islamia, New Delhi

2022

Index

------------------------------ Program 1 ---------------------------------

```
// Write a program to implement STUDENT class consisting of name, enrollment_Id and
// marks as class data members. Create three objects for the class using the concept
// of array of objects. Write member functions to read and display the student
// information. Also write the main program to create objects and call the member
// functions from the class.

#include <bits/stdc++.h>
using namespace std;
class Student
{
private:
    string name;
    int enrollment_Id;
    vector<int> marks;

public:
    Student()
    {
    }
    Student(string name_value, int id, vector<int> marks_value)
    {
        this->name = name_value;
        this->enrollment_Id = id;
        int ss = marks_value.size();
        marks.resize(ss);
        this->marks = marks_value;
    }
    void getData()
    {
        cout << "Name : \t" << name << endl;
        cout << "Enrollment Id : " << enrollment_Id << endl;
        for (int i = 0; i < marks.size(); i++)
        {
            cout << "Marks of Subject " << i + 1 << " : " << marks[i] << endl;
        }
    }
};

int main()
{
    cout << "\n____20BCS008 Aiman Fatima____\n";
    cout << "\n----Enter details of three students----\n";
    Student *s = new Student[3];
    vector<int> S;
    for (int i = 0; i < 3; i++)
    {
        cout << "\n|| Enter data of Student " << i + 1 << " ||" << endl;
        cout << "Enter Name: ";
        string name;
        cin >> name;
        cout << "Enter Id: ";
        int id;
        cin >> id;
        cout << "Enter No. of subjects: ";
        int sub;
        cin >> sub;
        S.resize(sub);
        for (int i = 0; i < sub; i++)
        {
            cout << "Enter marks of Subject " << i + 1 << " : ";
```

```
            cin >> S[i];
        }
        s[i] = Student(name, id, S);
        S.clear();
    }
    cout << "\n----Getting details of the students----\n";
    for (int i = 0; i < 3; i++)
    {
        cout << "\n|| Student " << i + 1 << " ||" << endl;
        s[i].getData();
    }
    delete[] s;
    return 0;
}
```

_____ Output _____

```
____20BCS008 Aiman Fatima____

----Enter details of three students----

|| Enter data of Student 1 ||
Enter Name: Aiman
Enter Id: 1
Enter No. of subjects: 2
Enter marks of Subject 1 : 78
Enter marks of Subject 2 : 85

|| Enter data of Student 2 ||
Enter Name: Tree
Enter Id: 2
Enter No. of subjects: 1
Enter marks of Subject 1 : 98

|| Enter data of Student 3 ||
Enter Name: Lynle
Enter Id: 3
Enter No. of subjects: 3
Enter marks of Subject 1 : 23
Enter marks of Subject 2 : 45
Enter marks of Subject 3 : 67
```

```
----Getting details of the students----

|| Student 1 ||
Name :  Aiman
Enrollment Id : 1
Marks of Subject 1 : 78
Marks of Subject 2 : 85

|| Student 2 ||
Name :  Tree
Enrollment Id : 2
Marks of Subject 1 : 98

|| Student 3 ||
Name :  Lynle
Enrollment Id : 3
Marks of Subject 1 : 23
Marks of Subject 2 : 45
Marks of Subject 3 : 67
```

