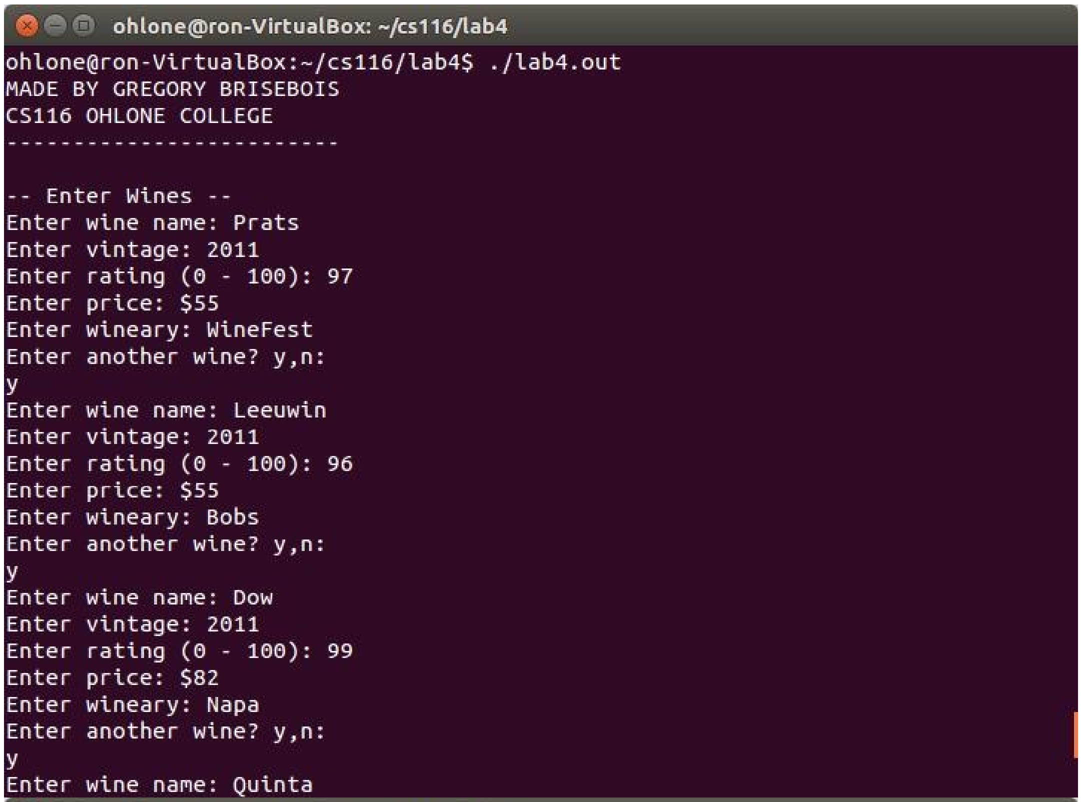
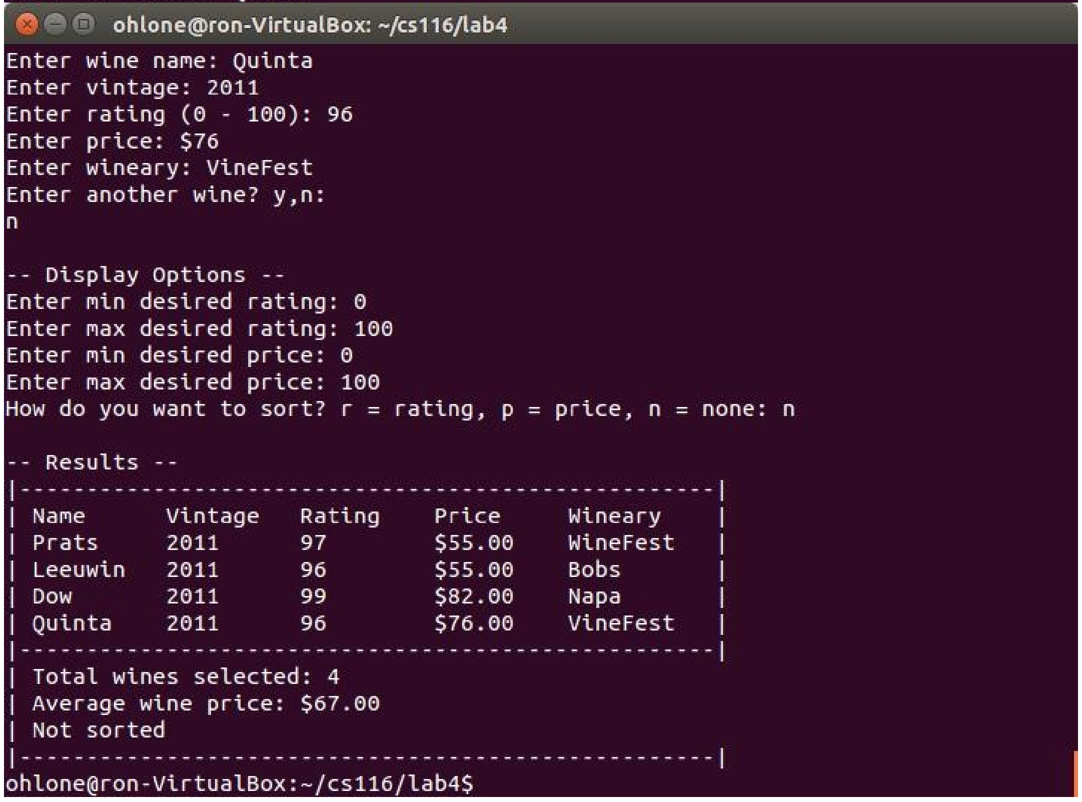
Gregory Brisebois, CS-116 Lab 4

