01 - Pointers and Structures

Pre-Lab Exercises

Exercise 1

What is a pointer in computer programming? How are pointers used to create dynamic data structures in C++? Use examples of dynamically changing arrays. You may use Google search results to find answers to these questions, however, citing the reference in proper citation style (use any standard style) is necessary.

A pointer in C++ is simply a variable that stores a memory address, usually of another variable (Deitel and Deitel, 2012).

One of the simplest uses of pointers in programming is the creation of dynamic data structures or data structures whose size can change during runtime. Conventional C++ arrays are of limited use in practical applications like database programmes because their size is fixed at compile time, and cannot be changed. In practice, this means memory reserved by arrays is either too much or too little for real programmes.

However, arrays can be resized during compile time using pointers. This allows programmes to reserve only as much memory as is needed for a particular application at a particular point during runtime.

The following code illustrates the use of pointers and dynamic arrays.

```
#include <iostream>
using namespace std;

void printArray(int theArray[], int arraySize);

int main()
{
    //APPLICATION 1 – Extending static arrays
    int maxSize = 10;          //initially max size is to be 10
    int staticArray[maxSize];  //reserve space in memory for 10 ints

    for (int i = 0; i < maxSize; i++)
        staticArray[i] = i + 1;    //initialise theArray

    cout << "Initially, our static array is as follows" << endl;
    printArray(staticArray, maxSize);  //echo theArray

    cout << "\nWant to add another element to the array." << endl;
    cout << "Can't do this with staticArray" << endl;
    cout << "staticArray has space for only 10 ints." << endl;

    //create a pointer to a new int array of size 11
    int* dynamicArrayPtr = new int[maxSize + 1];
```

```
    //copy over existing elements to new array
    for (int i = 0; i < maxSize; i++)
        *(dynamicArrayPtr + i) = staticArray[i];

    //now max size of an array in our programme - 11
    maxSize++;

    //adding element to the new array
    dynamicArrayPtr[maxSize - 1] = 11;

    //echoing dynamicArray
    cout << "\nThe dynamic array is now:\n";
    printArray(dynamicArrayPtr, maxSize);

    //deleting dynamicArray
    delete []dynamicArrayPtr;   //return memory to freestore


    //APPLICATION 2 - Using dynamic arrays only
    cout << "\nCreating new dynamic array of size specified by user."
<< endl;
    cout << "How many elements would you like in your array?" << endl;

    int newArraySize;        //will store user entered size of array
    int* newArrayPtr = NULL; //will be assigned to a new memory block

    cin >> newArraySize;        //user enters size
    if (newArraySize > 0)       //if size positive integer
        newArrayPtr = new int[newArraySize];
    //create dynamic array, assign to newArrayPtr

    //initialise dynamic array to be even numbers
    for (int i = 0; i < newArraySize; i++)
        *(newArrayPtr + i) = 2*i;

    //output dynamic array
    cout << "\nHere's your array." << endl;
    printArray(newArrayPtr, newArraySize);

    //deleting dynamic array
    delete[] newArrayPtr;       //return memory to freestore

    cout << endl;
    return 0;
}//end main

void printArray(int theArray[], int arraySize)
{
    for (int i = 0; i < arraySize; i++)
        cout << theArray[i] << " ";
    cout << endl;
}
```

```
Initially, our static array is as follows
1 2 3 4 5 6 7 8 9 10

Want to add another element to the array.
Can't do this with staticArray
staticArray has space for only 10 ints.

The dynamic array is now:
1 2 3 4 5 6 7 8 9 10 11

Creating new dynamic array of size specified by user.
How many elements would you like in your array?
10

Here's your array.
0 2 4 6 8 10 12 14 16 18

Program ended with exit code: 0
```

Explanation

Application 1 shows how an array can be created during runtime with a pointer and the 'new' keyword. Such arrays take up memory in the freestore, and may cause a memory leak if they are not removed from memory through '`delete`' as soon as they are no longer needed. This can be problematic when the dynamic array/data structure has several complex, memory-intensive elements. If not removed from the freestore, these elements will continue to occupy memory, leaving less memory for concurrent programmes such as the OS.

Application 2 shows a more conventional use of dynamic arrays. Instead of copying elements from an existing static array, the dynamic array is used to store a user-defined/programmatically defined number of elements from the beginning. Application 2 requires less memory than application 1 because in the latter, memory conserved for the static array at compile time cannot be reallocated (Salvia, 2013).

The C++ STL `vector` class is actually a dynamically allocated array class. Using the principles of manual memory management and pointer manipulation similar to those demonstrated in the example above, the vector class encapsulates the functionality of a dynamic data structure in an efficient, robust, and reliable interface (En.cppreference.com, 2018).

Exercise 2

Define a database. What is a record inside a database?

A database is a collection of data or information in a computer's memory organised in a way that facilitates quick access, search, insertion, and deletion. A single entry in a database is called a record. Database information is always stored in the form of records, where each record is a collection of information classified into the same fields. (Rouse, 2017)

References

Deitel, P. and Deitel, H. (2012). *C++ How to Program*. 8th ed. Boston: Prentice Hall.

En.cppreference.com. (2018). *std::vector - cppreference.com*. [online] Available at: http://en.cppreference.com/w/cpp/container/vector
[Accessed 20 Jan. 2018].

Rouse, M. (2017). *What is database (DB)? - Definition from WhatIs.com*. [online] SearchSQLServer. Available at: http://searchsqlserver.techtarget.com/definition/database
[Accessed 20 Jan. 2018].

Salvia, C. (2013). *Must I delete a static array in C++?*. [online] Stackoverflow.com. Available at: https://stackoverflow.com/questions/14888737/must-i-delete-a-static-array-in-c
[Accessed 20 Jan. 2018].