------------------------------ Program 2 ----------------------------------

```
// Write a C++ program handling the following details for
// students and staff using inheritance.
//• Student Details: name, address, percentage marks
//• Staff Details: name, address, salary
// Create appropriate base and derived classes. Input the details and output them.

#include <bits/stdc++.h>
using namespace std;
class School
{
private:
    string name;
    string address;

public:
    School()
    {
```

```
    }
    void getDetails()
    {
        cout << "Enter School name : ";
        cin >> name;
        cout << "Enter address : ";
        cin >> address;
    }
    void showDetails()
    {
        cout << "School Name : " << name << endl;
        cout << "Address : " << address << endl;
    }
};

class Student : public School
{
    string name;
    string roll_no;
    float percentage;

public:
    Student()
    {
    }
    void getDetails()
    {
        School::getDetails();
        cout << "Enter roll no : ";
        cin >> roll_no;
        cout << "Enter name : ";
        cin >> name;
        cout << "Enter percentage : ";
        cin >> percentage;
    }
    void printStudent()
    {
        cout << "\n----Details of the student----\n";
        School::showDetails();
        cout << "Name : " << name << endl;
        cout << "Roll_no : " << roll_no << endl;
        cout << fixed;
        cout << "Percentage : " << setprecision(2) << percentage << endl;
    }
};

class Staff : public School
{
    string name;
    string staff_id;
    float salary;

public:
    Staff()
    {
    }
    void getDetails()
    {
        School::getDetails();
        cout << "Enter staff id: ";
        cin >> staff_id;
        cout << "Enter name: ";
```

```
        cin >> name;
        cout << "Enter salary: ";
        cin >> salary;
    }
    void printStaff()
    {
        cout << "\n----Details of the Staff----\n";
        School::showDetails();
        cout << "Name : " << name << endl;
        cout << "Staff Id : " << staff_id << endl;
        cout << fixed;
        cout << "Percentage : " << setprecision(2) << salary << endl;
    }
};

int main()
{
    cout << "\n____20BCS008 Aiman Fatima____\n";
    Student student;
    Staff staff;
    cout << "\n----Enter details of the student----\n";
    student.getDetails();

    cout << "\n----Enter details of the staff----\n";
    staff.getDetails();
    student.printStudent();
    cout << endl;
    staff.printStaff();
    return 0;
}
```

_____ Output _____

```
____20BCS008 Aiman Fatima____

----Enter details of the student----
Enter School name : VSEC
Enter address : Kanpur
Enter roll no : 1
Enter name : Aiman
Enter percentage : 97

----Enter details of the staff----
Enter School name : JMI
Enter address : Delhi
Enter staff id: 1243
Enter name: Daniel
Enter salary: 5000000

----Details of the student----
School Name : VSEC
Address : Kanpur
Name : Aiman
Roll_no : 1
Percentage : 97.00


----Details of the Staff----
School Name : JMI
Address : Delhi
Name : Daniel
Staff Id : 1243
Percentage : 5000000.00
```

------------------------------ Program 3 ---------------------------------

```
// Write a C++ program to perform the addition of two time
// objects in hour and minute format, display the result in
// hour: minute format using object as a function argument.
#include <bits/stdc++.h>
using namespace std;

class AddTime
{
private:
    int hr;
    int min;

public:
    AddTime()
    {
    }
    AddTime(int h, int m)
    {
        this->hr = h;
        this->min = m;
    }
    AddTime Add(AddTime &a)
    {
        int hh = a.hr;
        int mm = a.min;
        AddTime A;
        int tot_hr = hh + hr;
        int tot_min = mm + min;
        if (tot_min >= 60)
        {
            tot_min -= 60;
            tot_hr += 1;
        }
        A = AddTime(tot_hr, tot_min);
        return A;
    }
    void gettime()
    {
        cout << hr << ": ";
        if (min >= 0 && min < 10)
        {
            cout << "0";
        }
        cout << min;
    }
};
int main()
{
    int h, m;
    cout << "\n____20BCS008 Aiman Fatima____\n";
    cout << "\n----Enter time 1----\n";
    cout << "Enter hour : ";
    cin >> h;
    cout << "Enter minute : ";
    cin >> m;
    AddTime A(h, m);
    cout << "\n----Enter time 2----\n";
    cout << "Enter hour : ";
    cin >> h;
    cout << "Enter minute : ";
```

```
    cin >> m;
    AddTime B(h, m);
    cout << "\n|| Displaying time ||\n";
    cout << "Time 1 -> ";
    A.gettime();
    cout << "\nTime 2 -> ";
    B.gettime();
    AddTime C = A.Add(B);
    cout << "\n\n|| Resultant Time ->  ";
    C.gettime();
    cout << " ||\n\n";
    return 0;
}
```