- Overview
The user can enter wines, and parameters to display them
(Output is too big to capture in one terminal window)



```
ohlone@ron-VirtualBox:~/cs116/lab4$ ./lab4.out
MADE BY GREGORY BRISEBOIS
CS116 OHLONE COLLEGE
-------------------------

-- Enter Wines --
Enter wine name: Prats
Enter vintage: 2011
Enter rating (0 - 100): 97
Enter price: $55
Enter wineary: WineFest
Enter another wine? y,n:
y
Enter wine name: Leeuwin
Enter vintage: 2011
Enter rating (0 - 100): 96
Enter price: $55
Enter wineary: Bobs
Enter another wine? y,n:
y
Enter wine name: Dow
Enter vintage: 2011
Enter rating (0 - 100): 99
Enter price: $82
Enter wineary: Napa
Enter another wine? y,n:
y
Enter wine name: Quinta
```



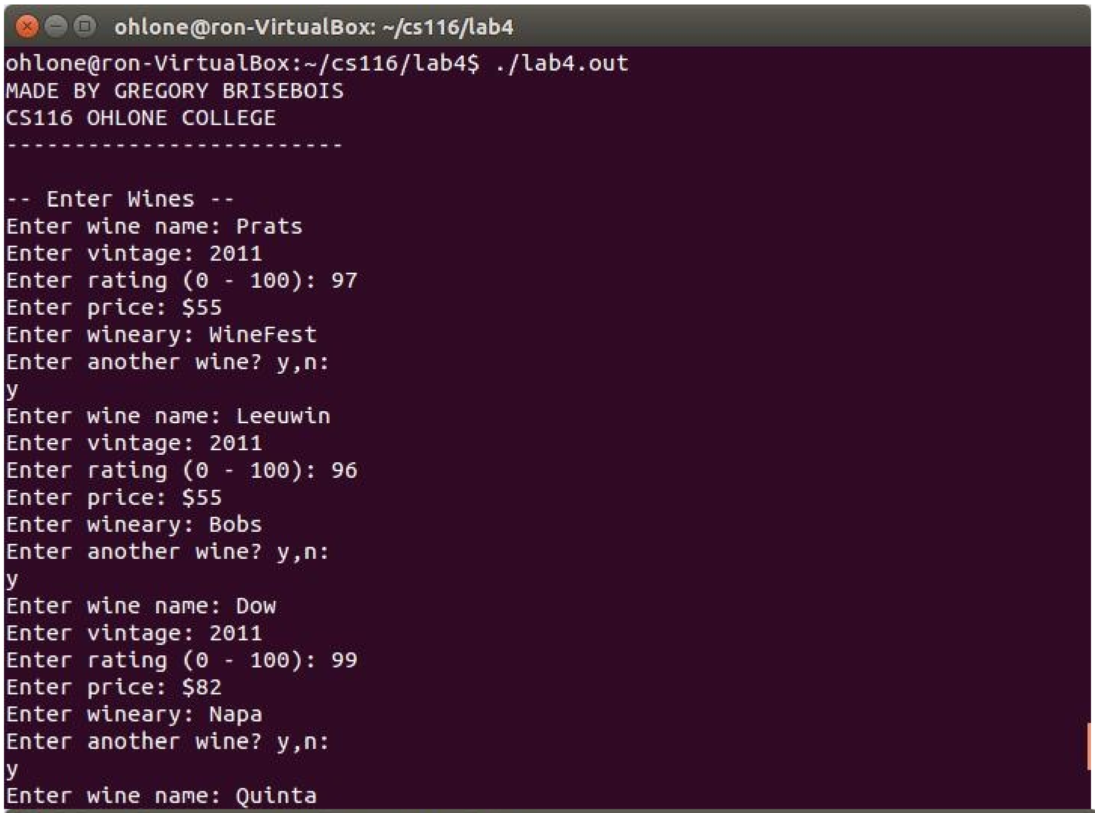
```
Enter wine name: Quinta
Enter vintage: 2011
Enter rating (0 - 100): 96
Enter price: $76
Enter wineary: VineFest
Enter another wine? y,n:
n

-- Display Options --
Enter min desired rating: 0
Enter max desired rating: 100
Enter min desired price: 0
Enter max desired price: 100
How do you want to sort? r = rating, p = price, n = none: n

-- Results --
|----------------------------------------------------|
| Name      Vintage   Rating    Price     Wineary    |
| Prats     2011      97        $55.00    WineFest   |
| Leeuwin   2011      96        $55.00    Bobs       |
| Dow       2011      99        $82.00    Napa       |
| Quinta    2011      96        $76.00    VineFest   |
|----------------------------------------------------|
| Total wines selected: 4
| Average wine price: $67.00
| Not sorted
|----------------------------------------------------|
ohlone@ron-VirtualBox:~/cs116/lab4$
```

