Homework 14: Exception Handling

Overview

- You have a working StackType class
- balance.cpp uses the StackType class to decided if the entered statement is well formed
- Well formed means every open bracket (ie {,[,(, and <) has a matching closed bracket (ie },],),>)
- Everything is working (yay!), but it does not have exception handling

Your tasks

Use *try-catch* and *throw* statements.

- Read the documentation in StackType.h and add the appropriate throw statements into StackType.cpp.
- Then add try-catch statements in main to handle the potential errors, all caught errors are considered not well formed (so `balanced = false`).

Reminder:

- a *try* clause encloses the code segment in which an exception might occur.
- *throw* signals the fact that an exception has occurred and control should pass to the exception handler.
- *catch* processes the thrown exception. Each catch clause is an exception handler.

Throw Statement

To throw (or raise) an exception, the programmer uses a *throw* statement, whose syntax is as follows:

```
        throw Expression;
```

Specifically, in this assignment (in StackType.cpp) two exceptions are thrown:

```
        throw FullStack();
```

and

```
        throw EmptyStack();
```

Notice, that these are both "exception classes" which have been defined in StackType.h:

```
class FullStack
// Exception class thrown by Push when stack is full.
{};

class EmptyStack
// Exception class thrown by Pop and Top when stack is emtpy.
{};
```

Make sure that all exceptions are caught. An uncaught exception results in program termination with an error message.

Helpful Hints:

- You are provided with the usual makefile and a test file
  - `make main` will compile the appropriate files
  - `make run` will run the program with keyboard input
  - `make test` will use `test.txt` as input
- When you run `make test` on files as given, it gives an erroneous value for the total number of well formed expressions (it should be 4).
- After you add the appropriate throws in StackType.cpp and run make test, it gives a message like this:

  ```
  terminating due to uncaught exception of type FullStack
  ```

  - This is a hint pointing to where and how to put the try-catch in balance.cpp
- Special Note: The hardest part is probably deciding which statements will be enclosed in the *try clause*. Things to note are:
  - If no exceptions are thrown, then the *catch clauses* are skipped and control goes to the statement following the entire try-catch statement.
  - However, if an exception is thrown by a statement in the *try clause*, control immediately transfers to the appropriate exception handler (*catch clause*). That means that, if there are statements in the *try clause* following the one that caused the exception, they are skipped.

Turn-in Procedure

Submit the files you updated (balanced.cpp and StackType.cpp) on Canvas as attachments.

Adapted from: Department of Computer Science, University of Regina.