**Object Oriented Programming**

**Lab 14**

**Submitted To:**

Ma’am Amber Madeeha Zeb

**Submitted By:**

Manaal Waseem

FA18-BCE-074

**In Lab:**

**Task 2:**

**Create a class which only works for absolute numbers, if it encounters any negative occurrence, then it throw an exception to its handler and display errors.**

**Code:**

1 #include <iostream>

2 #include <string>

3

4 **using namespace std**;

5

6 **class** exception1

7 {

8 **public**:

9 **int** check(**int** val)

10 {

11 **try** // start of the try block

12 {

13 **if**(val<0) // throw exception if FirstValue is = 0

14 **throw** val;

15 **else**

16 **cout**<< "Absolute value is: "<<val<<**endl**<<**endl**;

17 } // end of the try block

18 **catch**( **int** ) // start of the catch block

19 {

20 **cout**<< "Error!!!!!! Negative value entered \n"<<**endl**<<**endl**;

21 }// end of the catch block

22 }

23 };

24

25 **int** main()

26 {

27 exception1 e1;

28

29 e1.check(12);

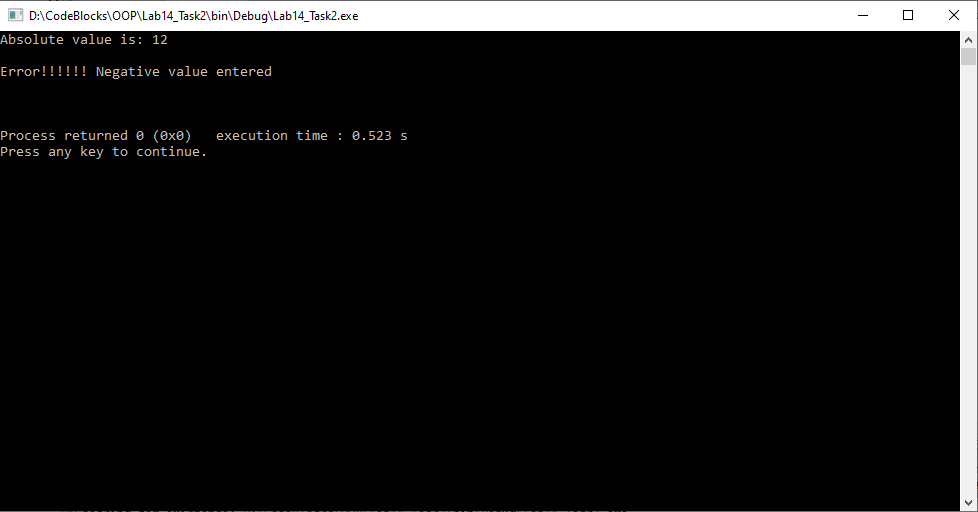
30 e1.check(-5);

31

32 **return** 0;

33 }

**Output:**



**Task 3:**

**Modify the above task, by creating an exception class with an error code and corresponding error message. Code and message should be thrown and displayed in catch block.**

**Code:**

1 #include<iostream>

2 #include<conio.h>

3

4 **using namespace std**;

5

6 **class** exception1

7 {

8 **public**:

9 **int** checking(**int** f1)

10 {

11 **try**

12 {

13 **if** (f1%2==0)

14 {

15 **cout**<<"The no. is a even number\n\n";

16 }

17 **else**

18 **throw** f1;

19 }

20 **catch** (**int** )

21 {

22 **cout**<< "Error!! It is an odd number \n\n";

23 }

24 }

25 };

26

27 **void** Subtract (**int** FirstValue, **int** SecondValue)

28 {

29 **cout**<< "Inside Subtraction Function \n";

30 **try**

31 {

32 **if**(FirstValue==0)

33 **throw** FirstValue;

34 **else**

35 **cout**<< "Subtraction =" <<FirstValue-SecondValue <<**endl**;

36 }

37 **catch**( **int** )

38 {

39 **cout**<< "Caught Null Value \n";

40 **throw**;

41 }

42 **cout**<< "End of Subtract Function \n \n";