- Rating range & sort by price

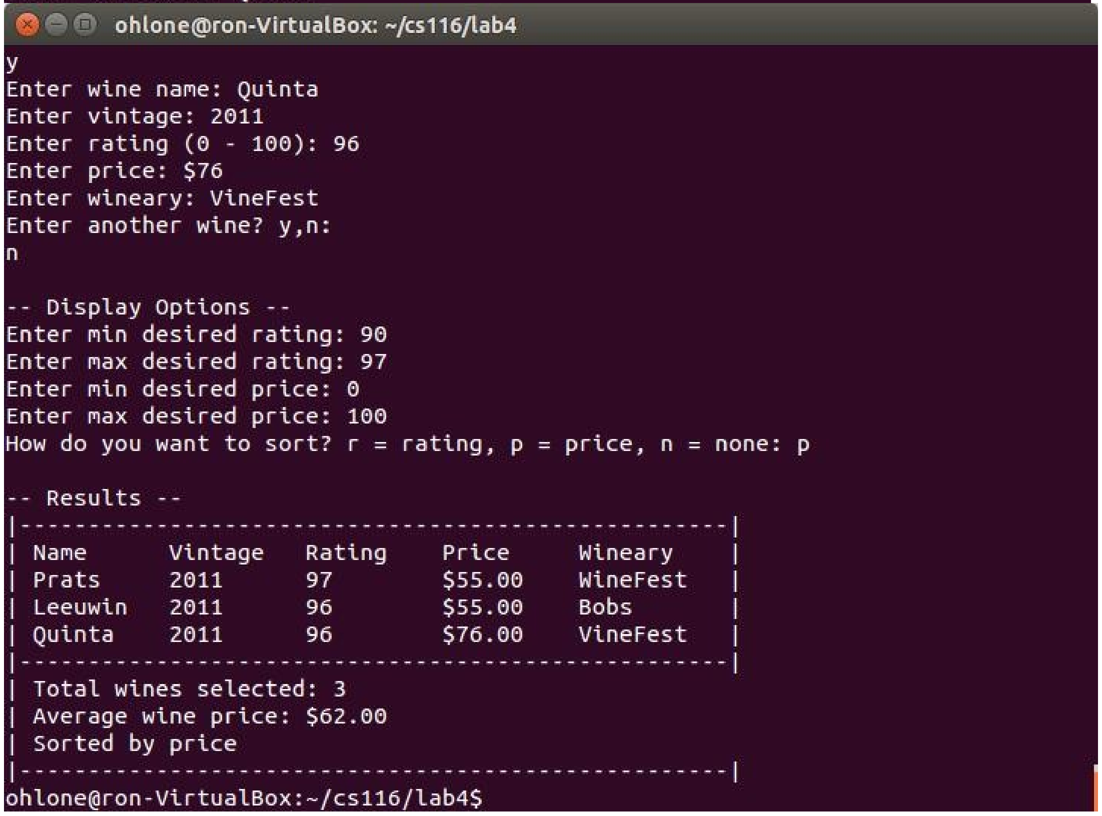
Only show wines between 90 and 97 rating, and sort by price (low-hi)