_____ Output _____

```
____20BCS008 Aiman Fatima____

----Enter time 1----
Enter hour : 13
Enter minute : 25

----Enter time 2----
Enter hour : 10
Enter minute : 56

|| Displaying time ||
Time 1 -> 13: 25
Time 2 -> 10: 56

|| Resultant Time ->  24: 21 ||
```

------------------------------ Program 4 ---------------------------------

```
/*Write a C++ program based on following scenario:
Consider an example of a bookshop which sells books and video tapes. These two classes are
inherited from the base class called media. The media class has command data members such as
title and publication. The book class has data members for storing a number of pages in a
book, and the tape class has the playing time in a tape. Each class will have member functions
such as read() and show(). In the base class, these members have to be defined as virtual
functions. Write a program which models the class hierarchy for the bookshop and processes
objects of these classes using pointers to the base class.*/

#include <bits/stdc++.h>
#include <iostream>
using namespace std;

// base class
// pure virtual function
class Media
{
protected:
    string title;
    string publication;

public:
    virtual void read() = 0;
    virtual void show() = 0;
};
```

```
// derived class
class Book : public Media
{
private:
    int pages;

public:
    void read()
    {
        cout << "Enter title: ";
        cin >> title;
        cout << "Enter publication: ";
        cin >> publication;
        cout << "Enter number of pages: ";
        cin >> pages;
    }
    void show()
    {
        cout << "Title: " << title << endl;
        cout << "Publication: " << publication << endl;
        cout << "Number of pages: " << pages << endl;
    }
};

// derived class
class Tape : public Media
{
private:
    int playingTime;

public:
    void read()
    {
        cout << "Enter title: ";
        cin >> title;
        cout << "Enter publication: ";
        cin >> publication;
        cout << "Enter playing time: ";
        cin >> playingTime;
    }
    void show()
    {
        cout << "Title: " << title << endl;
        cout << "Publication: " << publication << endl;
        cout << "Playing time: " << playingTime << endl;
    }
};

int main()
{
    Media *base_ptr;
    // Media k;
    cout << "\n____20BCS008 Aiman Fatima____\n";
    cout << "\n|| Accessing the details using pointer of base class ||\n";
    cout << "\nEnter details of a book " << endl;
    Book b;
    base_ptr = &b;
    base_ptr->read();
    cout << "\nEnter details of a tape " << endl;
    Tape t;
    base_ptr = &t;
    base_ptr->read();
```

```
    cout << "\nDetails of the book " << endl;
    base_ptr = &b;
    base_ptr->show();
    cout << "\nDetails of the tape " << endl;
    base_ptr = &t;
    base_ptr->show();
    return 0;
}
```

_____ Output _____

```
 ____20BCS008 Aiman Fatima____

 || Accessing the details using pointer of base class ||

 Enter details of a book
 Enter title: Percy
 Enter publication: Penguin
 Enter number of pages: 256

 Enter details of a tape
 Enter title: Mossarte
 Enter publication: Vinyl
 Enter playing time: 120

 Details of the book
 Title: Percy
 Publication: Penguin
 Number of pages: 256

 Details of the tape
 Title: Mossarte
 Publication: Vinyl
 Playing time: 120
```

