PLAYBOOK NOTES: Exception Handling in C++

Exceptions are situations when an error occurs.

Example: divide by 0, a file that doesn't pro

Key Words: *try, catch, throw, finally*

A **try-block** will be a block of code that will need to test for situations where an exception may be thrown.

When an error is encountered, an exception is **thrown**.

A **catch-block** will catch an exception that has been thrown. Then a block of code can execute that will process the error.

A **finally block** will always execute. Sometimes used to do something like close files. A finally block is optional.

Example code:

```
int main(void)
{
  int valueA = // Some value
  int valueB = // some value
  try
   {
       // We are about to calculate valueA / valueB
       // If valueB is a 0…. we have a problem!
       if (valueB == 0)
       {
          throw valueB;
       }
   }

   // This will catch exceptions that are of integer type
   catch (int value)
   {
      // Print error message
      printf("Can't divide by 0\n");
   }

   catch(…) // This will catch all exceptions
   {

   }

   finally
   {
      // Print bye-bye message
      printf("Thanks for playing\n")
   }
```

You can also create classes that are used for exception throwing.

Example:

```
class MyApplicationException
{
 public:
   // This is an example of a method you might write.
   void PrintErrorInformatin(void);

 private:

};

// You can fill the class out with whatever methods and data you need


int main(void)
{
  int valueA = // Some value
  int valueB = // some value
  try
   {
       // We are about to calculate valueA / valueB
       // If valueB is a 0…. we have a problem!
       if (valueB == 0)
       {
          throw new MyApplicationException;
       }
   }

   // This catch the above exception
   catch (MyApplicationException * error)
   {
      // Call whatever methods you need to provide feedback on the error
   }
```

YouTube Video shown in class: https://www.youtube.com/watch?v=QqWfw_CFR6Q