```
// This program sorts graphics cards
// Author James Wetters

#include <iostream>
#include <string>
#include <fstream>
#include <cstdlib>
#include <ctime>
#include "GPU.h"
using namespace std;

// Prototypes
void printMenu();
int deleteMember(graphicsCard theArray[], int size);
void sortAscending(graphicsCard theArray[], int size);
void sortDescending(graphicsCard theArray[], int size);
void randomSort(graphicsCard theArray[], int size);
void printResults(graphicsCard theArray[], int size);


// Constant variables
const int MAXARRAY = 15;

int main()
{
	// Initilize values
	char selection = 'n';
	graphicsCard gpu[MAXARRAY];
	int i = 0;
	int goodData = 0;			//number of objects with good data in the array

	// Input
	ifstream inputFile("gpu.txt");
	while (!inputFile.eof())
	{
		// Initilized place holder variables
		int x = 0;		// X holds an int
		string y;		// Y holds a string
		double z = 0; // Z holds a double

		// Input ID
		inputFile >> x;
		gpu[i].setId(x);

		// Input card name
		inputFile >> y;
		gpu[i].setCardName(y);

		// Input Cores
		inputFile >> x;
		gpu[i].setCores(x);

		// Input Clock
		inputFile >> x;
		gpu[i].setClock(x);

		// Input Memory
		inputFile >> x;
```

```
        gpu[i].setMemory(x);

        // Input Fill Rate
        inputFile >> z;
        gpu[i].setFillRate(z);

        i++;
}

// Number of objects with data
goodData = i;

//do while Menu
do
{
        // Initilized menue variable
        int option = 6;

        // Menu is printed to screen
        printMenu();

        // User Menus selction
        cin >> option;

        switch (option)
        {
        case 1:
                // Add member
                gpu[goodData].addGraphicsCard(goodData);
                break;

        case 2:
                // Delete member
                goodData = deleteMember(gpu, goodData);
        break;

        case 3:
                // Sort the list in ascending order
                sortAscending(gpu, goodData);
        break;

        case 4:
                // Sort the list in descending order
                sortDescending(gpu, goodData);
                break;

        case 5:
                // Random Sort
                randomSort(gpu, goodData);
                break;

        case 6:
                // Leave menu
                break;
        }

        // Prompt user to continue
        cout << "Would you like to continue? Y or N" << endl;
```

```
		cin >> selection;
	} while (selection == 'y' || selection == 'Y');

	// Write information to a file
	printResults(gpu, goodData);

	system("pause");
	return 0;
}



//************************************************************************
//      Print Menu
//
// Prints the menu to the screen
//************************************************************************
void printMenu()
{
	cout << "Graphics card database menu. " << endl;
	cout << "To add a graphics card to the database enter 1." << endl;
	cout << "To delete a graphics card to the database enter 2." << endl;
	cout << "To sort the database enter in ascending order of cores 3." << endl;
	cout << "To sort the database enter in descending order of cores 4." << endl;
	cout << "To randomize the graphics card database enter 5." << endl;
	cout << "To exit enter 6." << endl;
}


//************************************************************************
//     Delete member
//
// Deletes member
//************************************************************************
int deleteMember(graphicsCard theArray[], int size)
{
	// Initilize variables
	int id;
	int index = 0;
	int found = -1;			// Assume item will not be found
	int compare;

	// Propt user for ID# for deletion
	cout << "Enter ID#" << endl;
	// User enters ID of member to be deleted
	cin >> id;

	// Initilize compare
	compare = theArray[0].getId();

	// Step through the array of Graphics cards
	// comparing the ID number of each to find the
	// array index to be moved
	while (index < size && id != compare)
	{
		index++;
		compare = theArray[index].getId();
	}
```

```
	// Write over deleted array member with
	// the last item in the array
	if (index < size)
	{
		found = index;
		theArray[found] = theArray[size - 1];		// Move last list item to
overwrite target
		size = size - 1;
	}

	cout << found << endl;
}

//*************************************************************************
//	Sort Ascending
//
// Sort Graphics card array in ascending order
//*************************************************************************
void sortAscending(graphicsCard theArray[], int size)
{
	// Initilize temperary variables
	int end;
	graphicsCard temp;


	for (end = size - 1; end >= 0; end--)
	{
		for (int count = 0; count < end; count++)
		{
			if (theArray[count].getId() > theArray[count + 1].getId())
			{
				// Swap array indexes
				temp = theArray[count];
				theArray[count] = theArray[count + 1];
				theArray[count + 1] = temp;
			}
		}
	}
}


//*************************************************************************
//	Sort Descending
//
// Sort Graphics card array in ascending order
//*************************************************************************
void sortDescending(graphicsCard theArray[], int size)
{
	// Initilize temperary variables
	int end;
	graphicsCard temp;

	for (end = size - 1; end >= 0; end--)
	{
		for (int count = 0; count < end; count++)
		{
			if (theArray[count].getId() < theArray[count + 1].getId())
```

```
