**Object Oriented Programming**

**Lab Report**

**Lab07**



|  |  |
| --- | --- |
| Group Members Name & Reg #: | **Muhammad Haris Irfan**  **(FA18-BCE-090)** |
|  |  |
| Class | Object Oriented Programming CSC241 (**BCE-4B**) |
| Instructor’s Name | Maam Amber Madeeha Zeb |

**In Lab Tasks**

5.1 Question 1:

**Imagine a publishing company that markets both book and audio-cassette versions of**

**its works. Create a class publication that stores the title and price of a publication.**

**a. from this class derive two classes:**

* **book, which adds a page count and**
* **tape, which adds a playing time in minutes.**

**each of these three classes should have getdata() function to get its data from the**

**user at the keyboard and a putdata() function to display its data.**

**b. Write a main() program to test the book and tape class by creating instances of them,**

**asking the user to fill in their data with getdata() and then displaying the data with**

**putdata().**

**Solution:**The code is given below,

1 #include <iostream>

2

3 **using namespace std**;

4

5 **class** publication

6 {

7 **protected**:

8

9 **string** title;

10 **float** price;

11 **public**:

12

13 **void** getData()

14 {

15 **cout**<<"Enter title of Publication?"<<**endl**;

16 **cin**>>title;

17 **cout**<<"Enter price of Publication?"<<**endl**;

18 **cin**>>price;

19 }

20 **void** putData()

21 {

22 **cout**<<"\n------------------------------------------------\n";

23 **cout**<<"Title: "<<title<<**endl**;

24 **cout**<<"Price: "<<price<<**endl**;

25 }

26

27 };

28

29 **class** book:**public** publication

30 {

31 **private**:

32 **int** page\_count;

33 **public**:

34 **void** getData()

35 {

36 publication::getData();

37 **cout**<<"Enter page count of Book?"<<**endl**;

38 **cin**>>page\_count;

39 }

40 **void** putData()

41 {

42 publication::putData();

43 **cout**<<"Page count: "<<page\_count<<**endl**;

44 **cout**<<"\n------------------------------------------------\n";

45 }

46

47

48 };

49 **class** tape:**public** publication

50 {

51 **private**:

52 **int** playing\_time\_min;

53 **public**:

54 **void** getData()

55 {

56 publication::getData();

57 **cout**<<"Enter playing time(in mins) of Tape?"<<**endl**;

58 **cin**>>playing\_time\_min;

59 }

60

61 **void** putData()

62 {

63

64 publication::putData();

65 **cout**<<"Playing time in Mins: "<<playing\_time\_min<<**endl**;

66 **cout**<<"\n------------------------------------------------\n";

67 }

68

69 };

70

71 **int** main()

72 {

73 publication p;

74 book b;

75 tape t;

76

77

78 b.getData();

79 b.putData();

80

81 t.getData();

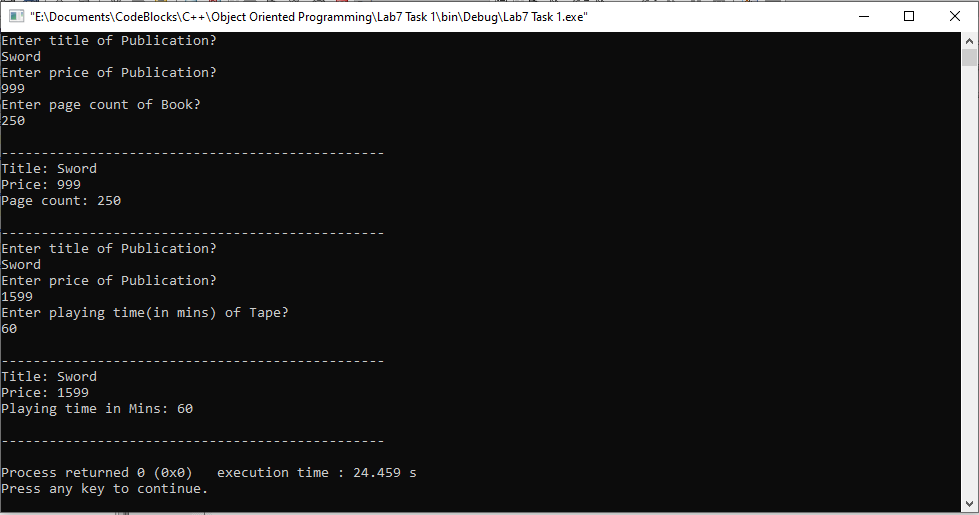
82 t.putData();

