University American College Skopje

Course: Object Programming

Exceptions

Exercises

Prepared by: Ognen Spiroski, M.Sc.

Assignment 1

Extend the given class **Fraction**, in the following way:

Modify the double value() function, which returns the value of the fraction (i.e. numerator / denominator), to also *throw* a **DenominatorZeroException** if the denominator has a value of zero.

Modify the friend **Fraction** operator+, operator-, operator*, operator/, to *throw* a **DenominatorZeroException** if the denominator of the resulting **Fraction** has a value of zero

Define the class **DenominatorZeroException** to be derived from **logic_error** (#include <stdexcept>)

Assignment 1

- In the main() function

  Run the previous solution and input zeroes (0) for the denominators to see how the exception handling terminates the program on calls to value()

  - Enclose the value() calls in try / catch blocks
  - Enclose the Fraction operators + , - , * , / in try / catch blocks

Run the new solution

Assignment 2

- Modify the **DenominatorZeroException** class:

  Add a parameterized constructor for the exception which accepts a character array / string error message

Modify the definitions of the **Fraction** operators + , - , * , / to use the new parameterized constructor and send a message identifying which operator is calling it

Use the exception reporting function **what()** to read and report the exceptions' error messages in the main() function

Assignment 3

- Extend the given class **Fraction**, in the following way:
  - Public:
    - A parameterized constructor, taking two integer parameters, setting both *num* and *den* to the values of the respective parameters.

      The constructor *throws* a **DenominatorZeroException** if the denominator has a value of zero

Assignment 4

Define a class **NegativeDiscriminantException** to be derived from **domain_error**

Define a parametrized constructor which accepts an error message

- Extend the given class **Fraction**, in the following way:
  - Modify the definitions of the **Fraction** operators +, -, *, /, to throw a **NegativeDiscriminantException** with a custom message, if the resulting denominator is negative (use parametrized constructor)
  - Include try / catch blocks for the new exception in the main()