Only 3 wines end up being shown, even though 4 are entered

Since the user entered “p”, it is sorted by price

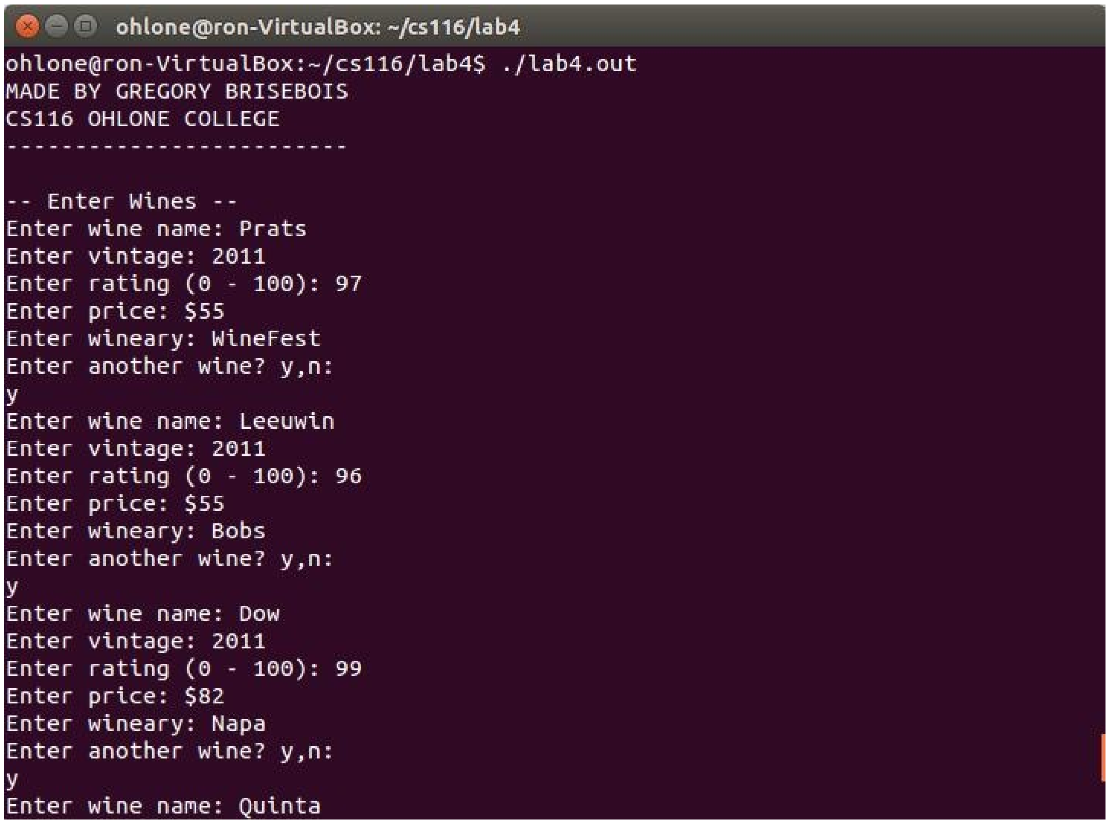


```
ohlone@ron-VirtualBox:~/cs116/lab4$ ./lab4.out
MADE BY GREGORY BRISEBOIS
CS116 OHLONE COLLEGE
-------------------------

-- Enter Wines --
Enter wine name: Prats
Enter vintage: 2011
Enter rating (0 - 100): 97
Enter price: $55
Enter winery: WineFest
Enter another wine? y,n:
y
Enter wine name: Leeuwin
Enter vintage: 2011
Enter rating (0 - 100): 96
Enter price: $55
Enter winery: Bobs
Enter another wine? y,n:
y
Enter wine name: Dow
Enter vintage: 2011
Enter rating (0 - 100): 99
Enter price: $82
Enter winery: Napa
Enter another wine? y,n:
y
Enter wine name: Quinta
```