83

84 **return** 0;

85 }

**Console Output is shown below.**

****

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5.2 Question 2:

**Write a class Person that has attributes of id, name and address. It has a constructor to initialize, a member function to input and a member function to display data members. Create another class Student that inherits Person class. It has additional attributes of rollnumber and marks. It also has member function to input and display its data members.**

**Solution:**The code is given below,

1 #include <iostream>

2

3 **using namespace std**;

4

5 **class** person

6 {

7 **private**:

8 **int** id;

9 **string** name;

10 **string** address;

11

12 **public**:

13

14

15 person()

16 {

17 id=0;

18 name="name";

19 address="address";

20 }

21 **void** getData()

22 {

23 **cout**<<"Enter id?"<<**endl**;

24 **cin**>>id;

25 **cout**<<"Enter name?"<<**endl**;

26 **cin**>>name;

27 **cout**<<"Enter address?"<<**endl**;

28 **cin**>>address;

29

30 }

31

32 **void** showData()

33 {

34 **cout**<<"\n------------------------------------------------\n";

35 **cout**<<"ID :"<< id<<**endl**;

36 **cout**<<"Name:"<<name<<**endl**;

37 **cout**<<"Address:"<<address<<**endl**;

38

39 }

40

41 };

42

43 **class** students:**public** person

44 {

45 **private**:

46 **int** roll;

47 **int** marks;

48

49 **public**:

50 **void** getData()

51 {

52 person::getData();

53 **cout**<<"Enter roll number?"<<**endl**;

54 **cin**>>roll;

55 **cout**<<"Enter Marks?"<<**endl**;

56 **cin**>>marks;

57 }

58 **void** showData()

59 {

60 person::showData();

61 **cout**<<"Roll Number: "<<roll<<**endl**;

62 **cout**<<"Marks : "<<marks<<**endl**;

63 **cout**<<"\n------------------------------------------------\n";

64 }

65

66 };

67 **int** main()

68 {

69

70 students one;

71

72 one.getData();

73 one.showData();

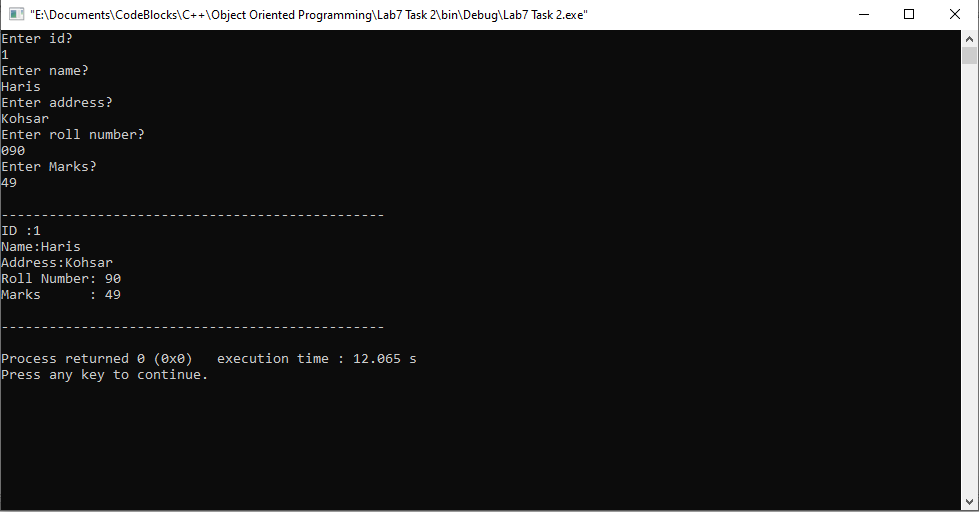
74

75

76 **return** 0;

