**Object Oriented Programming**

**Lab 9**

**Submitted To:**

Ma’am Amber Madeeha Zeb

**Submitted By:**

Manaal Waseem

FA18-BCE-074

**In Lab:**

**Task 1:**

**Code the example given above with mentioning some message indicating the class construction and destruction in the constructors and destructors of each class and check the calling of constructors and destructors.**

**Code:**

1 #include<iostream>

2 #include<string>

3

4 **using namespace std**;

5

6 **class** studentRecord

7 {

8 **private**:

9 **string** degree;

10 **public**:

11 studentRecord()

12 {

13 **cout**<< "I am constructor of Student Record." <<**endl**;

14 }

15

16 ~studentRecord()

17 {

18 **cout**<< "I am destructor of Student Record." <<**endl**;

19 }

20

21 **void** getdata()

22 {

23 **cout**<<"Enter Degree: ";

24 **cin**>>degree;

25 }

26 };

27

28 **class** employeeRecord

29 {

30 **private**:

31 **int** emp\_id;

32 **double** salary;

33

34 **public**:

35 employeeRecord()

36 {

37 **cout**<< "I am constructor of Employee Record." <<**endl**;

38 }

39

40 ~employeeRecord()

41 {

42 **cout**<< "I am destructor of Employee Record." <<**endl**;

43 }

44

45 **void** getdata()

46 {

47 **cout**<<"Enter Employee ID: ";

48 **cin**>>emp\_id;

49 **cout**<<"Enter Salary: ";

50 **cin**>>salary;

51 }

52 };

53

54 **class** manager

55 {

56 **private**:

57 **string** title;

58 **double** dues;

59 employeeRecord emp;

60 studentRecord stu;

61 **public**:

62 manager()

63 {

64 **cout**<< "I am constructor of Manager." <<**endl**;

65 }

66

67 ~manager()

68 {

69 **cout**<< "I am destructor of Manager." <<**endl**;

70 }

71

72 **void** getdata()

73 {

74 emp.getdata();

75 **cout**<<"Enter Title: ";

76 **cin**>>title;

77 **cout**<<"Enter Dues: ";

78 **cin**>>dues;

79 stu.getdata();

80 }

81 };

82

83 int main()

84 {

85 manager m1;

86 cout<<"Enter data for manager 1: ";

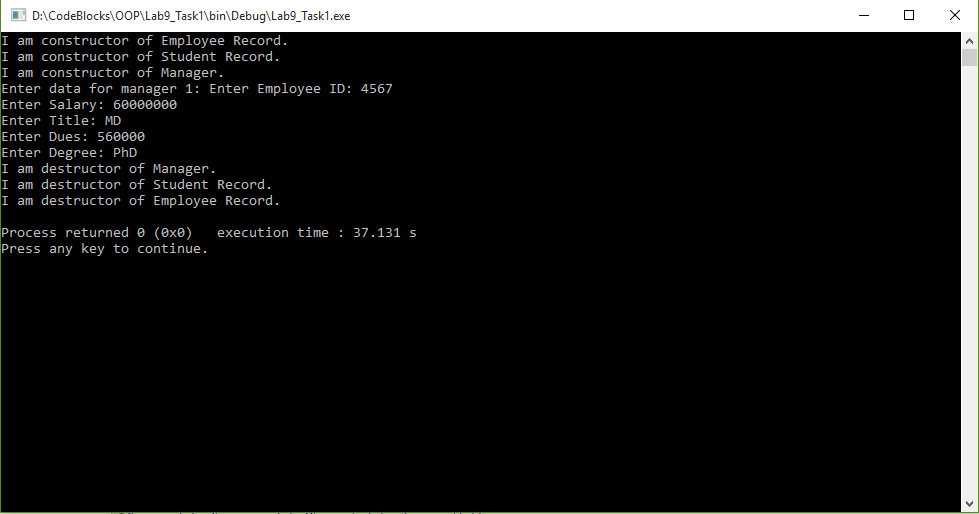
87 m1.getdata();

88

89 return 0;

90 }

**Output:**



**Task 2:**

**Create an Address class, which contains street#, house#, city and code (all of type char\*). Create another class Person that contains an address of type Address. Give appropriate get and set functions for both classes. Test class person in main.**

**Code:**

1 #include<iostream>

2 #include<string>

3

4 **using namespace std**;

5

6 **class** Address

7 {

8 **private**:

9 **char**\* street;

10 **char**\* house;

11 **char**\* city;

12 **char**\* code;

13

14 **public**:

15 Address()

16 {

17 street = **new char**[3];

18 house = **new char**[3];

19 city = **new char**[15];

20 code = **new char**[5];

21 }

22

23 ~Address()

24 {

25 **delete** street;

26 **delete** house;

27 **delete** city;

28 **delete** code;

29 }

30

31 **void** set\_data()

32 {

33 **cout**<< "Enter Street No: " <<**endl**;

34 **cin**>> street;

35 **cout**<< "Enter House No: " <<**endl**;

36 **cin**>> house;

37 **cout**<< "Enter City: " <<**endl**;

