**Object Oriented Programming**

**Lab Report**

**Lab12**



|  |  |
| --- | --- |
| 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:

****

**Solution:**

The code is given below,

1 #include <iostream>

2

3 **using namespace std**;

4

5

6 **class** counter

7 {

8 **private**:

9 **int** cnt=0;

10 **public**:

11

12 counter **operator** ++ ()

13 {

14 cnt=cnt+1;

15 }

16

17 counter **operator** ++ (**int**)

18 {

19 cnt=cnt-1;

20 }

21

22 **int** display()

23 {

24 **int** c;

25 c=cnt;

26

27 **return** c;

28 }

29 };

30

31

32 **int** main()

33 {

34 counter c1;

35 **int** n,i,j;

36 **cout**<<"Enter the number whose table you want?"<<**endl**;

37 **cin**>>n;

38 **for**(i=1;i<11;i++)

39 { **for**(j=1;j<=n;j++)

40 {

41 ++c1;

42 }

43 **cout**<<j-1<<" x "<<i<<" ="<<c1.display()<<**endl**;

44 }

45

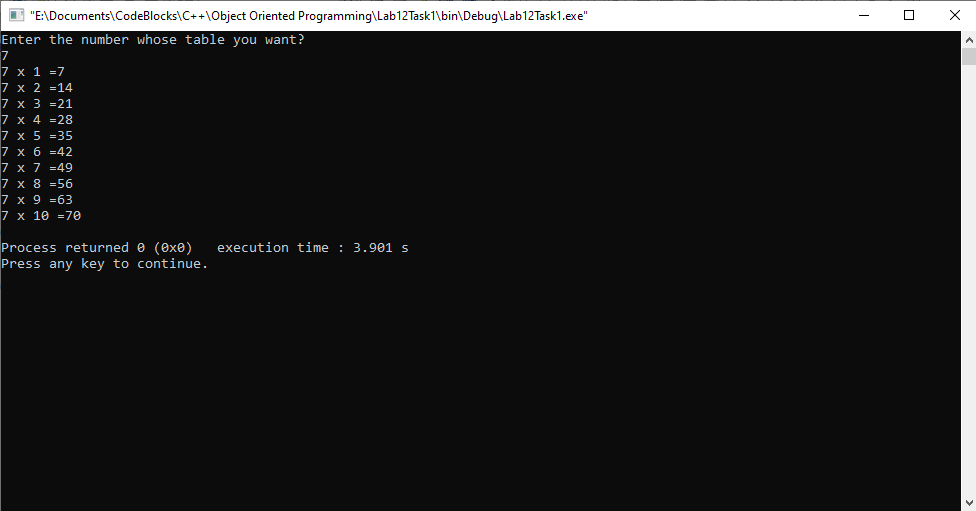
46

47

48 **return** 0;

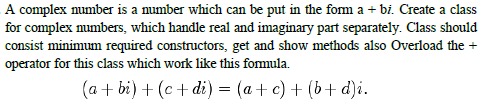
49 }

**Console Output is shown below.**

****

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

5.2 Question 2:

****

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

1 #include <iostream>

2

3 **using namespace std**;

4

5 **class** complexN

6 {

7 **private**:

8 **float** real;

9 **float** img;

10

11 **public**:

12 complexN()

13 {

14 real=0.0;

15 img=0.0;

16 }

17

18 **void** get()

19 {

20 **cout**<<"Enter the real part?"<<**endl**;

21 **cin**>>real;

22 **cout**<<"Enter the imaginary part?"<<**endl**;

23 **cin**>>img;

24 }

25

26 **void** show()

27 {

28 **cout**<<**endl**;

29 **cout**<<"Real part : "<<real<<**endl**;

30 **cout**<<"Imaginary part: "<<img<<**endl**;

31 **cout**<<"("<<real<<","<<img<<"i)"<<**endl**;

32

33 }

34

35 complexN **operator** +(complexN c1)

36 {

37 complexN c2;

38

39 c2.real=real+c1.real;

40 c2.img=img+c1.img;

41

42 **return** c2;

43 }

44

45

46 };

47

48

49

50

51

52 **int** main()

53 {

54 complexN c1,c2,c3;

55

56 c1.get();

57 c2.get();

58

59 c3=c1+c2;

60

61 c1.show();

62 c2.show();

63 c3.show();