77 }

**Console Output is shown below.**

****

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5.3 Question 3:

**Write a base class Computer that contains data members of wordsize(in bits), memorysize (in megabytes), storagesize (in megabytes) and speed (in megahertz). Derive a Laptop class that is a kind of computer but also specifies the object’s length, width, height, and weight. Member functions for both classes should include a default constructor, a constructor to inialize all components and a function to display data members.**

**Solution:**The code is given below,

1 #include <iostream>

2

3 **using namespace std**;

4

5 **class** Computer

6 {

7 **protected**:

8 **int** wordsize;

9 **int** memorysize;

10 **int** storagesize;

11 **int** speed;

12

13 **public**:

14

15 Computer()

16 {

17 wordsize=0;

18 memorysize=0;

19 storagesize=0;

20 speed=0;

21 }

22

23 **void** getData()

24 {

25 **cout**<<"Enter wordsize?"<<**endl**;

26 **cin**>>wordsize;

27 **cout**<<"Enter memorysize?"<<**endl**;

28 **cin**>>memorysize;

29 **cout**<<"Enter storagesize?"<<**endl**;

30 **cin**>>storagesize;

31 **cout**<<"Enter speed?"<<**endl**;

32 **cin**>>speed;

33 }

34 **void** showData()

35 {

36 **cout**<<"\n------------------------------------------------\n";

37 **cout**<<"Word size :"<<wordsize<<**endl**;

38 **cout**<<"Memory size:"<<memorysize<<**endl**;

39 **cout**<<"Storage size:"<<storagesize<<**endl**;

40 **cout**<<"Speed :"<< speed<<**endl**;

41

42 }

43 };

44 **class** laptop: **public** Computer

45 {

46 **private**:

47 **int** length;

48 **int** width;

49 **int** height;

50 **int** weight;

51 **public**:

52 laptop()

53 {

54 length=0;

55 width=0;

56 height=0;

57 weight=0;

58 }

59 **void** getData()

60 {

61 Computer:: getData();

62 **cout**<<"Enter Length?"<<**endl**;

63 **cin**>>length;

64 **cout**<<"Enter width?"<<**endl**;

65 **cin**>>width;

66 **cout**<<"Enter height?"<<**endl**;

67 **cin**>>height;

68 **cout**<<"Enter weight?"<<**endl**;

69 **cin**>>weight;

70 }

71 **void** showData()

72 {

73 Computer::showData();

74

75 **cout**<<"Length:"<<length<<**endl**;

76 **cout**<<"Width:" <<width<<**endl**;

77 **cout**<<"Height:"<<height<<**endl**;

78 **cout**<<"Weigh :"<<weight<<**endl**;

79 **cout**<<"\n------------------------------------------------\n";

80 }

81

82 };

83

84 **int** main()

85 {

86 laptop l;

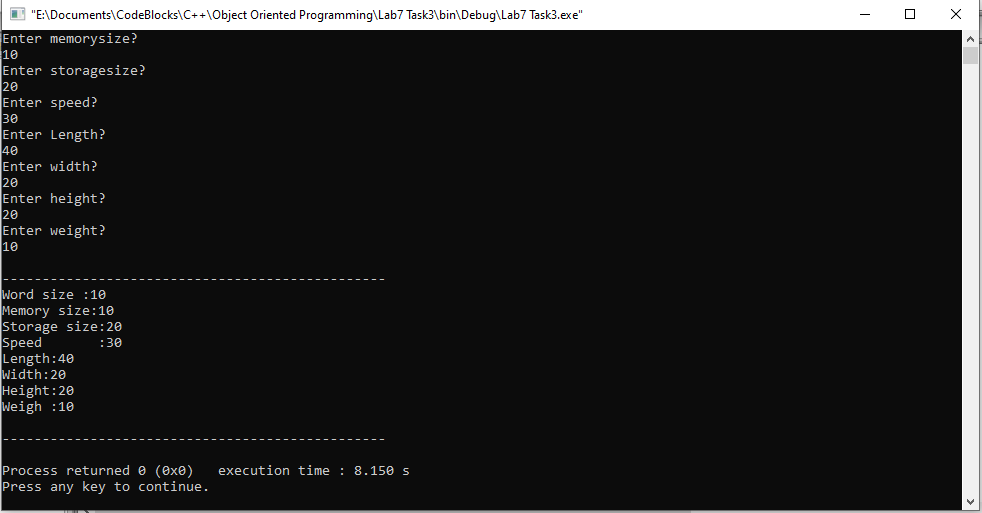
87 l.getData();

