CSE225L – Data Structures and Algorithms Lab
Lab 04
Unsorted List (array based)

In today's lab we will design and implement the List ADT where the items in the list are unsorted.

unsortedtype.h

```
#ifndef UNSORTEDTYPE_H
#define UNSORTEDTYPE_H

const int SIZE = 5;

template <class T>
class UnsortedType
{
private:
    T *data;
    int currentSize;
    int pointTo;

public:
    UnsortedType();
    ~UnsortedType();
    int Length();
    bool IsFull();
    void MakeEmpty();
    void Insert(T value);
    void Delete(T value);
    void Search(T value, bool &found);
    void GetNext(T &value);
    void Reset();
};

#endif // UNSORTEDTYPE_H
```

unsortedtype.cpp

```
#include "unsortedtype.h"

template <class T>
UnsortedType<T>::UnsortedType()
{
    data = new T[SIZE];
    currentSize = 0;
    pointTo = -1;
}

template <class T>
UnsortedType<T>::~UnsortedType()
{
    delete[] data;
}

template <class T>
int UnsortedType<T>::Length()
{
    return currentSize;
}
```

```
template <class T>
bool UnsortedType<T>::IsFull()
{
    return (SIZE == currentSize);
}

template <class T>
void UnsortedType<T>::MakeEmpty()
{
    currentSize = 0;
}

template <class T>
void UnsortedType<T>::Insert(T value)
{
    data[currentSize] = value;
    currentSize++;
}

template <class T>
void UnsortedType<T>::Delete(T value)
{
    int i = 0;
    while (value != data[i])
    {
        i++;
    }
    data[i] = data[currentSize - 1];
    currentSize--;
}

template <class T>
void UnsortedType<T>::Search(T value, bool &found)
{
    found = false;

    int i = 0;

    while (i < currentSize)
    {
        if (data[i] == value)
        {
            found = true;
            break;
        }
        else
        {
            i++;
        }
    }
}

template <class T>
void UnsortedType<T>::GetNext(T &value)
{
    pointTo++;
    value = data[pointTo];
}

template <class T>
void UnsortedType<T>::Reset()
{
    pointTo = -1;
}
```

Generate the **driver file (main.cpp)** where you perform the following tasks. Note that you cannot make any change to the header file or the source file.

Operation to Be Tested and Description of Action	**Input Values**	**Expected Output**
Create a list of integers		
Insert four items	5 7 6 9	
Print the list		5 7 6 9
Print the length of the list		4
Insert one item	1	
Print the list		5 7 6 9 1
Search 4 and print whether found or not		Item is not found
Search 5 and print whether found or not		Item is found
Search 9 and print whether found or not		Item is found
Search 10 and print whether found or not		Item is not found
Print if the list is full or not		List is full
Delete 5		
Print if the list is full or not		List is not full
Print the list		1 7 6 9
Delete 1		
Print the list		9 7 6
Delete 6		
Print the list		9 7
Write a class **Student** that represents a student record. It must have variables to store the student ID, student's name and student's CGPA. It also must have a function to print all the values. You will also need to overload a few operators.		
Create a list of objects of class **Student**		
Insert 5 student records	15234, Jon, 2.6 13732, Tyrion, 3.9 13569, Sandor, 1.2 15467, Ramsey, 3.8 16285, Arya, 3.1	
Delete the record with ID 15467		
Print the list		15234, Jon, 2.6 13732, Tyrion, 3.9 13569, Sandor, 1.2 16285, Arya, 3.1
Search the list for the item with ID 13569 and print whether it is found or not. If found, print the record.		Item is found 13569, Sandor, 1.2