                        {
                                // Swap array indexes
                                temp = theArray[count];
                                theArray[count] = theArray[count + 1];
                                theArray[count + 1] = temp;
                        }
                }
        }
}


//*************************************************************************
//      Random Sort
//
// Randomly sorts graphics cards
//*************************************************************************

void randomSort(graphicsCard theArray[], int size)
{
        // Initilize variables
        graphicsCard temp;
        srand(time(0));             // Seed random number
        //
        for (int i = 0; i < size; i++)
        {
                // Set random number equal to index
                int index = rand() % size;

                // Swap objects array positions
                temp = theArray[i];
                theArray[i] = theArray[index];
                theArray[index] = temp;
        }
}

//*************************************************************************
//      Print Results
//
// Sort Graphics card array in ascending order
//*************************************************************************
void printResults(graphicsCard theArray[], int size)
{
        // Initilize file
        ofstream outfile;
        outfile.open("gpu2.txt");

        // Write information to the txt file
        for (int i = 0; i < size; i++)
        {
                outfile << theArray[i].getId() << " ";
                outfile << theArray[i].getCardName() << " ";
                outfile << theArray[i].getCores() << " ";
                outfile << theArray[i].getClock() << " ";
                outfile << theArray[i].getMemory() << " ";
                outfile << theArray[i].getFillRate() << endl;
        }

        // Close file
```

```
	outfile.close();
}
```

```
//
// Author James Wetters

#ifndef GPU_H
#define GPU_H
using namespace std;

class graphicsCard
{
private:
	// Data members
	int		id;
	string cardName;
	int		cores;
	int		clock;
	int		memory;
	double fillRate;

public:
	int size;
	// Constructor
	graphicsCard();
	graphicsCard(int sId, string sCardName, int sCores, int sClock, int sMemory,
double sFillRate);

	// Gets
	int getId() const
	{ return id; }
	string getCardName() const
	{ return cardName; }
	int getCores() const
	{return cores; }
	int getClock() const
	{ return clock; }
	int getMemory() const
	{ return memory; }
	double getFillRate() const
	{ return fillRate; }

	// Sets
	void setId(int change)
	{ id = change; }
	void setCardName(string change)
	{ cardName = change; }
	void setCores(int change)
	{ cores = change; }
	void setClock(int change)
	{ clock = change; }
	void setMemory(int change)
	{memory = change; }
```

```
        void setFillRate(double change)
        { fillRate = change; }

// Member Functions
        //add new member
        void addGraphicsCard(int& size);

};
#endif
```

```
#include <iostream>
#include <string>
#include "GPU.h"

using namespace std;

//************************************************************************
//      Constructor
//
// Initilizes object member variables when called
//************************************************************************

graphicsCard::graphicsCard(int sId, string sCardName, int sCores, int sClock, int
sMemory, double sFillRate)
{
        id = sId;
        cardName = sCardName;
        cores = sCores;
        clock = sClock;
        memory = sMemory;
        fillRate = sFillRate;
}

graphicsCard::graphicsCard()
{}

//************************************************************************
//      Add A Graphics Card
//
// Adds a Graphics Card
//************************************************************************

void graphicsCard::addGraphicsCard(int& size)
{
        // User enters ID#
        cout << "Enter ID#: " << endl;
        cin >> id;

        // User enters Graphics Card Name
        cout << "Enter the graphics card name: " << endl;
        cin >> cardName;
```

```
        // User enters Cores
        cout << "Enter the number of cores of the graphics card: " << endl;
        cin >> cores;

        // User enters Clock speed
        cout << "Enter the clock speed of the graphics card MHz: " << endl;
        cin >> clock;

        // User enters Memory Config
        cout << "Enter the memory configuration of the graphics card in GB: " << endl;
        cin >> memory;

        // User enters Texture Fill Rate
        cout << "Enter the texture fill rate of the graphics card: " << endl;
        cin >> fillRate;

        size = size + 1;
}
```

```
1001 TitanX          3072   1000   12     192
1002 980Ti           2816   1000   6      176
1003 980             2048   1126   4      144
1004 970             1664   1050   4      109
1005 960             1024   1127   2      72
1006 950             768           1024   2      49.2
1007 TitanBlack 2880 889           6      213
1008 Titan           2688   837           6      187.5
1009 780Ti           2880   875           3      210
1010 780             2304   863           3      160.5
```

```
1010 780 2304 863 3 160.5
1009 780Ti 2880 875 3 210
1008 Titan 2688 837 6 187.5
1007 TitanBlack 2880 889 6 213
1006 950 768 1024 2 49.2
1005 960 1024 1127 2 72
1004 970 1664 1050 4 109
1003 980 2048 1126 4 144
1002 980Ti 2816 1000 6 176
1001 TitanX 3072 1000 12 192
```

Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

To exit enter 6.

1

Enter ID#:

2003

Enter the graphics card name:

TitainZ

Enter the number of cores of the graphics card:

5000

Enter the clock speed of the graphics card MHz:

1355

Enter the memory configuration of the graphics card in GB:

12

Enter the texture fill rate of the graphics card:

197.5

Would you like to continue? Y or N

y

Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

To exit enter 6.

2

Enter ID#

1005

Would you like to continue? Y or N

y

Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

To exit enter 6.

3

Would you like to continue? Y or N

y

Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

To exit enter 6.

4

Would you like to continue? Y or N

y

Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

To exit enter 6.

5

Would you like to continue? Y or N

y

Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

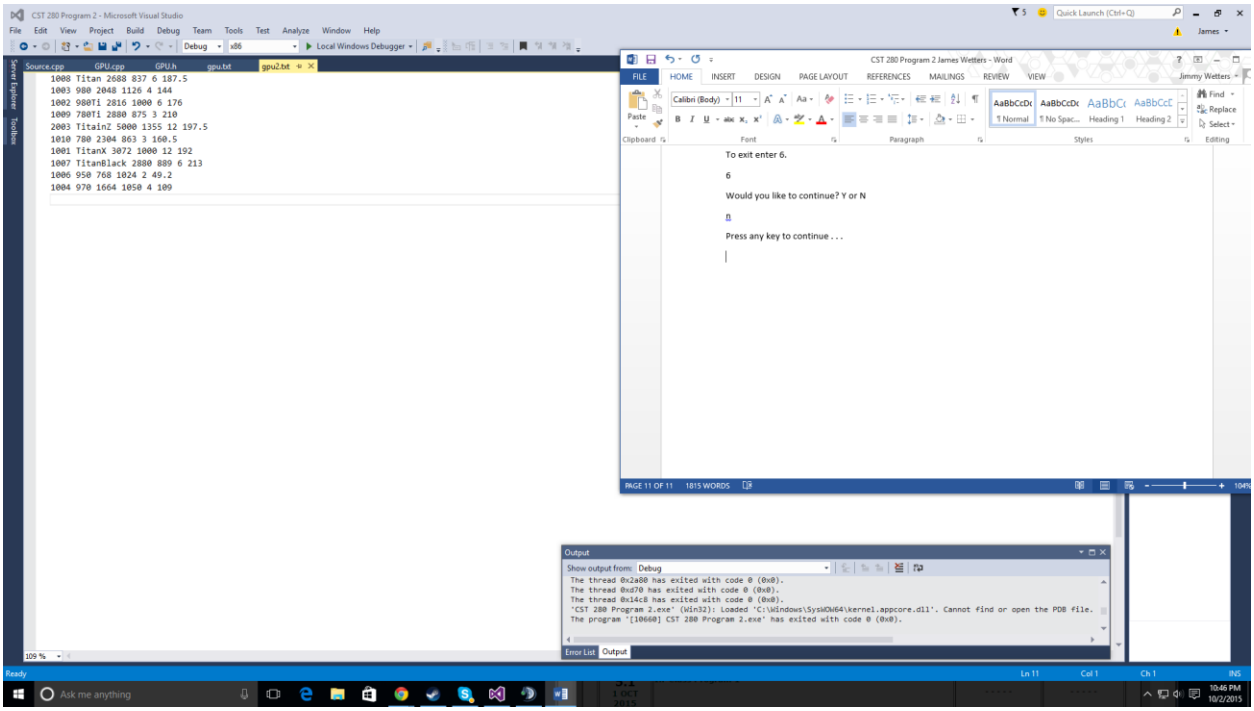
To exit enter 6.

