Coding workshop: Looking at the data and modelling it

In this worksheet, we are going to familiarise ourselves with the order book data set and to look at how we can model the data using basic data types.

The order book data set

Attached with this reading is a CSV file which is our trading dataset. Download the CSV file and open it in a text editor, to see the raw data, or in a spreadsheet program, to see the data in a more easily manipulated form.

Think about what kind of data each of the columns contains. What range do the values fall into?

Numerical types in C++

C++ comes with a set of data types which allow us to store basic data.

Check out the following link: https://www.cplusplus.com/doc/tutorial/variables/ . Scan down to the table in the section 'Fundamental data types' and you will see the range of available basic data types.

Let's say you want to create an unsigned int, which is an integer value which is positive or zero. There are two ways to initialise such a variable:

```
unsigned int aNumber {10};
```

and:

```
unsigned int aNumber = 10;
```

Both are equally valid, but the first one uses C++11 style braces. We'll see that style again later when we start creating objects. Sometimes you will see just 'int' without the specifier of signed or unsigned. If you use just int, what do you think happens? Do an experiment where you create an int:

```
int x{-5};
std::cout << x << std::end;
```

If it is unsigned, it should not be able to store that negative number. What happens? Is it signed or unsigned?

Referring to the table with the list of data types from the link above, can you select appropriate data types of the numerical columns in our trading dataset? Can you then write out a set of variable declarations and assignments to cover these numerical columns in the dataset? Here is an example:

```
#include <iostream>

int main()
```

```
{
    double amount = 4.5;
}
```

Which form of variable declaration is that, of the two seen above? Can you write your variable declarations using the C++11 style curly braces?

String data types

Some of the columns contain string data. C++ has a data type called std::string which is suitable for storing string data. This link describes various ways in which you can create and initialise strings:

https://www.cplusplus.com/reference/string/string/string/.

Let's declare some std::strings to store the data items in the CSV file. First off, include the string header:

```
#include <string>
```

Then in the main function, add some string declarations:

```
std::string product = "ETH/BTC";
```

Or better, in the C++11 style:

```
std::string product{"ETH/BTC"};
```

Write variable declarations for the other string types.

Categorical data types - enum class

One of the columns in the dataset only ever has one of two values, the order type. It can be "ask" or "bid". For this, we are going to define a new type using the enum and class keywords.

At this link, you can find a detailed description of how enums and enum class types work:

https://en.cppreference.com/w/cpp/language/enum

You do not need to read all of that, but it might useful if you do not fully grasp the following steps.

They use the example of colours. We are using the example of order types. First, we define our new type.

```
enum class OrderBookType {bid, ask};
```

Next, we can create a variable of this type:

```
OrderBookType orderType = OrderBookType::ask;
```

Trying setting the variable to different values of OrderBookType and printing them out. What values are used to actually represent these orderTypes?

Storing the data for multiple orders in a vector

Now let's see how we can store multiple orders in a vector.

Add this to the top of your cpp file, so you can use the std::vector class:

```
#include <vector>
```

Now in the main function, add several vectors, one for each column in the dataset. Here is a simple example program showing how to store one column in one vector:

```
#include <iostream>
#include <vector>
#include <string>

int main()
{
    std::vector<double> amounts;
    amounts.push_back(0.5);
    amounts.push_back(0.125);

    std::cout << "First row amount " << amounts[0] << std::endl;
    std::cout << "Second row amount " << amounts[1] << std::endl;

}
```

If you need a further reference about std::vector, here is a link:

https://www.cplusplus.com/reference/vector/vector/.

Can you adapt the above program so it has five vectors, one for each column, each using an appropriate data type. Make sure you add your enum class definition so you can store the asks and bids.

Note that we are not reading the real data file in yet - we are just hard coding the values of the columns for now.

Write statements that print out a complete row of information.

Conclusion

In this worksheet, we have experimented with different data types, including numerical types, std::string and enum class. With these data types, we can represent all the data in the order book data set. We have also used std::vectors to store multiple rows of data, one vector per column.