Using deep copy

pointerDataClass.h

```
#ifndef POINTER_DATA_CLASS_H
#define POINTER_DATA_CLASS_H

#include<iostream>

using namespace std;

class pointerDataClass

{
	int maxSize;//variable to store the maximum size of p

	int length;//variable to store the number of elements in p

	int* p;// pointer to an int array

public:
	//Constructor to create an array of the size specified by the parameter size.

	pointerDataClass(int size);

	//Destructor to deallocate the memory space occupied by the array p

	~pointerDataClass();

	//the function insertAt inserts num into array p at the position specified by
	//index

	void insertAt(int index, int num);

	//The function displayData displays all the array elements in p

	void displayData();

	pointerDataClass(pointerDataClass& pdc);

};
#endif
```

pointerDataClass.cpp

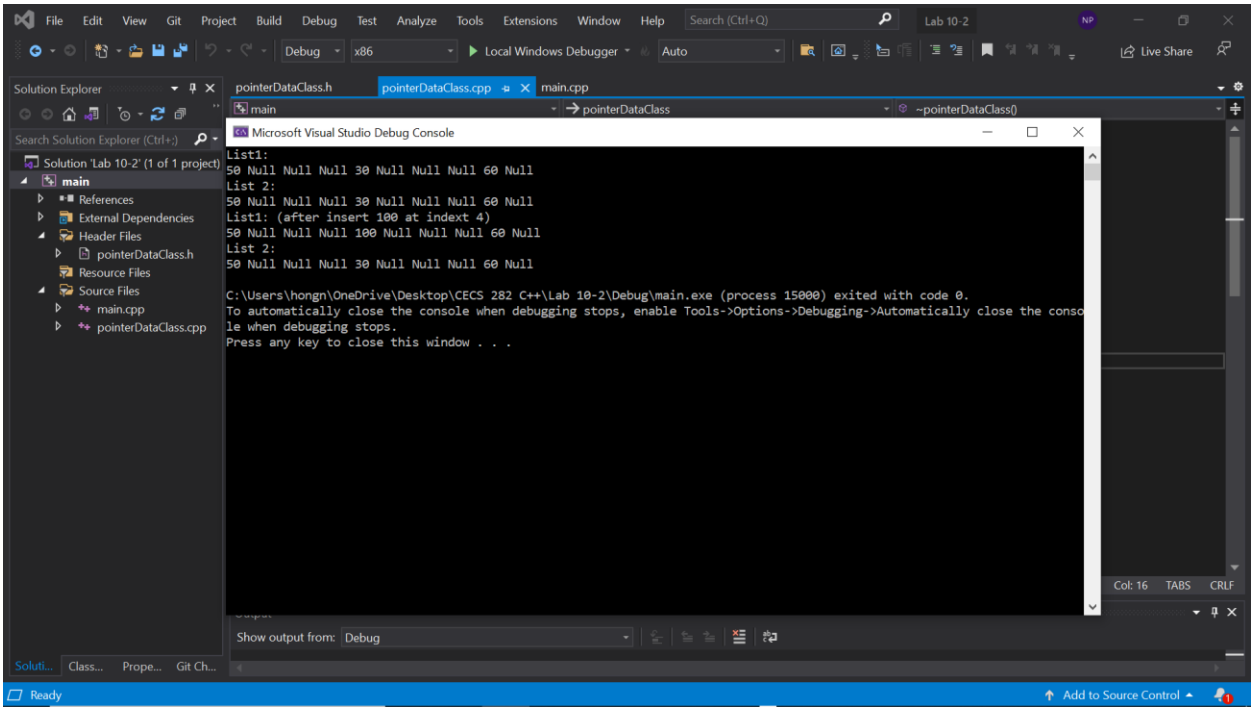
```
#include "pointerDataClass.h"
```

```
// using deep copy
pointerDataClass::pointerDataClass(int size) {
	maxSize = size;
	length = 0;
	p = new int[maxSize];
	for (int i = 0; i < maxSize; i++) {
		*(p + i) = NULL;
	}
}

//Destructor to deallocate the memory space occupied by the array p
pointerDataClass::~pointerDataClass() {
	delete[] p;
}

//the function insertAt inserts num into array p at the position specified by
//index
void pointerDataClass::insertAt(int index, int num) {
	*(p + index) = num;
	length++;
}

//The function displayData displays all the array elements in p
void pointerDataClass::displayData() {
	for (int i = 0; i < maxSize; i++) {
		if (*(p + i) == NULL) {
			cout << "Null" << " ";
		}
		else {
			cout << *(p + i) << " ";
		}
	}
	cout << endl;
}

pointerDataClass::pointerDataClass(pointerDataClass& pdc) {
	maxSize = pdc.maxSize;
	length = pdc.length;
	p = new int[maxSize];
	for (int i = 0; i < pdc.maxSize; i++) {
		*(p + i) = *(pdc.p + i);
	}
}
```

main.cpp

```
#include <iostream>
#include "pointerDataClass.h"

//Use deep copy
```

```
int main()
{
	pointerDataClass list1(10);
	list1.insertAt(0, 50);
	list1.insertAt(4, 30);
	list1.insertAt(8, 60);
	cout << "List1: " << endl;
	list1.displayData();
	cout << "List 2: " << endl;
	pointerDataClass list2(list1);
	list2.displayData();
	list1.insertAt(4, 100);
	cout << "List1: (after insert 100 at indext 4) " << endl;
	list1.displayData();
	cout << "List 2: " << endl;
	list2.displayData();
	return 0;
}
```



Demonstrated at 11:04 am on 10/21/2021.