88 l.showData();

89 **return** 0;

90 }

**Console Output is shown below.**

****

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

POST LAB

6.1 Question 4:

Write a program having a base class Student with data members rollno, name and Class define a member functions getdata() to input values and another function putdata() to display all values. A class Test is derived from class Student with data members T1marks, T2marks, T3marks, Sessional1, Sessional2, Assignment and Final. Also make a function getmarks() to enter marks for all variables except Final and also make a function putmarks() to display result. Make a function Finalresult() to calculate value for final variable using other marks. Then display the student result along with student data.

Solution:

I am attaching my code below,

1 #include <iostream>

2

3 **using namespace std**;

4

5 **class** student

6 {

7 **protected**:

8 **int** roll;

9 **int** classs;

10 **string** name;

11

12

13 **public**:

14 **void** getdata()

15 {

16 **cout** << "Enter Name?!" << **endl**;

17 **cin**>>name;

18 **cout** << "Enter Roll Number?" << **endl**;

19 **cin**>>roll;

20 **cout** << "Enter Class?" << **endl**;

21 **cin**>>classs;

22

23 }

24 **void** putdata()

25 {

26 **cout** << "Name:" <<name<< **endl**;

27 **cout** << "Roll Number:"<<roll << **endl**;

28 **cout** << "Class:" <<classs<< **endl**;

29

30 }

31 };

32

33 **class** test:**public** student

34 {

35 **private**:

36

37 **int** t1marks;

38 **int** t2marks;

39 **int** t3marks;

40 **int** S1;

41 **int** S2;

42 **int** asse;

43 **int** finall;

44

45 **public**:

46

47 **void** getmarks()

48 {

49 **cout** << "Enter T1 marks?" << **endl**;

50 **cin**>>t1marks;

51 **cout** << "Enter T2 marks?" << **endl**;

52 **cin**>>t2marks;

53 **cout** << "Enter T3 marks?" << **endl**;

54 **cin**>>t3marks;

55 **cout** << "Enter Sessional-l marks?" << **endl**;

56 **cin**>>S1;

57 **cout** << "Enter Sessional-2 marks?" << **endl**;

58 **cin**>>S2;

59 **cout** << "Enter Assignment marks?" << **endl**;

60 **cin**>>asse;

61

62 }

63

64 **void** putmarks()

65 {

66 **cout** << "T1 Marks:" <<t1marks<< **endl**;

67 **cout** << "T2 Marks:"<<t2marks << **endl**;

68 **cout** << "T3 Marks:" <<t3marks<< **endl**;

69 **cout** << "S1 Marks:" <<S1<< **endl**;

70 **cout** << "S2 Marks:"<<S2 << **endl**;

71 **cout** << "Assignment Marks:" <<asse<< **endl**;

72

73 }

74

75 **void** finalresult()

76 {

77 finall=t1marks+t2marks+t3marks+S1+S2+asse;

78 **cout**<<"-----------------------------------------"<<**endl**;

79 putdata();

80 **cout**<<"Final Result: "<<finall<<**endl**;

81 **cout**<<"-----------------------------------------"<<**endl**;

82 }

83

84

85 };

86 **int** main()

87 {

88 test t1;

89 t1.getdata();