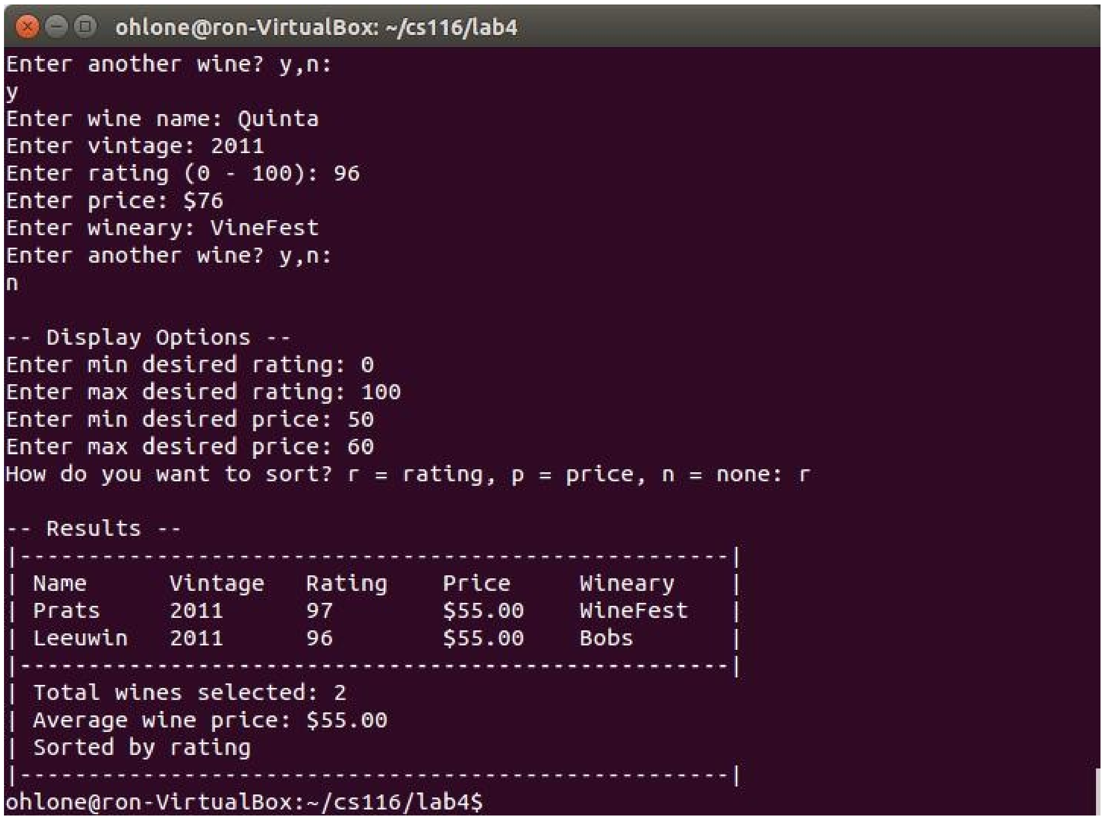


```
y
Enter wine name: Quinta
Enter vintage: 2011
Enter rating (0 - 100): 96
Enter price: $76
Enter winery: VineFest
Enter another wine? y,n:
n

-- Display Options --
Enter min desired rating: 90
Enter max desired rating: 97
Enter min desired price: 0
Enter max desired price: 100
How do you want to sort? r = rating, p = price, n = none: p

-- Results --
|-----------------------------------------------------|
| Name      Vintage   Rating    Price     Winery      |
| Prats     2011      97        $55.00    WineFest    |
| Leeuwin   2011      96        $55.00    Bobs        |
| Quinta    2011      96        $76.00    VineFest    |
|-----------------------------------------------------|
| Total wines selected: 3
| Average wine price: $62.00
| Sorted by price
|-----------------------------------------------------|
ohlone@ron-VirtualBox:~/cs116/lab4$
```

- Price range & sort by rating
The opposite of the last one: show only within a certain price and sort by rating.
Only 2 wines make the $50-60 cut, and they are in order of rating (high-low)



```
ohlone@ron-VirtualBox: ~/cs116/lab4
ohlone@ron-VirtualBox:~/cs116/lab4$ ./lab4.out
MADE BY GREGORY BRISEBOIS
CS116 OHLONE COLLEGE
-------------------------

-- Enter Wines --