------------------------------ Program 5 ---------------------------------

```
/*Write a C++ program to overload [] operator for the following scenario:
Create a class AccountBook that contains account holder details such as name and account
number. Take input for 5 account holders in the account table. When we enter account number,
then the program prints account holder name while entering of account holder name, it prints
account number of holder.*/

#include <iostream>
#include <string.h>
using namespace std;

class AccountBook
{
private:
    string name;
    int accountNumber;

public:
    void read();
    void display();
    string operator[](int);
    int operator[](string);
```

```
};

void AccountBook::read()
{
    cout << "\nEnter name: ";
    getchar();
    getline(cin, name);
    cout << "Enter account number: ";
    cin >> accountNumber;
}

void AccountBook::display()
{
    cout << name << "\t" << accountNumber << endl;
}

string AccountBook::operator[](int accountNumber)
{
    if (this->accountNumber == accountNumber)
        return name;
    else
        return "Not found";
}

int AccountBook::operator[](string name)
{
    if (this->name == name)
    {
        return accountNumber;
    }
    else
    {
        return -1;
    }
}

int main()
{
    cout << "\n____20BCS008 Aiman Fatima____\n";
    AccountBook accountBook[5];
    int choice;
    while (1)
    {
        cout << "\n_____Menu_____\n";
        cout << "1. Read 5 details" << endl;
        cout << "2. Display" << endl;
        cout << "3. Search by account number" << endl;
        cout << "4. Search by name" << endl;
        cout << "5. Exit" << endl;
        cout << "Enter your choice : ";
        cin >> choice;
        switch (choice)
        {
        case 1:
            for (int i = 0; i < 5; i++)
            {
                cout << "\n---- Details of customer " << i + 1 << " ----";
                accountBook[i].read();
            }
            break;
        case 2:
            cout << "\n---- Displaying Details ----\n";
```

```
            cout << "\nName\tAccount Number" << endl;
            for (int i = 0; i < 5; i++)
            {
                accountBook[i].display();
            }
            cout << endl;
            break;
        case 3:
        {
            int accountNumber;
            cout << "\nEnter account number: ";
            cin >> accountNumber;
            bool found = false;
            for (int i = 0; i < 5; i++)
            {
                if (accountBook[i][accountNumber] != "Not found")
                {
                    cout << "Name: " << accountBook[i][accountNumber] << endl;
                    found = true;
                }
            }
            if (!found)
            {
                cout << "\nAccount number not found" << endl;
            }
            cout << endl;
            break;
        }
        case 4:
        {
            string name;
            cout << "\nEnter name: ";
            cin >> name;
            bool found = false;
            for (int i = 0; i < 5; i++)
            {
                if (accountBook[i][name] != -1)
                {
                    cout << "Account number: " << accountBook[i][name] << endl;
                    found = true;
                }
            }
            if (!found)
            {
                cout << "\nName not found" << endl;
            }
            cout << endl;
            break;
        }
        case 5:
            cout << "\nThe End.\n\n";
            return 0;
        default:
            cout << "\nWrong choice!!\n";
        }
    }
    return 0;
}
```

```
_______ Output ________

____20BCS008 Aiman Fatima____

_____Menu_____
1. Read 5 details
2. Display
3. Search by account number
4. Search by name
5. Exit
Enter your choice : 1

---- Details of customer 1 ----
Enter name: Aiman
Enter account number: 11223

---- Details of customer 2 ----
Enter name: Tree
Enter account number: 22334

---- Details of customer 3 ----
Enter name: Mouse
Enter account number: 33445

---- Details of customer 4 ----
Enter name: Summer
Enter account number: 44556

---- Details of customer 5 ----
Enter name: Lonely
Enter account number: 55667


_____Menu_____
1. Read 5 details
2. Display
3. Search by account number
4. Search by name
5. Exit
Enter your choice : 3

Enter account number: 11223
Name: Aiman


_____Menu_____
1. Read 5 details
2. Display
3. Search by account number
4. Search by name
5. Exit
Enter your choice : 4

Enter name: 88991

Name not found


_____Menu_____
1. Read 5 details
2. Display
3. Search by account number
4. Search by name
5. Exit
Enter your choice : 2

---- Displaying Details ----

Name    Account Number
Aiman   11223
Tree    22334
Mouse   33445
Summer  44556
Lonely  55667


_____Menu_____
1. Read 5 details
2. Display
3. Search by account number
4. Search by name
5. Exit
Enter your choice : 3

Enter account number: 77889

Account number not found


_____Menu_____
1. Read 5 details
2. Display
3. Search by account number
4. Search by name
5. Exit
Enter your choice : 4

Enter name: Tree
Account number: 22334


_____Menu_____
1. Read 5 details
2. Display
3. Search by account number
4. Search by name
5. Exit
Enter your choice : 6

Wrong choice!!

_____Menu_____
1. Read 5 details
2. Display
3. Search by account number
4. Search by name
5. Exit
Enter your choice : 5

The End.
```

