Coding workshop: user I/O menu

In this worksheet, we are going to build a simple menu system which takes a numerical input from a user then prints out different messages accordingly.

Getting started

You will either be starting in a new Visual Studio project, a new XCode project or the web-based Visual Studio Code project.

Web-based Visual Studio Code

Create and open a file called main.cpp. Put this code in there:

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

You will put the code from this worksheet into main.cpp. To build and run your program (once you've put some code in it), select terminal from the View menu, or if it is not there, select New Terminal from the Terminal menu. Type this in the terminal to build the program:

```
g++ main.cpp -o merkelrex
```

Then run the program with:

```
./merkelrex
```

Visual Studio Windows

You need to create a new project in Visual Studio. The project should be a C++ console application. Once your new project opens in Visual Studio, you should see there is a single cpp file in the file explorer, with the same name as your project. You should write the code in the worksheet in this file. To run your project, choose Run without debugging (ctrl-F5) from the Debug menu in Visual Studio. This should pop up a console window showing the output of the program. You might want to switch the target from x86 to x64. There should be a dropdown at the top of the code editor that lets you change that.

XCode

You need to create a new project in XCode. The project should be a macOS command-line tool. Make sure you select C++ as the language in the step where you specify the project name. You can run the program by clicking on the play button at the top left of the XCode window.

Print out a menu

The main function

You may already have a main function in your cpp file if you are in Visual Studio or XCode. IT should look something like this:

```
#include <iostream>

int main()
{

}
```

The starter program created by your IDE may have a more elaborate main function, like this:

```
#include <iostream>

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

}
```

That allows you to process command-line arguments sent into your program. We'll not be needing those, for now, so the plain main function is fine for now.

Actually print the menu

You can print out menu options like this:

```
#include <iostream>

int main ()
{
    std::cout << "1: Print help" << std::endl;
}
```

Go ahead and add six print outs for the following choices:

- 1: Print help
- 2: Print exchange stats
- 3: Place an ask
- 4: Place a bid
- 5: Print wallet
- 6: Continue

Read user input

Now we are going to prompt the user to select a choice from the menu. Add this code below your menu printing code:

```
int userOption;
std::cout << "Type in 1-6" << std::endl;
std::cin >> userOption;
```

That declares a variable called userOption that can store an integer. Then it will read from std::cin to the choice variable.

Can you add a line that prints our which choice they made?

For more information on std::cin, check out the textbook, Chapter 2 p38 where there is a detailed example.

Do something based on the user input

Now add some if statements after you have read in the user input. Here is one to process the help option:

```
if (userOption == 1)
{
   std::cout << "Help - choose options from the menu" << std::endl;
   std::cout << "and follow the on screen instructions." << std::endl;
}
```

Add five more, one for each of the menu options. Also, add one that checks if they entered a number below 1 or above 6 and prints out a warning if they did. Can you do this with a switch statement instead?

Wrap it in a loop

Now wrap the 'print menu, read user input, respond to user input' code in a while loop:

```
while(true)
{
    std::cout << "1: Print help " << std::endl;
    // ... more options here
    int userOption;
    std::cout << "Type in 1-6" << std::endl;
    std::cin >> userOption;
    if (userOption == 1)
    {
       std::cout << "Help - choose options from the menu" << std::endl;
       std::cout << "and follow the on screen instructions." << std::endl;
    }
    // more if statements for the other options here
}
```

This program will not end. To exit from the program in Visual Studio Code, use ctrl-c. In Xcode, you can click the stop button, which should be where you clicked the play button to run the program. In Visual Studio, you should be able to close the terminal that popped up.

You can find more details about while loops in the textbook, Chapter 5 p138.

Conclusion

You should now have a functioning menu system where the user can repeatedly choose options from the menu, and different messages are printed out according to their choices.