6

Would you like to continue? Y or N

n

Press any key to continue . . .



Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

To exit enter 6.

2

Enter ID#

1005

Would you like to continue? Y or N

y

Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

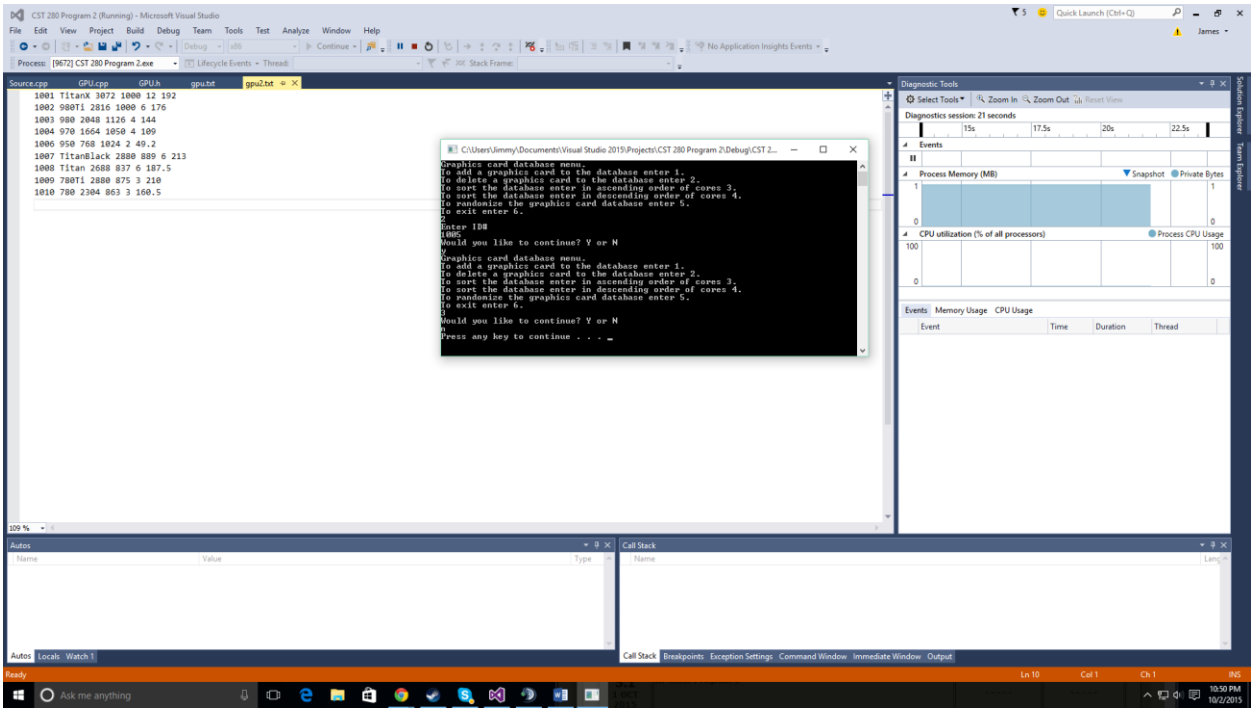
To exit enter 6.

3

Would you like to continue? Y or N

n

Press any key to continue . . .



Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

To exit enter 6.

2

Enter ID#

1005

Would you like to continue? Y or N

y

Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

To exit enter 6.

3

Would you like to continue? Y or N

y

Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

To exit enter 6.

2

Enter ID#

1002

Would you like to continue? Y or N

y

Graphics card database menu.

To add a graphics card to the database enter 1.

To delete a graphics card to the database enter 2.

To sort the database enter in ascending order of cores 3.

To sort the database enter in descending order of cores 4.

To randomize the graphics card database enter 5.

To exit enter 6.

4

Would you like to continue? Y or N

n

Press any key to continue . . .

```
1010 780 2304 863 3 160.5
1009 780Ti 2880 875 3 210
1008 Titan 2688 837 6 187.5
1007 TitanBlack 2880 889 6 213
1006 950 768 1024 2 49.2
1004 970 1664 1050 4 109
1003 980 2048 1126 4 144
1001 TitanX 3072 1000 12 192
```