ECM2433- Coursework 2 report

I designed my system to take read each line from the input file and interpret them as an object of the appropriate class: customer, sale, or end of day, these are then stored in an appropriate object vector and the relevant output message is sent to the standard output stream.

This is done by first using an ifstream object (an object used to read files) from the imported fstream header file to open the file given as the second argument on the command line which is then checked to see if it correctly opened.

Each line is traversed using a while loop with getline taking the input file and a string “recordInfo” as arguments. The first character of is then checked and compared to C, S and E, representing the first character of the line in the input files which designates the line as a customer, sale, or end of day.

If the first character is a C the information needed is gathered from the string form of the line using the substr function from the string header file and the stoi (string to int) function, a customerRecord object is then created and the values are assigned to it before the object is pushed to the back of the customerVector vector. The relevant order processing message is then sent to the standard output stream and the program returns to the loop.

If the first character is an S the information needed is gathered from the string form of the line using the substr function from the string header file and the stoi (string to int) function, a salesRecord object is then created and the values are assigned to it before the object is pushed to the back of the salesVector vector. The char “salesRecordExpress” is then checked to see if it equal to N, meaning a normal order or X meaning an express order. If it is a normal order the order processing message is sent to the standard output stream and the relevant customers current order quantity is updated. If it is an express order the order processing message is sent to the standard output stream and the relevant customers current order quantity is updated, an invoice message is sent to the standard output stream, the invoice number is incremented and the customer’s order quantity is set to 0. If the char is anything else, then an error message is given explaining that the format of the file is invalid.

If the first character is an E the date is gathered from the string form of the line using the substr function from the string header file and the stoi (string to int) function, an eodRecord object is then created and the values are assigned to it before the object is pushed to the back of the eodVector vector. An end of day message with the date is then sent to the standard output stream before iterating through each object in the customerVector. If the customerRecord’s quantityOrdered variable is not equal to 0, indicating that the customer has outstanding orders that day, a shipping message is sent to the standard output stream, along with an invoice message, the invoice number is then incremented and the quantityOrdered variable of the customer is reset to 0.

If the first character is anything else, then an error message is sent to the error message stream informing the user that the format of the file is invalid.

In my system I used classes as a method of storing the necessary information that would need to be accessed later by different parts of the system, for this same reason I also used vectors as particularly the customerVector vector provided an easy way of storing the customerRecord objects to be easily accessed by the sale and end of day processing parts of the system as the position of the object in the vector would be equal to its customerNo – 1.

I made assumptions that the user would input the correct type of data, for example where they were supposed to put in dates or quantities they would only put in numbers so the system could convert it to int form.