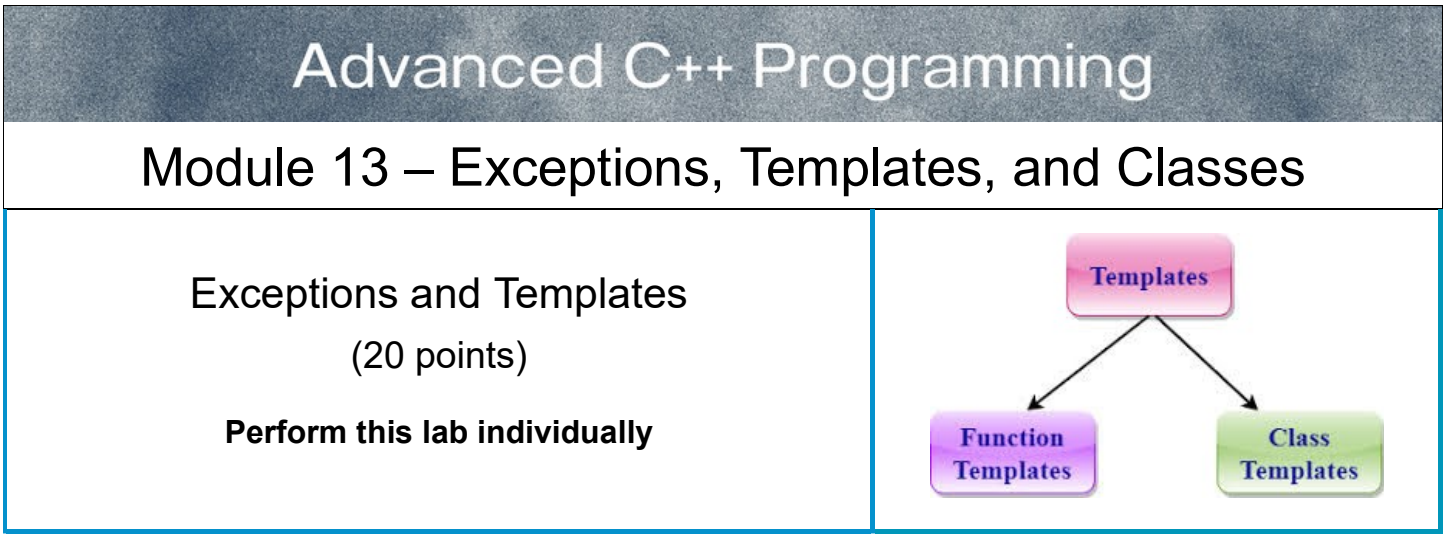
Lab 13 - Exceptions and Templates

Start Assignment

Due Monday by 11:59pm **Points** 20 **Submitting** a file upload



Summary

Determine the output of (slightly modified) Programming Challenges #2 and #4 from Chapter 16 of the textbook.

Please add comments throughout your code.

In both projects, please put all functions after **main** in the program file. This means that you need to place prototypes before **main**.

Programming Challenges (10 points each)

Challenge #2 - Arithmetic Exception

Write a function that accepts an integer parameter and returns its integer square root (if it exists). The function should throw an exception if it is passed an integer that is not a perfect square.

Do not handle the exception in the function, but instead create the exception handler in **main**.

Make the exception a new exception class which your program creates.

Demonstrate the function with a "tester" program that uses a loop to pass to your function the numbers 0 through 25, then prints whether or not the number is a perfect square. (A "perfect square" is a whole number whose square root is also a whole number.) If the number is a perfect

square, then print its square root (returned from the function).

Challenge #4 - Sequence Accumulation

In a separate program, write a template function with the heading

```
T accum(vector <T> v)
```

that forms and returns the "sum" of all items in the **vector v** passed to it. For example, if **T** is a numeric type such as **int** or **double**, the numeric sum will be returned, and if **T** represents the STL string type, then the result of concatenating all of the strings in v is returned.

Note: For any type **T**, the expression **T( )** yields the zero value or an object created by the default constructor for that type. For example, if **T** is a numeric type such as **int** or a **double,** then **T( )** yields 0. If **T** is the string class, then **T( )** yields the empty string. If **T** is another class variable, then **T( )** is the resulting when the default (no-argument) constructor is used. Use this fact to initialize the value to be returned.

In **main**, use your function to "sum" vectors of ints, of doubles, and of strings. In your output, show the contents of each vector and the "sum" of the elements in that vector.

When completed, turn in all program files and screen snips showing the output of your programs.

Links

Additional Files and Programs	Next Lab
none	Lab 14 - Linked Lists (https://miracosta.instructure.com/courses/31330/assignments/842812)

Homework Assignment	Prior Lab
Homework 13 (https://miracosta.instructure.com/courses/31330/assignments/842799)	Lab 12 - Polymorphism, Virtual Functions (https://miracosta.instructure.com/courses/31330/assignments/842810)