------------------------------ Program 6 ---------------------------------

/*Write a C++ Program to implement Complex class representing complex numbers. A complex number in mathematics is defined as x + iy where x defines the real part of the number and y is the imaginary part. The letter i represents the square root of -1 (which means i2 is -1). Include operator functions to overload the operators +=, -=, *=, /= and the << operator for the class. Here << operator should be used for printing the results of complex number operation.*/

```
#include <bits/stdc++.h>
using namespace std;

class Complex
{
    float real;
    float img;

public:
    Complex()
    {
        real = img = 0;
    }
    // += operator overloading
    void operator+=(Complex c)
    {
        real = real + c.real;
        img = img + c.img;
    }
    // -= operator overloading
    void operator-=(Complex c)
    {
        real = real - c.real;
        img = img - c.img;
    }
    // *= operator overloading
    void operator*=(Complex c)
    {
        real = real * c.real - img * c.img;
        img = real * c.img + img * c.real;
    }
    // /= operator overloading
    void operator/=(Complex c)
    {
        float denominator = c.real * c.real + c.img * c.img;
        real = (real * c.real + img * c.img) / denominator;
        img = (img * c.real - real * c.img) / denominator;
    }
    // Input stream Operator overloading
    friend istream &operator>>(istream &In, Complex &c)
    {
        cout << "Enter Real Part : ";
        In >> c.real;
        cout << "Enter Imaginary Part : ";
        In >> c.img;
        return In;
    }
    // Output stream Operator overloading
    friend ostream &operator<<(ostream &Out, Complex &c)
    {
        Out << fixed;
        Out << "Resulatnt Complex Number : (" << setprecision(2) << c.real;
        Out << " + i(" << setprecision(2) << c.img << "))" << endl;
```

```
        return Out;
    }
};

int main()
{
    Complex c1, c2, c3;
    cout << "\n____20BCS008 Aiman Fatima____\n";
    cout << "\n|| Arithmetic Assignment Operators and Stream Operators overloading ||\n";
    cout << "\nEnter Complex Number c1\n";
    cin >> c1;
    cout << "Enter Complex Number c2\n";
    cin >> c2;
    c3 = c1;
    c3 += c2;
    cout << "\n----| Addition : c1 + c2 |----\n";
    cout << c3;
    c3 = c1;
    c3 -= c2;
    cout << "\n----| Substraction : c1 + c2 |----\n";
    cout << c3;
    c3 = c1;
    c3 *= c2;
    cout << "\n----| Multiplication : c1 + c2 |----\n";
    cout << c3;
    c3 = c1;
    c3 /= c2;
    cout << "\n----| DivisionI : c1 + c2 |----\n";
    cout << c3;
}
```

_____ Output _____

```
____20BCS008 Aiman Fatima____

|| Arithmetic Assignment Operators and Stream Operators overloading ||

Enter Complex Number c1
Enter Real Part : 12
Enter Imaginary Part : 5
Enter Complex Number c2
Enter Real Part : 3.4
Enter Imaginary Part : 7.9

----| Addition : c1 + c2 |----
Resulatnt Complex Number : (15.40 + i(12.90))

----| Substraction : c1 + c2 |----
Resulatnt Complex Number : (8.60 + i(-2.90))

----| Multiplication : c1 + c2 |----
Resulatnt Complex Number : (1.30 + i(27.27))

----| DivisionI : c1 + c2 |----
Resulatnt Complex Number : (1.09 + i(0.11))
```

