Object Oriented and Programming Lab

Lab#12

Table of Contents

To Watch:

1. https://www.youtube.com/watch?v=EyXXLpFriMc

Exceptional handling:

When executing C++ code, different errors can occur coding errors made by the programmer, errors due to wrong input, or other unforeseeable things.

When an error occurs, C++ will normally stop and generate an error message. The technical term for this is: C++ will throw an **exception** (throw an error).

C++ try and catch

Exception handling in C++ consist of three keywords: `try`, `throw` and `catch`:

The `try` statement allows you to define a block of code to be tested for errors while it is being executed.

The `throw` keyword throws an exception when a problem is detected, which lets us create a custom error.

The `catch` statement allows you to define a block of code to be executed, if an error occurs in the try block.

The `try` and `catch` keywords come in pairs:

```
try {
  // Block of code to try
  throw exception; // Throw an exception when a problem arise
}
catch () {
  // Block of code to handle errors
}
```

Consider the following example:

```
  int age = 15;
  if (age > 18) {
    cout << "Access granted - you are old enough.";
  } else {
    throw (age);
```

```
  }
}
catch (int myNum) {
  cout << "Access denied - You must be at least 18 years old.\n";
  cout << "Age is: " << myNum;
}
```

We use the `try` block to test some code: If the `age` variable is less than `18`, we will `throw` an exception, and handle it in our `catch` block.

In the `catch` block, we catch the error and do something about it.
The `catch` statement takes a **parameter**: in our example we use an `int` variable (`myNum`) (because we are throwing an exception of `int` type in the `try` block (`age`)), to output the value of `age`.

If no error occurs (e.g. if `age` is `20` instead of `15`, meaning it will be be greater than 18), the `catch` block is skipped:

You can also use the `throw` keyword to output a reference number, like a custom error number/code for organizing purposes:

```
try {
  int age = 15;
  if (age > 18) {
    cout << "Access granted - you are old enough.";
  } else {
    throw 505;
  }
}
catch (int myNum) {
  cout << "Access denied - You must be at least 18 years old.\n";
  cout << "Error number: " << myNum;
}
```

Output

```
Access denied - You must be at least 18 years old.
Error number: 505
```

Handle Any Type of Exceptions (...)/generalize catch

If you do not know the `throw` **type** used in the `try` block, you can use the "three dots" syntax (`...`) inside the `catch` block, which will handle any type of exception:

```
try {
  int age = 15;
  if (age > 18) {

string  abc="not acceptable age” ;
    cout << "Access granted - you are old enough.";
  } else {
    throw 505;
  }
}

catch (string abc) {
  cout << "Error: " << abc;
}
catch (...) {
  cout << "Access denied - You must be at least 18 years old.\n";
}
```

Reference:

https://www.geeksforgeeks.org/exception-handling-c/

https://www.tutorialspoint.com/cplusplus/cpp_exceptions_handling.htm

http://www.cplusplus.com/doc/tutorial/exceptions/