43 }

44

45 **int** main()// start of the main

46 {

47 exception1 e1;

48

49 e1.checking(8);

50 e1.checking(33);

51

52 **cout**<<"Inside Main Function: \n";

53 **try**

54 {

55 Subtract(8, 5);

56 Subtract(0, 8);

57 }

58 **catch** (**int** )

59 {

60 **cout**<< "Caught Null inside Main \n";

61 }

62

63 **cout**<< "End of Main Function \n";

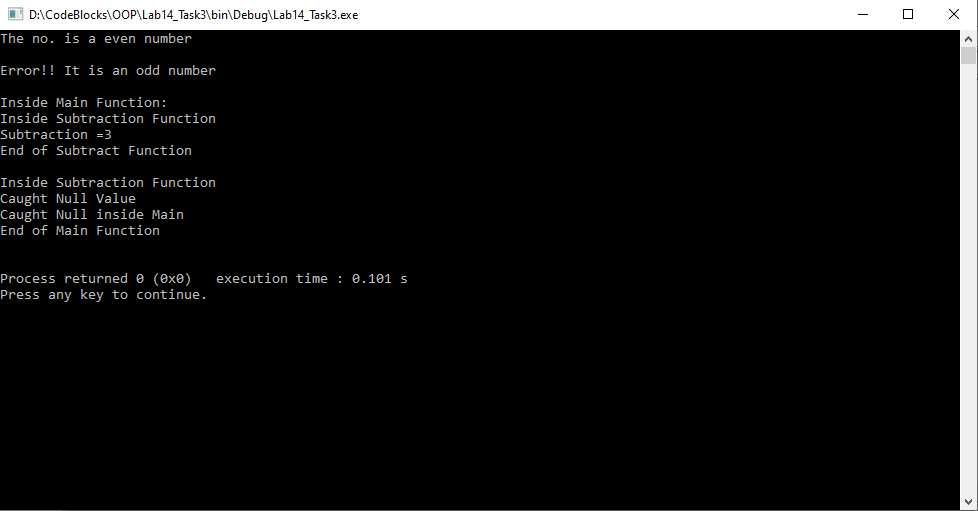
64 **cout**<<**endl**;

65

66 **return** 0;

67 }

**Output:**



**Post Lab:**

**Task 1:**

**Write a program to create a queue class and do queue operations with exception handling.**

**Code:**

1 #include<iostream>

2

3 #include<iomanip>

4

5 **using namespace std**;

6

7 **class queue**

8 {

9 **private**:

10 **int** \*q;

11 **int max**, **front**, rear, cnt;

12 **public**:

13 **class** FULL{}; //for exception handling

14 **class** EMPTY{}; //for exception handling

15 **queue**(**int**);

16 **void** enqueue(**int**);

17 **int** dequeue(**void**);

18 **void** display(**void**);

19 };

20

21 **queue**::**queue**(**int** m)

22 {

23 q=**new int**[m];

24 rear=0;

25 **front**=0;

26 cnt=0;

27 **max**=m;

28 }

29

30 **void queue**::enqueue(**int** item)

31 {

32 **if**(cnt<**max**)

33 {

34 **front** = **front**%**max**;

35 q[**front**++]=item;

36 cnt++;

37 }

38 **else**

39 **throw** FULL(); //FULL object is thrown

40 }

41

42 **int queue**::dequeue(**void**)

43 {

44 **if**(cnt>0)

45 {

46 cnt--;

47 rear = rear %**max**;

48 **return** q[rear++];

49 }

50 **else**

51 **throw** EMPTY(); //EMPTY object is thrown

52 }

53

54 **void queue**::display(**void**)

55 {

56 **if**(cnt>0)

57 **for**(**int** i=0, j=**front**; i<cnt;i++,j++)

58 **cout**<<"|"<<q[j%**max**]<<"|";

59 **else**

60 **throw** EMPTY();

61 }

62

63 **int** main()

64 {

65 **int** item, **size**;

66 **int** ch=1;

67

68 **cout**<<"\nEnter the size of the queue:";

