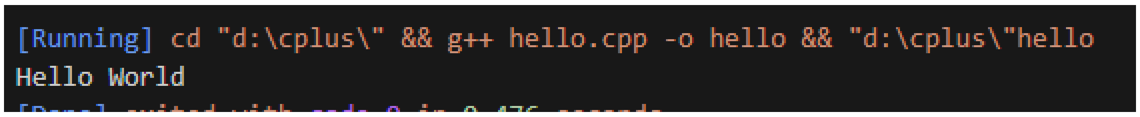
# C++ notes:-

**hello.cpp :-**

```cpp
// C++ program to display "Hello World"

// Header file for input output functions
#include <iostream>
using namespace std;

// Main() function: where the execution of program begins
int main()
{
    // prints hello world
    cout << "Hello World";

    return 0;
}
```

**Output:-**

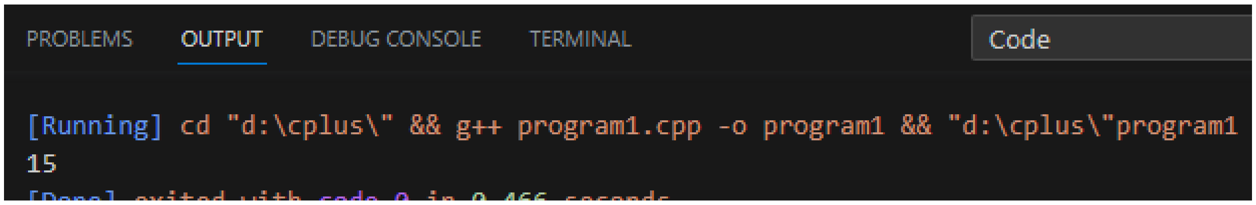


```
[Running] cd "d:\cplus\" && g++ hello.cpp -o hello && "d:\cplus\"hello
Hello World
[Done] exited with code 0 in 0.476 seconds
```

**program1.cpp (number example) :-**

```cpp
#include <iostream>
using namespace std;

int main() {
  int myNum = 15;
  cout << myNum;
  return 0;
}
```

**Output:-**



```
[Running] cd "d:\cplus\" && g++ program1.cpp -o program1 && "d:\cplus\"program1
15
[Done] exited with code=0 in 0.466 seconds
```

**program2.cpp (user input example):-**

```cpp
#include <iostream>
using namespace std;

int main() {
  int x;
  cout << "Type a number: "; // Type a number and press enter
  cin >> x; // Get user input from the keyboard
  cout << "Your number is: " << x;
  return 0;
}
```

# Create and Write To a File

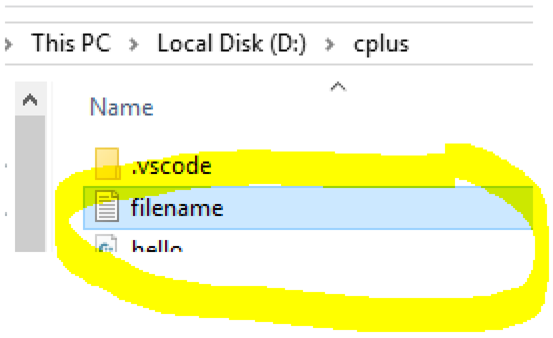
To create a file, use either the `ofstream` or `fstream` class, and specify the name of the file.

To write to the file, use the insertion operator (`<<`).

**Program3.cpp (create a file ):-**

```cpp
#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();
}
```

**Output:-**



## Read a File

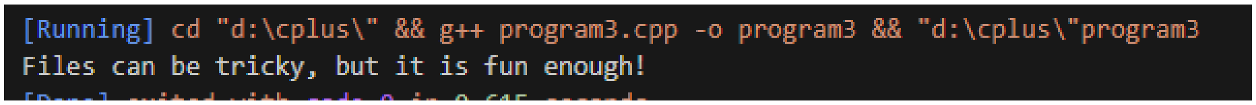
To read from a file, use either the `ifstream` or `fstream` class, and the name of the file.

Note that we also use a `while` loop together with the `getline()` function (which belongs to the `ifstream` class) to read the file line by line, and to print the content of the file:

**program3.py file:-**

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;

int main () {
  // Create a text file
  ofstream MyWriteFile("filename.txt");

  // Write to the file
  MyWriteFile << "Files can be tricky, but it is fun enough!";

  // Close the file
  MyWriteFile.close();

  // 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();
}
```

**Output:-**



```
[Running] cd "d:\cplus\" && g++ program3.cpp -o program3 && "d:\cplus\"program3
Files can be tricky, but it is fun enough!
[Done] exited with code=0 in 0.645 seconds
```

# 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

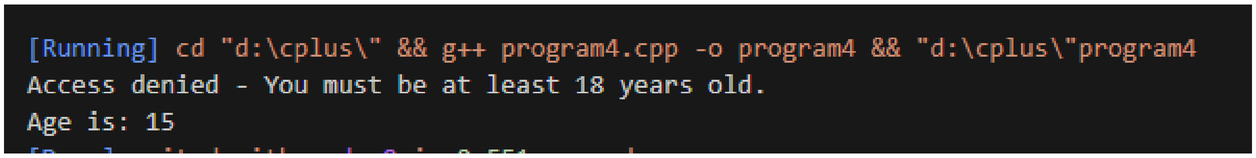
```
try {
  // Block of code to try
  throw exception; // Throw an exception when a problem arise
}
catch () {
  // Block of code to handle errors
}
```

Consider the following example:-

**program4.cpp:-**

```cpp
#include <iostream>
using namespace std;

int main() {
  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;
  }
  return 0;
}
```

**Output:-**



```
[Running] cd "d:\cplus\" && g++ program4.cpp -o program4 && "d:\cplus\"program4
Access denied - You must be at least 18 years old.
Age is: 15
```

## *Example explained*

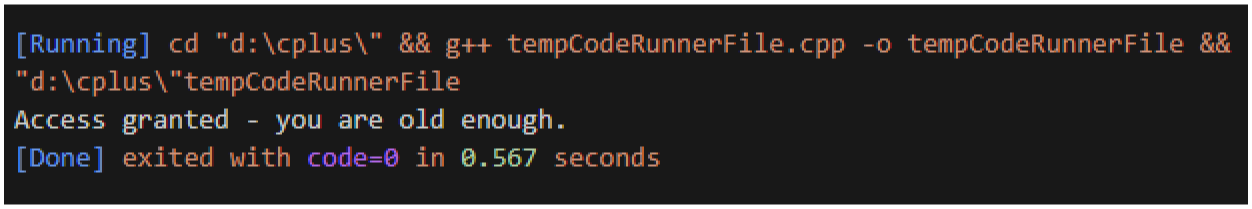
We use the try block to test some code: If the age variable is less than 18, we will throw an exception, and handle it in our catch block.

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.

```
#include <iostream>
using namespace std;

int main() {
  try {
    int age = 20;
    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;
  }
  return 0;
}
```

**output:-**



```
[Running] cd "d:\cplus\" && g++ tempCodeRunnerFile.cpp -o tempCodeRunnerFile &&
"d:\cplus\"tempCodeRunnerFile
Access granted - you are old enough.
[Done] exited with code=0 in 0.567 seconds
```