**STL Containers**   
  
Workshop 6

In this workshop, you store a set of polymorphic objects in an STL container.

**LEARNING OUTCOMES**

Upon successful completion of this workshop, you will have demonstrated the abilities to

* store and manage polymorphic objects using an STL vector
* store a set of uniform rates for all instances of a class using a class array
* report and handle an exception
* reflect on what you have learned in this workshop

**SPECIFICATIONS**

This workshop consists of three modules:

* **w7** - the main program receives the name of the file that holds information about the products sold to a customer
* **Sale** - contains the list of products sold to the customer
* **iProduct** hierachy - each object describes a single product sold to the customer

The **main** program receives the a user-specified file.  The records hold price information about products, some of which are taxable.  Taxable and non-taxable products belong to the same hierarchy; the taxable product class derives from the simpler product class.

**File Records**

Each record in the file consists of two or three fields: a product number field, a product cost field and an optional taxable code field.  The cost field contains the price before any tax.  The contents of the [test file](https://scs.senecac.on.ca/~oop345/pages/workshops/Sales.dat) for this workshop are:

|  |
| --- |
| **10012 34.56**  **10023 45.67 H**  **10234 12.32 P**  **10056 67.54**  **10029 54.12 H**  **10034 96.30** |

* **H** identifies a taxable product at the HST rate
* **P** identifies a taxable product at the PST rate

A non-taxable record terminates with a newline character immediately following the price field.  A record with a taxable code other than the two listed above is an invalid record.  A file with an invalid record is a corrupted file.

**iProduct Hierarchy**

The **iProduct** hierarchy holds information about the different products purchased by a customer.  The interface to this hierarchy is:

|  |
| --- |
| **#ifndef \_I\_PRODUCT\_H\_**  **#define \_I\_PRODUCT\_H\_**  **// Workshop 7 - STL Containers**  **// iProduct.h**  **#include <iostream>**  **#include <fstream>**  **namespace w7 {**  **class iProduct {**  **public:**  **virtual double getCharge() const = 0;**  **virtual void display(std::ostream&) const = 0;**  **};**  **std::ostream& operator<<(std::ostream&, const iProduct&);**  **iProduct\* readProduct(std::ifstream&);**  **}**  **#endif** |

The **double getCharge() const** query returns the price chargable to the customer.

The **void display(std::ostream& os) const** query receives a reference to the output stream and inserts the product information in the form of a single line into the output stream.

The overloaded insertion operator (**<<**) receives a reference to the output stream and displays the product information in the form of a single line on the output stream.

The **readProduct** function receives a reference to the input file stream, reads the next record from the stream, allocates memory for the corresponding **iProduct** object, copies the record information into that object and returns its address to the caller function.

**Implementations**

Complete the definition of the polymorphic **iProduct** object by implementing two distinct classes:

* **Product** - holds information about a product that consists of its product number and its cost
* **TaxableProduct** - derives from **Product** and holds additional information about the product's taxable status

The same tax rates apply to all **TaxableProduct** objects.  Use a class array (an array of class variables) to store the tax rates for HST and PST tax codes (13% and 8%).

**Sale Class**

Code a class named **Sale** that holds information about the set of products sold to a customer.  Upon instantiation, a **Sale** object receives the address of a C-style null-terminated string that holds the name of a file.  This file contains the list of products sold to the customer.  The one-argument constructor copies the product information into an STL container.

Since the products are polymorphic objects, your class stores and accesses them through a set of pointers to the objects.

Your class design includes the following member function:

* **void display(std::ostream& os) const** - receives a reference to the output object and displays the sale information as shown in the sample output above.  The output includes column headings and the total sale amount including taxes paid.  The product number and cost occupy fields of width 10.

**Main Program**

Complete main program listed below.  The complete coding reports any exception thrown by the **Sale** object or any **iProduct** object object.  If an exception is thrown, the main program returns an error code of 2.

|  |
| --- |
| **// Workshop 7 - STL Containers**  **// w7.cpp**  **#include <iostream>**  **#include <string>**  **#include "Sale.h"**  **int main(int argc, char\*\* argv) {**  **if (argc != 2) {**  **std::cerr << argv[0] << ": incorrect number of arguments\n";**  **std::cerr << "Usage: " << argv[0] << " file\_name\n";**  **return 1;**  **}**  **w7::Sale sale(argv[1]);**  **sale.display(std::cout);**  **std::cout << "\nPress any key to continue ... ";**  **std::cin.get();**  **}** |

The output from your completed main program for the file contents listed above should like:

|  |
| --- |
| **Product No Cost Taxable**  **10012 34.56**  **10023 45.67 HST**  **10234 12.32 PST**  **10056 67.54**  **10029 54.12 HST**  **10034 96.30**  **Total 324.47**  **Press any key to continue ...** |

**SUBMISSION**

Follow your professor's submission instructions