69 **cin**>>**size**;

70

71 **queue** q(**size**);

72

73 **cout**<<"\nQueue Operations using Exception Handling";

74 **cout**<<"\n\n\tMENU\n1.ENQUEUE\n2.DEQUEUE\n3.SHOW QUEUE\n4.EXIT";

75 **cout**<<"\nEnter your choice:";

76 **cin**>>ch;

77

78 **do**

79 {

80 **switch**(ch)

81 {

82 **case** 1:

83 **cout**<<"\nEnter the item to insert in to the queue:";

84 **cin**>>item;

85 **try**

86 {

87 q.enqueue(item);

88 }

89 **catch**(**queue**::FULL) //FULL object is caught

90 {

91 **cout**<<"\n\*\*\*Queue Full\*\*\*\n";

92 }

93 **break**;

94

95 **case** 2:

96 **try**

97 {

98 **cout**<<"\nRemoved Item from the Queue"<<q.dequeue();

99 }

100 **catch**(**queue**::EMPTY) //EMPTY object is caught

101 {

102 **cout**<<"\n\*\*\*Queue Empty\*\*\*\n";

103 }

104 **break**;

105

106 **case** 3:

107 **cout**<<"\nThe Queue is:\n";

108 **try**

109 {

110 q.display();

111 }

112 **catch**(**queue**::EMPTY)

113 {

114 **cout**<<"\n\*\*\*Queue Empty\*\*\*\n";

115 }

116 **break**;

117

118 **case** 4:

119 exit(0);

120 }

121

122 **cout**<<"\nEnter your choice:";

123 **cin**>>ch;

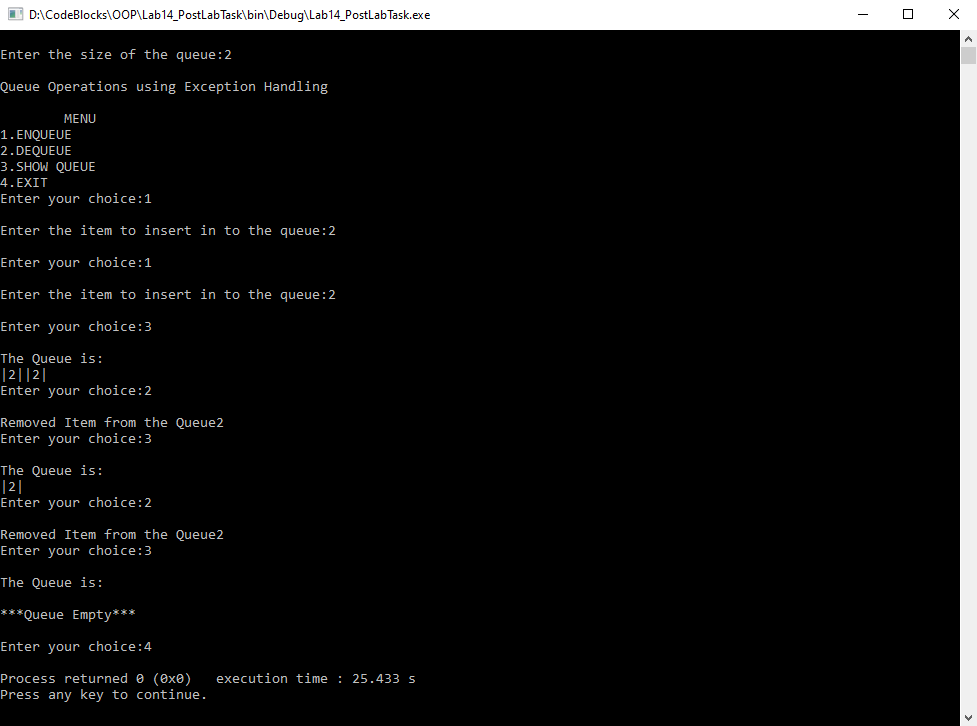
124 }**while**(ch<5);

125

126 **return** 0;

127 }

**Output:**



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

**THE END**