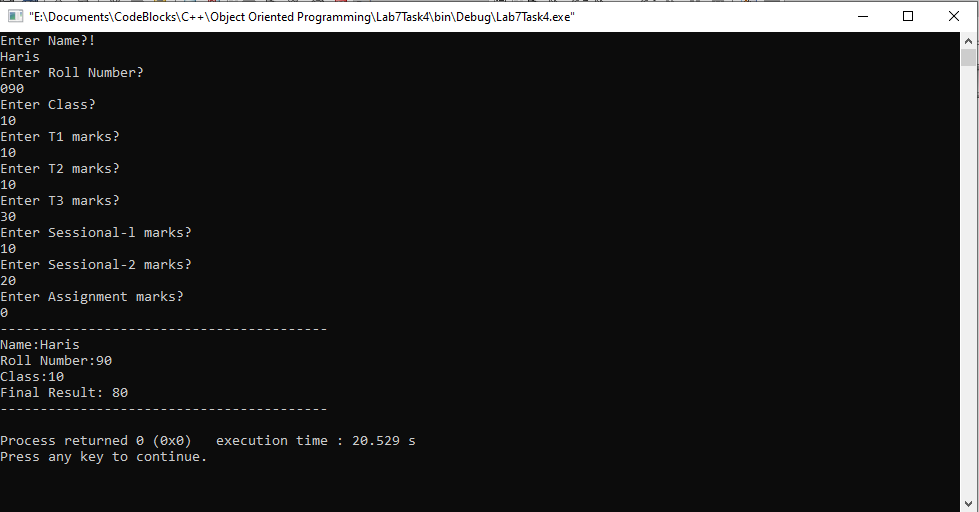
90 t1.getmarks();

91 t1.finalresult();

92 **return** 0;

93 }

**The result for this program is shown below,**



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6.2 Question 5:

Write a program that declares two classes. The parent class is called Simple that has

two data members num1 and num2 to store two numbers. It also has four member

functions.

.

The add() function adds two numbers and displays the result.

.

The sub() function subtracts two numbers and displays the result.

.

The mul() function multiplies two numbers and displays the result.

.

The div() function divides two numbers and displays the result.

The child class is called Complex that overrides all four functions. Each function in

the child class checks the value of data members. It calls the corresponding member

function in the parent class if the values are greater than 0. Otherwise it displays error

message.

Solution:

I am attaching my code below,

1 #include <iostream>

2

3 **using namespace std**;

4

5 **class** simple

6 {

7 **protected**:

8 **int** num1;

9 **int** num2;

10 **public**:

11

12 **void** get()

13 {

14 **cout**<<"Enter value for num1?"<<**endl**;

15 **cin**>>num1;

16 **cout**<<"Enter value for num2?"<<**endl**;

17 **cin**>>num2;

18

19 }

20 **void** add()

21 {

22 **if**(num1<0 ||num2<0)

23 {

24 **cout**<<"error"<<**endl**;

25

26 }

27 **else**

28 **cout**<<"Value after adding: "<<num1+num2<<**endl**;

29 }

30 **void** sub()

31 {

32 **if**(num1<0 ||num2<0)

33 {

34 **cout**<<"error"<<**endl**;

35

36 }

37 **else**

38 **cout**<<"Value after subtracting: "<<num1-num2<<**endl**;

39 }

40 **void** mul()

41 {

42 **if**(num1<0 ||num2<0)

43 {

44 **cout**<<"error"<<**endl**;

45

46 }

47 **else**

48 **cout**<<"Value after multiplying: "<<num1\*num2<<**endl**;

49 }

50 **void** div()

51 {

52 **if**(num1<0 ||num2<0)

53 {

54 **cout**<<"error"<<**endl**;

55

56 }

57 **else**

58 **cout**<<"Value after dividing: "<<num1/num2<<**endl**;

59 }

60 };

61

62 **class** com :**public** simple

63 {

64 **public**:

65

66

67

68 **void** add()

69 {

70 simple::add();

71 }

72 **void** sub()

73 {

74 simple::sub();

75 }

76 **void** mul()

77 {

78 simple::mul();

79 }

80 **void** div()

81 {

82 simple::div();

83 }

84 };

85 **int** main()

86 {

87 com s1;

88 s1.get();

89 s1.add();

90 s1.sub();

91 s1.mul();

92 s1.div();