38 **cin**>> city;

39 **cout**<< "Enter Code: " <<**endl**;

40 **cin**>> code;

41 }

42

43 **void** get\_data()

44 {

45 **cout**<< "Street No: " << street <<**endl**;

46 **cout**<< "House No: " << house <<**endl**;

47 cout<< "City: " << city <<endl;

48 cout<< "Code: " << code <<endl;

49 }

50 };

51

52 class Person

53 {

54 private:

55 string name;

56 Address address;

57

58 public:

59 void set\_data()

60 {

61 cout<< "Enter Name: " <<endl;

62 cin>>name;

63

64 address.set\_data();

65 }

66

67 void get\_data()

68 {

69 cout<< endl << "Name: " << name <<endl;

70

71 address.get\_data();

72 }

73 };

74

75 int main()

76 {

77 Person p1;

78

79 p1.set\_data();

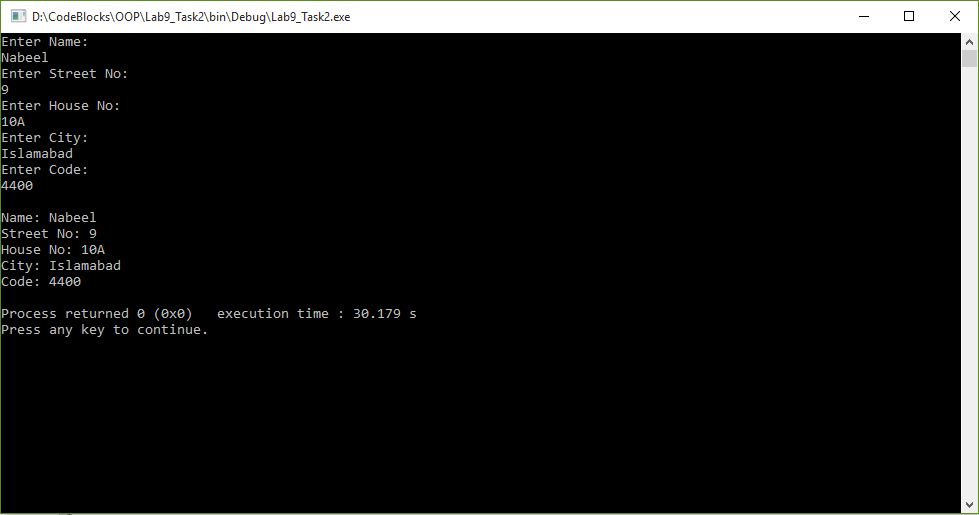
80 p1.get\_data();

81

82 return 0;

83 }

**Output:**



**Task 3:**

**Write the program, which has two classes one, is Date having members (day, month, year) and the other class is called Employee. The employee has Date class as member as each employee has Date of joining, Date of Birth etc.**

**a. Determine if an employee joined the organization within last five years if the current year is 2012.**

**b. Determine if an Employee has age less than 40 years?**

**Code:**

1 #include<iostream>

2 #include<string>

3

4 **using namespace std**;

5

6 **class** Date

7 {

8 **private**:

9 **int** day;

10 **int** month;

11 **int** year;

12

13 **public**:

14 **void** set\_data()

15 {

16 **cout**<< "Enter Day: " <<**endl**;

17 **cin**>> day;

18 **cout**<< "Enter Month: " <<**endl**;

19 **cin**>> month;

20 **cout**<< "Enter Year: " <<**endl**;

21 **cin**>> year;

22 }

23

24 **void** get\_data()

25 {

26 **cout**<< "Day: " << day <<**endl**;

27 **cout**<< "Month: " << month <<**endl**;

28 **cout**<< "Year: " << year <<**endl**;

29 }

30

31 **int** difference(**int** current\_year)

32 {

33 **return** (current\_year - year);

34 }

35 };

36

37 **class** Employee

38 {

39 **private**:

40 **string** name;

41 Date date\_of\_joining;

42 Date Date\_of\_Birth;

43

44 **public**:

45 **void** set\_data()

46 {

47 **cout**<< "Enter Name: " <<**endl**;

48 **cin**>>name;

49

50 **cout**<< "Enter Date of Joining: " <<**endl**;

51 date\_of\_joining.set\_data();

52

53 **cout**<< "Enter Date of Birth: " <<**endl**;

54 Date\_of\_Birth.set\_data();

55

56

57 }

58

59 **void** get\_data()

60 {

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

62

63 **cout**<< "Date of Joining: " <<**endl**;

64 date\_of\_joining.get\_data();

65

66 **cout**<< "Date of Birth: " <<**endl**;

67 Date\_of\_Birth.get\_data();

68 }

69

70 **void** years\_employment(**int** current\_year)

71 {

72 **if** (date\_of\_joining.difference(current\_year) <=5)

73 **cout** << **endl** << "Employee has joined the organization within last five years." << **endl**;

74 **else**

75 **cout** << **endl** << "Employee has not joined the organization within last five years." << **endl**;

76 }

77

78 **void** age(**int** current\_year)

