**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **26-06-2020** | **Name:** | **Karthik J** |
| **Course:** | C++ Tutorials - Solo Learn | **USN:** | **4AL16EC030** |
| **Topic:** | Files and Exception | **Semester & Section:** | **8TH A** |
| **GitHub Repository:** | Karthik-J |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image Section**   **C++ Files** The fstream library allows us to work with files.  To use the fstream library, include both the standard <iostream> **AND** the <fstream> header file: Example #include <iostream> #include <fstream>  There are three objects included in the fstream library, which are used to create, write or read files:   |  |  | | --- | --- | | **Object/Data Type** | **Description** | | ofstream | Creates and writes to files | | ifstream | Reads from files | | fstream | A combination of ofstream and ifstream: creates, reads, and writes to files |  Create and Write To a File To create a file, use either the ofstream or fstream object, and specify the name of the file.  To write to the file, use the insertion operator (<<). Example #include <iostream> #include <fstream> using namespace std;  int main() {   // Create and open a text file   ofstream MyFile("filename.txt");    // Write to the file   MyFile << "Files can be tricky, but it is fun enough!";    // Close the file   MyFile.close(); } **Read a File** To read from a file, use either the ifstream or fstream object, and the name of the file.  Note that we also use a while loop together with the getline() function (which belongs to the ifstream object) to read the file line by line, and to print the content of the file: Example // Create a text string, which is used to output the text file string myText;  // Read from the text file ifstream MyReadFile("filename.txt");  // Use a while loop together with the getline() function to read the file line by line while (getline (MyReadFile, myText)) {   // Output the text from the file   cout << myText; }  // Close the file MyReadFile.close(); **C++ Exceptions** 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: Example 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: Example try {   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;  } Example explained 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: Example int age = 20;  You can also use the throw keyword to output a reference number, like a custom error number/code for organizing purposes: Example 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;  } **Handle Any Type of Exceptions (...)** 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: Example try {   int age = 15;   if (age > 18) {     cout << "Access granted - you are old enough.";   } else {     throw 505;   } } catch (...) {   cout << "Access denied - You must be at least 18 years old.\n"; } |