Enter wine name: Prats
Enter vintage: 2011
Enter rating (0 - 100): 97
Enter price: $55
Enter winery: WineFest
Enter another wine? y,n:
y
Enter wine name: Leeuwin
Enter vintage: 2011
Enter rating (0 - 100): 96
Enter price: $55
Enter winery: Bobs
Enter another wine? y,n:
y
Enter wine name: Dow
Enter vintage: 2011
Enter rating (0 - 100): 99
Enter price: $82
Enter winery: Napa
Enter another wine? y,n:
y
Enter wine name: Quinta
```



```
ohlone@ron-VirtualBox: ~/cs116/lab4
Enter another wine? y,n:
y
Enter wine name: Quinta
Enter vintage: 2011
Enter rating (0 - 100): 96
Enter price: $76
Enter winery: VineFest
Enter another wine? y,n:
n

-- Display Options --
Enter min desired rating: 0
Enter max desired rating: 100
Enter min desired price: 50
Enter max desired price: 60
How do you want to sort? r = rating, p = price, n = none: r

-- Results --
|--------------------------------------------------|
| Name      Vintage   Rating    Price     Winery    |
| Prats     2011      97        $55.00    WineFest  |
| Leeuwin   2011      96        $55.00    Bobs      |
|--------------------------------------------------|
| Total wines selected: 2
| Average wine price: $55.00
| Sorted by rating
|--------------------------------------------------|
ohlone@ron-VirtualBox:~/cs116/lab4$
```

