Lab 10 Notes

Hayden Whiteford

104001272

Part A: Binary file Output / Input

Code:

```
#include <iostream>

class CompoundType {
public:
    char Var1 = 'A';
    int Var2 = 1;
    float Var3 = 3.14;
};

CompoundType* test = new CompoundType();

void printVar(CompoundType* aClass){
    std::cout << aClass->Var1 << "\n";
    std::cout << aClass->Var2 << "\n";
    std::cout << aClass->Var3 << "\n";
}

int main(int argc, const char * argv[]) {
    // insert code here...
    CompoundType* test = new CompoundType();
    printVar(test);
    return 0;
}
```

Q5.
There are a few open modes:
- std::iOS::in - open the file for reading inputs
- std::iOS::out - open the file for writing
- std::iOS::binary - treat the file as binary not text
- std::iOS::app - open in “append” mode (perform all output operations at the end of the file)
- std::iOS::ate - open the file and seek to the end
- std::iOS::trunc - truncate the file if it already exists when opened using std::iOS::out

Q8.
The file size is 9 bytes - although this doesn’t make immediate sense based on what I inputted - from my research I got 13 bytes (1 + 4 + 8)

Part B: Simple Text File Input with Split

```
#include <iostream>
#include <fstream>
#include <sstream>
```

```
int main(int argc, const char * argv[]) {
    // insert code here...
    std::ifstream inputFile("test2.txt", std::ios::in);
    std::string myText;
    while (getline (inputFile, myText)) {
      // Output the text from the file
        if (myText[0] != '\n' && myText[0] != '#'){

            //split the last line up and print

            std::stringstream ss(myText);
            std::string part;

            while (getline(ss, part, ':')) {
            // Print each part of the split line
                std::cout << "Bit: " << part << std::endl;
            }
        }

    }

    return 0;
}
```

Part C: Reading JSON Files

```
#include <iostream>

#include <fstream>
#include "json.hpp"

using json = nlohmann::json;

int main(int argc, const char * argv[]) {

    //open the file for reading
    std::ifstream inputFile("test3.json");

    //parse the json data?
    json jsonData;
    inputFile >> jsonData;

    //close it again - we put the data into the json data file
    inputFile.close();

    //we use the & symbol here so we get the values by reference
instead of making copies
    for (auto& entry : jsonData.items()) {
        std::string key = entry.key();        // Get the key
        json value = entry.value();           // Get the value
```

```
        // Print the key and value
        std::cout << "Key: " << key << ", Value: " << value << 
std::endl;
        }


    return 0;
}
```