------------------------------ Program 7 ---------------------------------

```
/*Design classes such that they support the following statements: Rupee rl, r2; Dollar dl, d2;
dl = r2; // converts rupee (Indian currency) to dollar (US currency) r2 = d2; // converts
dollar (US currency) to rupee (Indian currency) Write a complete program which does such
conversions according to the world market value.*/

#include <bits/stdc++.h>
#include <iostream>
#include <vector>
#include <iomanip>
#include <string>
using namespace std;

// current value of 1 USD in INR is 81.79 INR
const float VAL = 81.79;

// class rupee
class rupee
{
    float r;

public:
    rupee()
    {
        r = 0.0;
    }

    void inINR(float a)
    {
        r = a;
    }

    rupee(float dlr)
    {
        r = dlr;
    }

    void printinr()
    {
        cout << fixed;
        cout << "\nValue in INR : " << setprecision(2) << r << endl;
    }
    float getInr()
    {
        return r;
    }
};

// class dollar
class dollar
{
    float d;

public:
    dollar()
    {
        d = 0.0;
    }

    void inDlr(float a)
    {
```

```
        d = a;
    }

    dollar(rupee rr)
    {
        d = rr.getInr() / VAL;
    }

    operator rupee()
    {
        return (rupee(d * VAL));
    }
    void printdlr()
    {
        cout << fixed;
        cout << "\nValue in Dollar : " << setprecision(2) << d << "\n"
             << endl;
    }
};

// main function
int main()
{
    rupee r1, r2;
    dollar d1, d2;
    float a, b;
    cout << "\n____20BCS008 Aiman Fatima____\n";
    cout << "\n<Current value of 1 USD in INR is 81.79>\n";
    cout << "\nEnter value in dollar(s) : ";
    cin >> a;
    d1.inDlr(a);
    cout << "\n----Operator overloading in source class----";
    r1 = d1; // operator overloading in source class
    r1.printinr();
    ///////////////
    cout << "\nEnter value in rupee(s) : ";
    cin >> b;
    r2.inINR(b);
    cout << "\n----Constructor in destination class----";
    d2 = r2; // constructor in destination class
    d2.printdlr();
    return 0;
}
```

_____ Output _____

```
____20BCS008 Aiman Fatima____

<Current value of 1 USD in INR is 81.79>

Enter value in dollar(s) : 8.6

----Operator overloading in source class----
Value in INR : 703.39

Enter value in rupee(s) : 59876.99

----Constructor in destination class----
Value in Dollar : 732.08
```

```
------------------------------ Program 8 ---------------------------------

// Write suitable C++ program to implement following OOPS concepts:
// (a) Pure Virtual Function
// (b) Pointers to Derived Class Object
// (c) Virtual Destructor
// (d) Overloading through friend function
#include <bits/stdc++.h>
using namespace std;

class Base
{
public:
    Base()
    {
        cout << "\nBase class created\n";
    }
    // Virtual Destructor
    virtual ~Base()
    {
        cout << "\nBase class destructor!" << endl;
    }
    // Pure virtual function
    virtual void DisplayAction()
    {
        cout << "\nDispay function of Base class called\n";
    }
};

class Derived : public Base
{
public:
    Derived()
    {
        cout << "\nDerived class created" << endl;
    }

    ~Derived()
    {
        cout << "\nDerived class destructor!" << endl;
    }

    void DisplayAction()
    {
        cout << "\nDisplay of Derived class called!" << endl;
    }
};

class Overloading_Through_Friend
{
private:
    int number;
public:
    Overloading_Through_Friend(int r = 0)
    {
        number = r;
    }
    void Display()
    {
        cout << number;
    }    friend Overloading_Through_Friend operator+(Overloading_Through_Friend c1,
Overloading_Through_Friend c2);
```

```
};

Overloading_Through_Friend operator+(Overloading_Through_Friend c1, Overloading_Through_Friend
c2)
{
    Overloading_Through_Friend temp;
    temp.number = c1.number + c2.number;
    return temp;
}

int main()
{
    cout << "\n____20BCS008 Aiman Fatima____\n";
    Base *obj;
    // pointer to derived class
    obj = new Derived();
    obj->DisplayAction();

    Overloading_Through_Friend C1(4), C2(8), C3;
    cout << endl;
    C1.Display();
    cout << " + ";
    C2.Display();
    cout << " = ";
    C3 = C1 + C2;
    cout << "\n";
    C3.Display();
    obj->~Base();
    cout << endl;
    return 0;
}
```