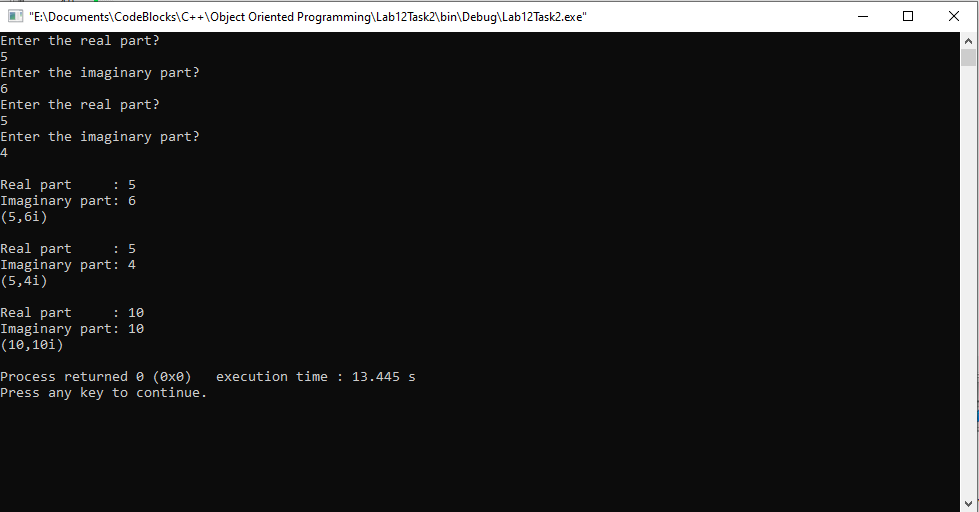
64

65

66 **return** 0;

67 }

**Console Output is shown below.**

****

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

5.3 Question 3:

****

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

1 #include <iostream>

2

3 **using namespace std**;

4

5 **class** distancee

6 {

7 **private**:

8 **int** feet;

9 **int** inch;

10

11 **public**:

12

13 distancee()

14 {

15 feet=0;

16 inch=0;

17 }

18 **void** get()

19 {

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

21 **cin**>>feet;

22 **cout**<<"Enter inches?"<<**endl**;

23 **cin**>>inch;

24

25 **while**(inch>=12)

26 {

27 feet++;

28 inch=inch-12;

29 }

30 }

31 **void** show()

32 {

33 **cout**<<**endl**;

34 **cout**<<"Feet : "<<feet<<**endl**;

35 **cout**<<"Inches : "<<inch<<**endl**;

36 **cout**<<feet<<"'"<<inch<<"''"<<**endl**;

37

38 }

39

40 distancee **operator** %(distancee d1)

41 {

42 distancee d2;

43

44

45

46 d2.feet=feet % d1.feet;

47 d2.inch=inch % d1.inch;

48

49

50 **return** d2;

51 }

52 };

53 **int** main()

54 {

55

56 distancee d1,d2,d3;

57

58 d1.get();

59 d2.get();

60

61 d3=d1%d2;

62

63 d1.show();

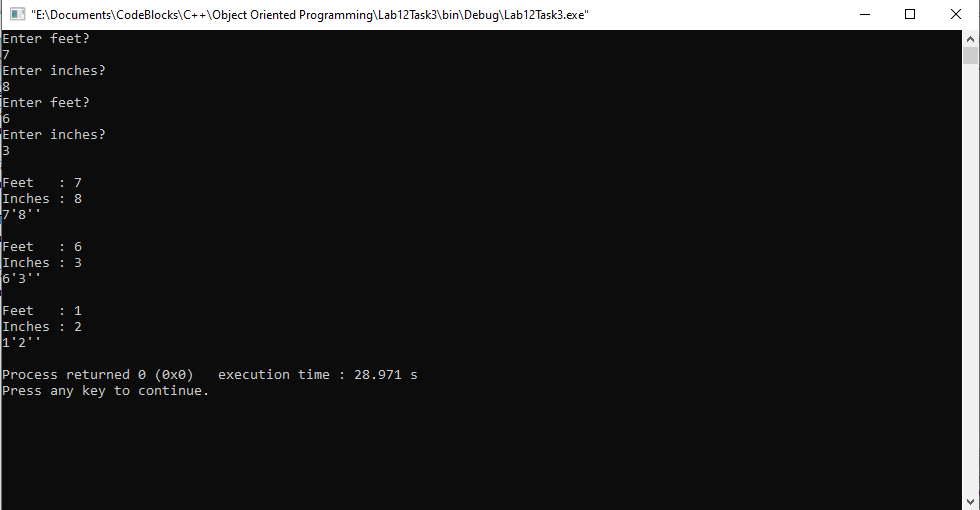
64 d2.show();

