Exceptions

"importance, exceptions, exception handlers"

Fundamentals of OOPs

Shakirullah Waseeb
shakir.waseeb@gmail.com

Nangarhar University

December 5, 2017

Agenda

1. Overview
2. Ground Settings
   - Understanding Exceptions
   - Sequence of Events for Exception
3. Exception Syntax
   - Syntax
   - How it works
4. Example
5. Multiple Exceptions
6. Exceptions with members
7. Questions and Discussion

Overview

- **Exceptions:** errors that occur at run time
  a variety of exception circumstances are:
  - running out of memory
  - failed to open a file
  - trying to initialize an object to incompatible value
  - index out-of-bounds
- provide a systematic, object-oriented approach to handling run-time errors generated by C++ classes

Agenda

Understanding Exceptions

- mostly applications creates and interacts with objects of a certain class
- application's calls, to the class member functions cause no problem
- however, sometimes, application makes a mistake, causing an error to be detected in a member function
- given member function, in turn informs the application that a certain error has occurred called (throwing exception)
- in application we insert a separate section of code to handle this error called (exception handler or catch block)
- any code in application that uses objects of the class is enclosed in a try block
- generated errors will be caught in the catch block

Agenda

Sequence of Events for Exception

- Upon rise of an exception the following sequence of events occurs
  - code is executing normally outside a try block
  - control enters the try block
  - a statement in try block causes an error in a **member function**
  - **member function** throws an exception
  - control transfers to the **exception handler** (catch block) following the try block


Agenda

Exception Syntax

Syntax

```
class A{
  public:
    class Error {};
    void Some (){
      if (/* condition */){
        throw Error();
      }
};
int main() {
  try {
    A obj;
    obj.Some();
  }
  catch(A::Error) {
    /* inform user and handle the error */
  }
}
```


Agenda

How it works

- **Exception class**: an exception class (Error in prior slide) must be specified, which's name is used to connect a throw statement with a catch block
- **Throwing Exception**: when application is trying to make a mistake while manipulating an object, subsequently, throw an exception (throw Error() in prior slide)
- **Watch on Exception (try Block)**: statements in main() that might cause given exception (i.e statements that manipulates given object) are enclosed in **try** block
- **Exception Handler (catch Block)**: code that handles the exception given the exception class in parentheses; exception class must specify the class in which it is located, must immediately follow the **try** block


A Stack Example

- A Stack Example on board


Multiple Exceptions

- Explanation on board


Exceptions with members

- Descriptions on board

Next Lecture

The End


Your Turn: Time to hear from you!

1

[1] https://fensafitters.files.wordpress.com/2013/07/3d095.jpg

References

Robert Lafore
Object-Oriented Programming in C++, 4th Edition .
2002.

Piyush Kumar
Object oriented Programming (Using C++)
http://www.compgeom.com/ piyush/teach/3330