_____ Output _____

```
____20BCS008 Aiman Fatima____

Base class created

Derived class created

Display of Derived class called!

4 + 8 =
12
Derived class destructor!

Base class destructor!
```

------------------------------ Program 9 ---------------------------------

```
/*Write a java program to create an abstract class named Shape that contains two integers and
an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle
such that each one of the classes extends the class Shape. Each one of the classes contain
only the method printArea( ) that prints the area of the given shape.*/

import java.util.Scanner;

abstract class Shape {

  int a, b;

  abstract void printArea();
```

```
}

class Rectangle extends Shape {

  void printArea() {
    System.out.println("\nArea of Rectangle: " + (a * b));
  }
}

class Triangle extends Shape {

  void printArea() {
    System.out.println("\nArea of Triangle: " + (0.5 * a * b));
  }
}

class Circle extends Shape {

  void printArea() {
    System.out.println("\nArea of Circle: " + (3.14 * a * a));
  }
}

public class P9_abstract_class {

  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    Rectangle r = new Rectangle();
    Triangle t = new Triangle();
    Circle c = new Circle();

    System.out.println("\nEnter the length and breadth of rectangle: ");
    r.a = sc.nextInt();
    r.b = sc.nextInt();
    System.out.println("\nEnter the base and height of triangle: ");
    t.a = sc.nextInt();
    t.b = sc.nextInt();
    System.out.println("\nEnter the radius of circle: ");
    c.a = sc.nextInt();
    r.printArea();
    t.printArea();
    c.printArea();
    sc.close();
  }
}
```

_____ Output _____

```
Enter the length and breadth of rectangle:
45 36

Enter the base and height of triangle:
77 11

Enter the radius of circle:
32

Area of Rectangle: 1620

Area of Triangle: 423.5

Area of Circle: 3215.36
```

```
------------------------------ Program 10 ---------------------------------

/*  Write a java program that implements a multi-thread application that has three
 threads. First thread generates random integer every 1 second and if the value is even,
 second thread computes the square of the number and prints. If the value is odd, the third
 thread will print the value of cube of the number.
*/
import java.util.Random;

class RandomNumberThread extends Thread {

  public void run() {
    Random random = new Random();
    for (int i = 0; i < 10; i++) {
      int randomInteger = random.nextInt(100);
      System.out.println("\nRandom Integer generated: " + randomInteger);
      if ((randomInteger % 2) == 0) {
        SquareThread sThread = new SquareThread(randomInteger);
        sThread.start();
      } else {
        CubeThread cThread = new CubeThread(randomInteger);
        cThread.start();
      }
      try {
        Thread.sleep(1000);
      } catch (InterruptedException ex) {
        System.out.println(ex);
      }
    }
    System.out.println("\n");
  }
}

class SquareThread extends Thread {

  int number;

  SquareThread(int randomNumber) {
    number = randomNumber;
  }

  public void run() {
    System.out.println("Square of " + number + "=" + (number * number));
  }
}

class CubeThread extends Thread {

  int number;

  CubeThread(int randomNumber) {
    number = randomNumber;
  }

  public void run() {
    System.out.println("Cube of " + number + "=" + (number * number * number));
  }
}

class P10_multithreading {

  public static void main(String[] args) {
```

```
    RandomNumberThread rnThread = new RandomNumberThread();
    rnThread.start();
  }
}
```

_____ Output _____

```
Random Integer generated: 14
Square of 14=196

Random Integer generated: 71
Cube of 71=357911

Random Integer generated: 50
Square of 50=2500

Random Integer generated: 10
Square of 10=100

Random Integer generated: 89
Cube of 89=704969

Random Integer generated: 31
Cube of 31=29791

Random Integer generated: 61
Cube of 61=226981

Random Integer generated: 1
Cube of 1=1

Random Integer generated: 43
Cube of 43=79507

Random Integer generated: 45
Cube of 45=91125
```