Source Code:

lab4.cpp (main)

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

#include "wine.h"

using namespace std;

// Course requirement - prints my own name on the console
void printMyName()
{
	cout << "MADE BY GREGORY BRISEBOIS" << endl;
	cout << "CS116 OHLONE COLLEGE" << endl;
	cout << "-------------------------" << endl;
}

// Prompts the user to enter 1 wine
// @return the created wine class
Wine getWineInput()
{
	string newName = "";
	int newVintage = 0;
	int newRating = 0;
	double newPrice = 0;
	string newWineary = "";

	cout << "Enter wine name: ";
	cin >> newName;

	cout << "Enter vintage: ";
	cin >> newVintage;

	cout << "Enter rating (0 - 100): ";
	cin >> newRating;

	cout << "Enter price: $";
	cin >> newPrice;

	cout << "Enter wineary: ";
	cin >> newWineary;

	Wine newWine(newName, newVintage, newRating, newPrice, newWineary);

	return newWine;
}

// Sort functions used by the vector to sort itself
bool sortByPrice(Wine left, Wine right)
{
	return left.getPrice() < right.getPrice();
}
bool sortByRating(Wine left, Wine right)
{
	return left.getRating() > right.getRating();
}

// Prints a list of all wines, with the options to sort or display only those in a certain
```

```
range
// @param wines - the vector of wines to display
// @param ratingMin - the minimum rating to include in the display
// @param ratingMax - max rating to include
// @param minPrice, maxPrice - min and max prices to include
void printAllWines(vector<Wine> wines, int ratingMin = 0, int ratingMax = 99, int priceMin =
0, int priceMax = 100, char sortId = 0)
{
	// String that will tell the user how we sorted
	string sortTypeDisplay = "Not sorted";

	// Sort (or not) based on the input
	switch(sortId)
	{
		case 'r':
			sort(wines.begin(), wines.end(), sortByRating);
			sortTypeDisplay = "Sorted by rating";
			break;

		case 'p':
			sort(wines.begin(), wines.end(), sortByPrice);
			sortTypeDisplay = "Sorted by price";
			break;
	}

	// Variables that control the width of the output columns
	int nameWidth = 10;
	int vintageWidth = 10;
	int ratingWidth = 10;
	int priceWidth = 10;
	int winearyWidth = 10;
	int totalWidth = nameWidth + vintageWidth + ratingWidth + priceWidth + winearyWidth;

	// Make a bar to use for the ends (formatting)
	string bar = "|";
	bar.append(totalWidth + 2, '-');
	bar.append("|");

	// Print column labels
	cout << bar << endl;
	cout << "| ";
	cout << setw(nameWidth) << left << "Name"
	     << setw(vintageWidth) << "Vintage"
	     << setw(ratingWidth) << "Rating"
	     << setw(priceWidth) << "Price"
	     << setw(winearyWidth) << "Wineary"
	     << " |" << endl;

	// This will keep track of how many wines make the cut
	int numSelectedWines = 0;

	// These will keep track of selected prices
	double totalPrice = 0;
	double avgPrice = 0;

	// Print the wine, if it makes the cut
	for(int i = 0; i < wines.size(); i++)
	{
		// IF rating and price are within set bounds
		if(wines[i].getRating() <= ratingMax && wines[i].getRating() >= ratingMin
		&& wines[i].getPrice() <= priceMax && wines[i].getPrice() >= priceMin)
		{
			// Print out the row for the wine
			cout << "| ";
```

```
                    cout << setw(nameWidth) << left << wines[i].getName()
                         << setw(vintageWidth) << wines[i].getVintage()
                         << setw(ratingWidth) << wines[i].getRating()
                         << "$" << setw(priceWidth-1) << fixed << setprecision(2) <<
wines[i].getPrice() // We have to subtract width by 1 to make space for the dollar sign
                         << setw(winearyWidth) << wines[i].getWineary()
                         << " |" << endl;