93

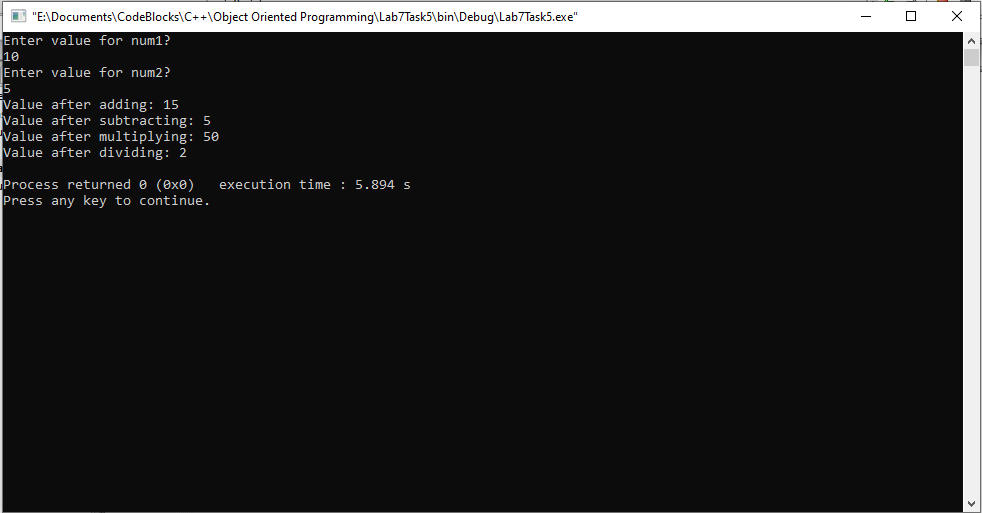
94

95

96

97 }

**The result for this program is shown below,**



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6.3 Question 6:

An electricity board charges the following rates to domestic users to discourage large

consumption of energy.

.

For the first 100 units - 50 P per unit

.

Beyond 100 units - 60 P per unit

If the total cost is more than Rs.250.00 then an additional surcharge of 15% is added on the difference. Define a class Electricity in which the function Bill computes the cost. Define a derived class More\_Electricity and override Bill to add the surcharge.

Solution:

I am attaching my code below,

1 #include <iostream>

2

3 **using namespace std**;

4

5 **class** Electricity

6 {

7 **protected**:

8 **int** units=0;

9 **int** cost=0;

10 **int** factor;

11 **int** surcharge=0;

12

13 **public**:

14 **void** getunits()

15 {

16 **cout**<<"Enter consumed units?"<<**endl**;

17 **cin**>>units;

18 }

19 **void** bill()

20 {

21 **for**(**int** i=0;i<=units;i++)

22 {

23 **if**(i>0 && i<=100)

24 {

25 factor=50;

26 cost=cost+(factor);

27 }

28 **if**(i>100)

29 {

30 factor=60;

31 cost=cost+(factor);

32 }

33

34 }

35

36 **if**(cost>250)

37 {

38 surcharge=15/100\*cost;

39 }

40 cost=cost+surcharge;

41 }

42 };

43 **class** More\_electricity:**public** Electricity

44 {

45

46 **public**:

47

48 **void** bill()

49 {

50 Electricity::bill();

51 }

52 **void** showbill()

53 {

54 **cout**<<"Your units consumed: "<<units<<**endl**;

55 **cout**<<"Your bill: "<<cost<<**endl**;

56

57 }

58 };

59 **int** main()

60 {

61 More\_electricity m1;

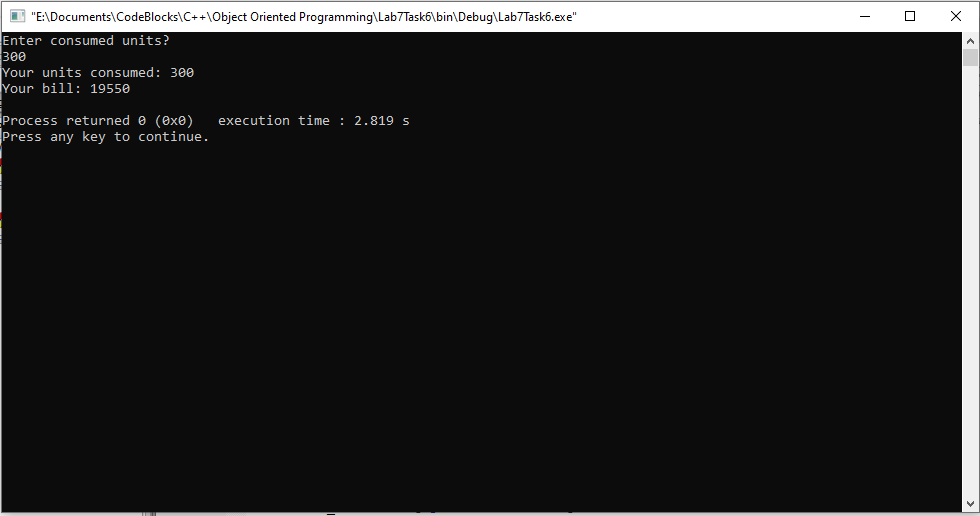
62 m1.getunits();