65 d3.show();

66 **return** 0;

67 }

**Console Output is shown below.**

****

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

POST LAB

6.1 Question 4:



Solution:

I am attaching my code below,

1 #include <iostream>

2

3 **using namespace std**;

4

5 **class** complexN

6 {

7 **private**:

8 **int** value;

9

10 **public**:

11

12 complexN()

13 {

14 value=0;

15 }

16

17 **int** disp()

18 {

19 **return** value;

20 }

21 complexN **operator** ++()

22 {

23 value++;

24 }

25 complexN **operator** --()

26 {

27 value--;

28 }

29 complexN **operator** + (complexN c1)

30 {

31 complexN c2;

32 c2.value= value+ c1.value;

33 **return** c2;

34

35 }

36 complexN **operator** - (complexN c1)

37 {

38 complexN c2;

39 c2.value=value-c1.value;

40 **return** c2;

41 }

42 complexN **operator** \* (complexN c1)

43 {

44 complexN c2;

45 c2.value=value\*c1.value;

46 **return** c2;

47 }

48 complexN **operator** / (complexN c1)

49 {

50 complexN c2;

51 c2.value=value/c1.value;

52 **return** c2;

53 }

54 };

55

56 **int** main()

57 {

58 complexN c1, c2 ,c3;

59

60 ++c1;

61 ++c1;

62 ++c1;

63 ++c1;

64 ++c1;

65 ++c1;

66 --c1;

67 --c1;

68

69 ++c2;

70 ++c2;

71

72

73 **cout**<<"Value of C1 :"<<c1.disp()<<**endl**;

74 **cout**<<"Value of C2 :"<<c2.disp()<<**endl**;

75 c3=c1+c2;

76 **cout**<<**endl**<<"Value after adding is : "<<c3.disp()<<**endl**;

77

78 c3=c1-c2;

79 **cout**<<"Value after subtracting is : "<<c3.disp()<<**endl**;

80

81 c3=c1\*c2;

82 **cout**<<"Value after multiplying is : "<<c3.disp()<<**endl**;

83

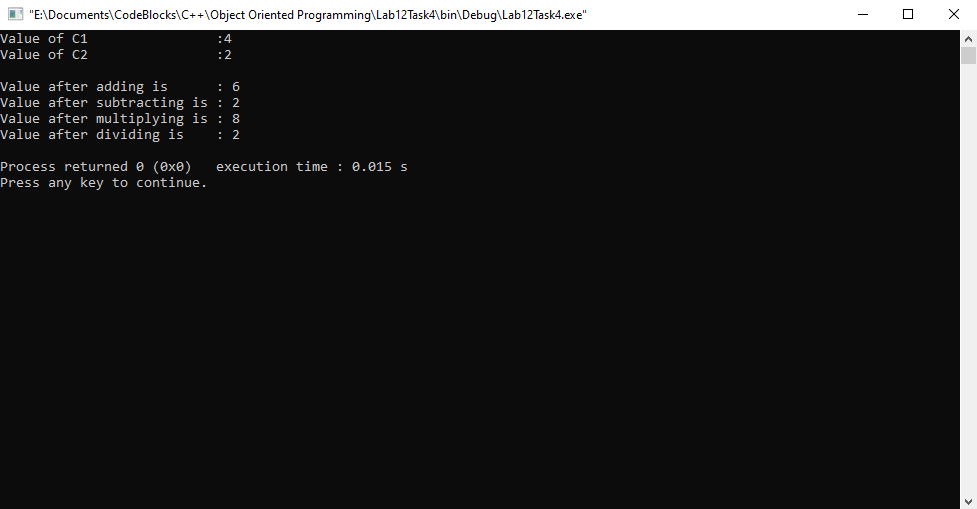
84 c3=c1/c2;

85 **cout**<<"Value after dividing is : "<<c3.disp()<<**endl**;

86 **return** 0;

87 }

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

****

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

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