                    // Update our selected wine statistics
                    numSelectedWines++;
                    totalPrice += wines[i].getPrice();
              }
       }

       // Calculate the average price
       avgPrice = totalPrice / numSelectedWines;

       // Print summary of results after the chart
       cout << bar << endl;
       cout << "| " << "Total wines selected: " << numSelectedWines << endl;
       cout << "| " << "Average wine price: $" << avgPrice << endl;
       cout << "| " << sortTypeDisplay << endl;
       cout << bar << endl;
}

int main()
{
       printMyName();

       cout << endl << "-- Enter Wines --" << endl;

       // This is our main vector of wines
       vector<Wine> wineVect;

       // loop that allows user to enter new wines until breaking out by not entering 'y' at
the prompt
       bool doloop = false;
       do
       {
              // Create the new wine
              Wine myWine = getWineInput();

              // Add it to the vector
              wineVect.push_back(myWine);

              // Ask if we want to enter another
              cout << "Enter another wine? y,n: " << endl;
              char cont;
              cin >> cont;
              doloop = (cont == 'y');
       }
       while (doloop);

       // Now we want to know the range of price and rating that the user wants to display
       int minRating = 0, maxRating = 100, minPrice = 0, maxPrice = 1000;
       cout << endl << "-- Display Options --" << endl;
       cout << "Enter min desired rating: "; cin >> minRating;
       cout << "Enter max desired rating: "; cin >> maxRating;
       cout << "Enter min desired price: "; cin >> minPrice;
       cout << "Enter max desired price: "; cin >> maxPrice;

       // Prompt for sort type
       char sortId = 0; // r = rating, p = price, other = none
       cout << "How do you want to sort? r = rating, p = price, n = none: ";
```

```
	cin >> sortId;

	cout << endl << "-- Results --" << endl;

	// Display the wines
	printAllWines(wineVect, minRating, maxRating, minPrice, maxPrice, sortId);
}
```

wine.cpp

```
#include <string>
#include "wine.h"

// Constructors

Wine::Wine()
{
	name = "newwine";
	vintage = 0;
	rating = 0;
	price = 0.00;
	winery = "newwineary";
}

Wine::Wine(std::string newWineName, int newVintage, int newRating, double
newPrice, std::string newWineary)
{
	name = newWineName;
	vintage = newVintage;
	rating = newRating;
	price = newPrice;
	winery = newWineary;
}

// Mutators

// Change the name
void Wine::setName(std::string newName)
{
	name = newName;
}

// Change the vintage
void Wine::setVintage(int newVintage)
{
	vintage = newVintage;
}

// Change name, price, and rating all at once
void Wine::setInfo(std::string newName, double newPrice, int newRating)
{
	name = newName;
	price = newPrice;
	rating = newRating;
}

// Accessors
// get the name
std::string Wine::getName()
{
      return name;
}

// get the vintage
int Wine::getVintage()
{
      return vintage;
}

// get the rating
int Wine::getRating()
{
      return rating;
}

// get the price
double Wine::getPrice()
{
      return price;
}

// get the wineary
std::string Wine::getWineary()
{
      return wineary;
}
```

wine.h

```
#ifndef PERSON_H
#define PERSON_H

// Each wine has the following attributes:
// - Name
// - Vintage
// - Rating
// - Price
// - Wineary

class Wine
{
public:
      Wine(double price, std::string name);
      Wine(std::string newWineName, int newVintage, int newRating, double
newPrice, std::string newWineary);
      Wine();

      // Mutators
      void setName(std::string newName);
      void setVintage(int newVintage);
      void setInfo(std::string newName, double newPrice, int newRating);
      // Accessors
      std::string getName();
      int getVintage();
      int getRating();
      double getPrice();
      std::string getWineary();


private:
      std::string name;
      int vintage;
      int rating;
      double price;
      std::string wineary;
};

#endif
```