63 m1.bill();

64 m1.showbill();

65 }

**The result for this program is shown below,**



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6.4 Question 7:

(Package Inheritance Hierarchy) Package-delivery services, such as FedEx®, DHL

S and UPS®, offer a number of different shipping options, each with specific costs

associated. Create an inheritance hierarchy to represent various types of packages. Use

Package as the base class of the hierarchy, then include classes TwoDayPackage and

OvernightPackage that derive from Package. Base class Package should include data

members representing the name, address, city, state and ZIP code for both the sender

and the recipient of the package, in addition to data members that store the weight (in

ounces) and cost per ounce to ship the package. Package's constructor should initialize

these data members. Ensure that the weight and cost per ounce contain positive

values. Package should provide a public member function calculateCost() that returns

a double indicating the cost associated with shipping the package. Package's

calculateCost() function should determine the cost by multiplying the weight by the

cost per ounce. Derived class TwoDayPackage should inherit the functionality of base

class Package, but also include a data member that represents a flat fee that the

shipping

company

charges

for

two-day-delivery

service.

TwoDayPackage's constructor should receive a value to initialize this data member. TwoDayPackage

should redefine member function calculateCost() so that it computes the shipping cost

by adding the flat fee to the weight-based cost calculated by base class Package's

calculateCost() function. Class OvernightPackage should inherit directly from class

Package and contain an additional data member representing an additional fee per

ounce charged for overnight-delivery service. OvernightPackage should redefine

member function calculateCost() so that it adds the additional fee per ounce to the

standard cost per ounce before calculating the shipping cost. Write a test program that

creates objects of each type of Package and tests member function calculateCost().

Solution:

I am attaching my code below,

1 #include <iostream>

2

3 **using namespace std**;

4

5 **class** package

6 {

7 **protected**:

8 **string** name;

9 **string** address;

10 **string** city;

11

12 **string** rname;

13 **string** raddress;

14 **string** rcity;

15

16 **int** weightounce;

17 **int** costperounce;

18 **int** cost;

19

20 **public**:

21 package()

22 {

23 **cout** << "Enter Your Name?" << **endl**;

24 **cin**>>name;

25 **cout** << "Enter your Address?" << **endl**;

26 **cin**>>address;

27 **cout** << "Enter your City?" << **endl**;

28 **cin**>>city;

29 **cout** << "Enter Receivers Name?" << **endl**;

30 **cin**>>rname;

31 **cout** << "Enter Receivers Address?" << **endl**;

32 **cin**>>raddress;

33 **cout** << "Enter Receivers City?" << **endl**;

34 **cin**>>rcity;

35

36 **cout**<<"-------------------------------------------"<<**endl**;

37 **cout** << "Enter Weight of your Package?" << **endl**;

38 **cin**>>weightounce;

39 **cout** << "Enter Cost per ounce of package?" << **endl**;

40 **cin**>>costperounce;

41 **cout**<<"-------------------------------------------"<<**endl**;

42

43

44 }

45

46 **double** calculateCost()

47 {

48 cost=weightounce\*costperounce;

49 **return** cost;

50 }

51

52 };

53 **class** twodaypackage:**public** package

54 {

55 **public**:

56 **int** flatfee;

57 twodaypackage(**int** j)

58 {

59 flatfee=j;

60 }

61

62 **double** calculateCost()