79 {

80 **if** (Date\_of\_Birth.difference(current\_year) <40)

81 **cout** << **endl** << "Employee is less than 40 years of age." << **endl**;

82 **else**

83 **cout** << **endl** << "Employee is more than 40 years of age." << **endl**;

84 }

85 };

86

87 **int** main()

88 {

89 Employee e1;

90 **int** cYear;

91

92 e1.set\_data();

93 e1.get\_data();

94

95 **cout** << **endl** << "Enter current year" << **endl**;

96 **cin** >> cYear;

97

98 e1.years\_employment(cYear);

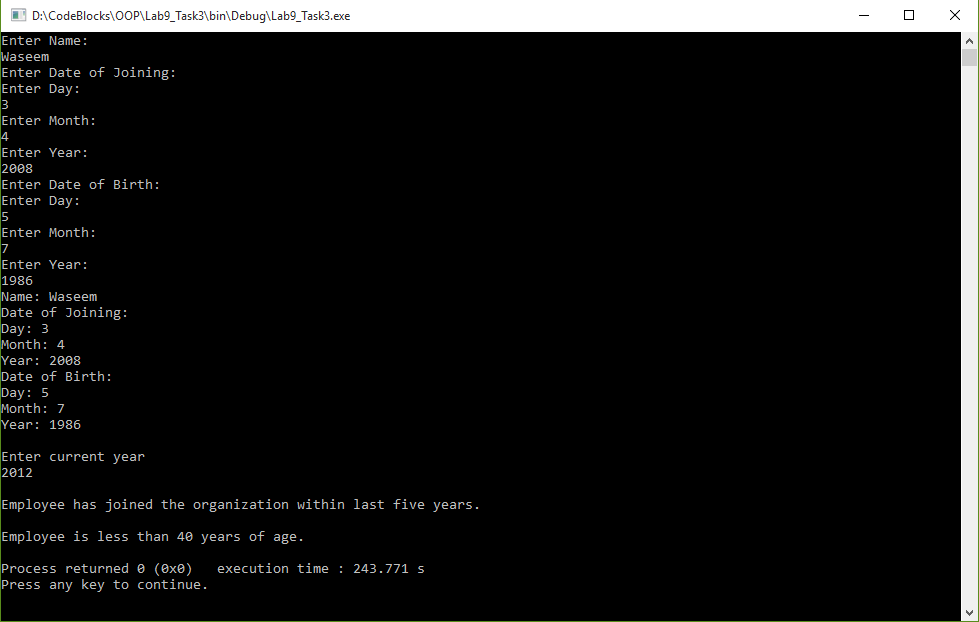
99 e1.age(cYear);

100

101 **return** 0;

102 }

**Output:**



**Post Lab:**

**Task 1:**

**Read digits as Characters input and convert to equivalent numeral values.**

**Code:**

1 #include <iostream>

2 #include <string>

3

4 **using namespace std**;

5

6 **int** main()

7 {

8 **string** num;

9 **int** dec = 0, i, j, len;

10 **cout**<< "Enter a number: "<<**endl**;

11 **cin**>>num;

12 len = num.length();

13 **for**(i=0; i<len; i++)

14 {

15 dec = dec \* 10 + ( num[i] - '0' );

16 }

17 **cout**<< "Digits: " << dec <<**endl**;

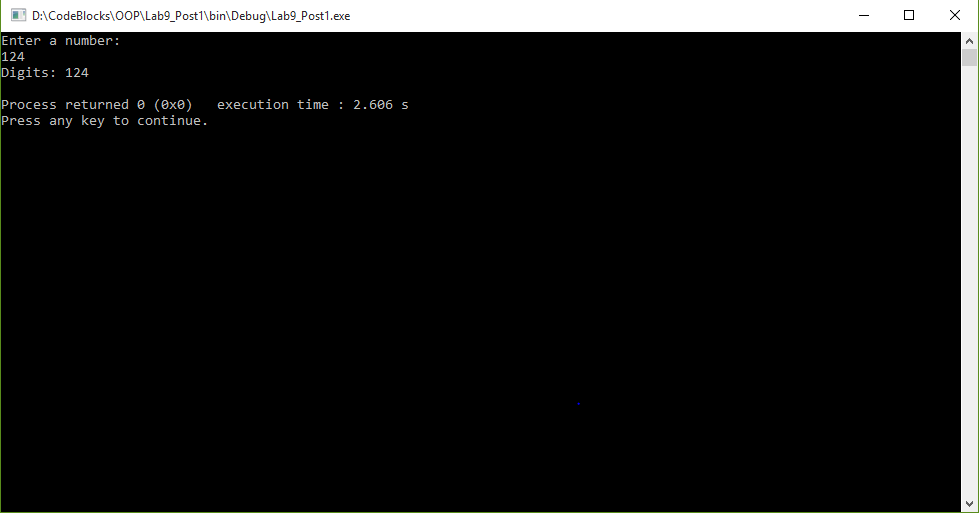
18

19 **return** 0;

20

21 }

**Output:**



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**THE END**