63 {

64 package::calculateCost();

65 cost=cost+flatfee;

66 **return** cost;

67 }

68 };

69 **class** overnightpackage:**public** package

70 {

71 **public**:

72 **int** addchargesPerounce;

73

74 overnightpackage(**int** j)

75 {

76 addchargesPerounce=j;

77 }

78 **double** calculateCost()

79 {

80 costperounce=costperounce+addchargesPerounce;

81 package::calculateCost();

82 **return** cost;

83 }

84

85 };

86 **int** main()

87 {

88 **int** cost2day,overpkg;

89 **cout**<<"For overnight package:"<<**endl**<<**endl**;

90 twodaypackage t1(100);

91 **cout**<<"Now for overnight package:"<<**endl**<<**endl**;

92 overnightpackage o1(10);

93 cost2day=t1.calculateCost();

94 overpkg=o1.calculateCost();

95

96

97 **cout**<<"Cost of two day package is: "<<cost2day<<**endl**;

98

99 **cout**<<"Cost of Overnight package is: "<<overpkg<<**endl**;

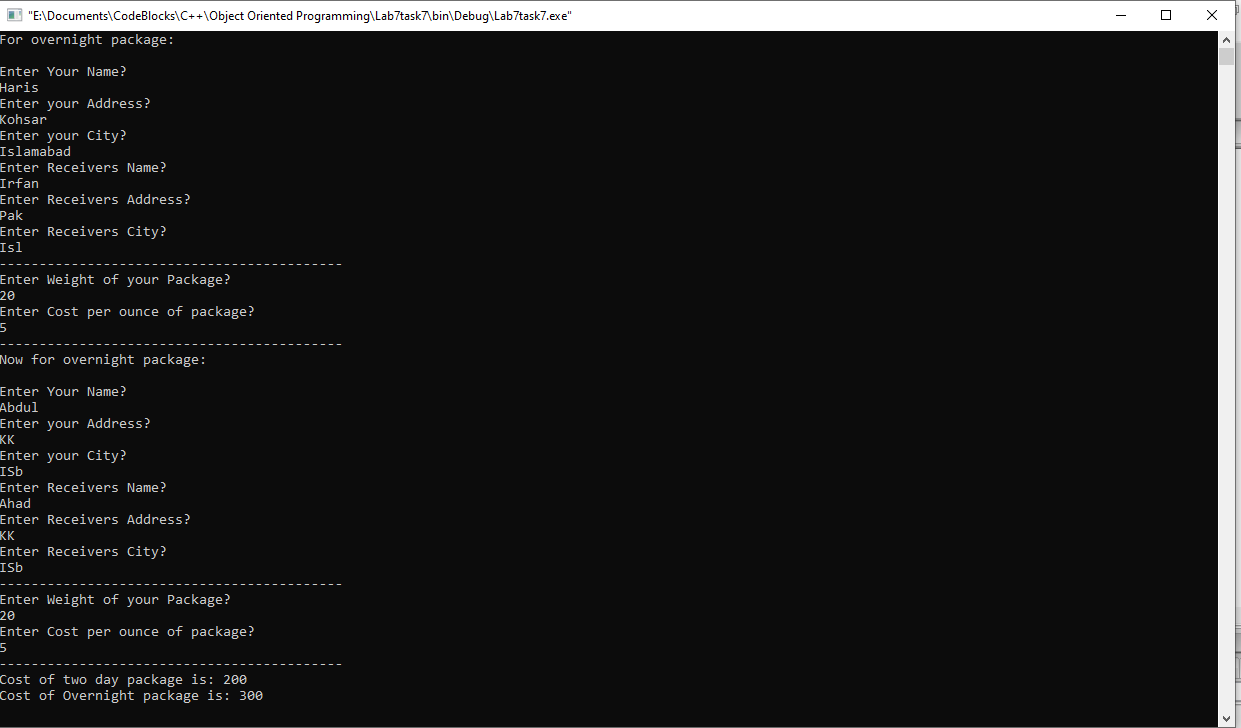
100

101

102 **return** 0;

103 }

**The result for this program is shown below,**



\_\_\_THE END\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_