**Programming Review**

Mark Tan

11 September 2017

CECS 326 MW

damtan1996@gmail.com

This C++ program allocates and deallocates a struct that comprises of an integer array and a char pointer array. It allows the user to access a pointer, deallocate memory and list the indices of the char pointer array that have been deallocated. When a user wants to access a pointer, one is allowed to either print the first ten characters in that pointer or delete all the chars associated with it. Memory is allocated based on the size defined in each index of the integer array, and chars are initialized to a random uppercase letter.

// Mark Tan

// progReview.cpp

// This file contains the program to allocate and deallocate

// a struct with an 20 char pointer array and a 20 integer array.

**#include** <cstdlib>

**#include** <iostream>

**using** **namespace** std;

**const** **int** index\_offset = 1;

**const** **int** doubler = 2;

**const** **int** letter\_count = 26;

**const** **int** ascii\_offset\_caps = 65;

**const** **int** size = 20;

**struct** **test** {

**char** \*\*array\_ptr;

**int** array\_int[size];

} a;

/\*\*

\* Performs the recursive function f(0) = 2900 & f(n+1)=2\*f(n).

\*/

**void** **recursion**(**int** array\_int[]) {

array\_int[0] = 2900;

**for**(**int** i = 0; i < size; i++)

array\_int[i + index\_offset] = doubler \* array\_int[i];

}

/\*\*

\* Initializes all char elements in pointer array to a random uppercase letter.

\*/

**void** **initialize**(**char**\*\* array\_ptr, **int** array\_int[], **int** i) {

**for**(**int** j = 0; j < array\_int[i]; j++)

array\_ptr[i][j] = (**char**) ((**rand**() % letter\_count) + ascii\_offset\_caps);

}

/\*\*

\* Allocates memory to all pointers and initializes char elements to a random uppercase letter.

\*/

**void** **allocate**(**char**\*\* array\_ptr, **int** array\_int[]) {

**for**(**int** i = 0; i < size; i++) {

array\_ptr[i] = **new** **char**[array\_int[i]];

**initialize**(array\_ptr,array\_int,i);

}

}

/\*\*

\* Checks if the index-defined slot in an array has been deallocated.

\*/

**void** **index\_check**(**int** index) {

**if**(a.array\_ptr[index] == 0) {

cout << "Index " << index << " in the s array is empty." << **endl**;

cout << "Now reallocating and reinitializing..." << **endl**;

a.array\_ptr[index] = **new** **char**[a.array\_int[index]];

**initialize**(a.array\_ptr,a.array\_int,index);

}

}

/\*\*

\* Prints the first ten chars in the index-defined slot in the pointer array.

\*/

**void** **print\_ten**(**int** index) {

**for**(**int** i = 0; i < 10; i++)

cout << a.array\_ptr[index][i];

cout << **endl**;

}

/\*\*

\* Deallocates the index-defined slot in the pointer array.

\*/

**void** **single\_deallocate**(**int** index) {

**delete**[] a.array\_ptr[index];

a.array\_ptr[index] = 0;

}

/\*\*

\* Lists all deallocated slots within the pointer array.

\*/

**void** **list\_deall\_mem**() {

cout << "Deallocated slots within the pointer array: ";

**for**(**int** i = 0; i < size; i++) {

**if**(a.array\_ptr[i] == 0)

cout << i << " ";

}

cout << **endl**;

}

/\*\*

\* Fully deallocates the pointer array.

\*/

**void** **full\_deallocate**() {

**for**(**int** i = 0; i < size; i++)

**delete**[] a.array\_ptr[i];

**delete**[] a.array\_ptr;

a.array\_ptr = 0;

}

**int** **main**()

{

//optionA controls main menu selection

//optionB controls access pointer menu selection

//index represents the slot in the array that the user wants to manipulate

//option\_count represents the number of the exit state

**int** optionA,optionB,index = 0;

**int** option\_count = 4;

a.array\_ptr = **new** **char**\*[size];

**recursion**(a.array\_int);

**allocate**(a.array\_ptr,a.array\_int);

**while**(optionA != option\_count) {

cout << "Main Menu" << **endl**;

cout << "(1) Access a Pointer" << **endl**;

cout << "(2) List Deallocated Memory" << **endl**;

cout << "(3) Deallocate All Memory" << **endl**;

cout << "(4) Exit Program" << **endl**;

cin >> optionA;

**switch**(optionA) {

**case** 1:

cout << "Which array index do you want to manipulate? ";

cin >> index;

**index\_check**(index);

cout << "Access Pointer Menu" << **endl**;

cout << "(1) Print the First 10 Chars in Chosen Array" << **endl**;

cout << "(2) Delete Chars Associated with This Pointer" << **endl**;

cout << "(3) Return to Main Menu" << **endl**;

cin >> optionB;

**switch**(optionB) {

**case** 1:

**print\_ten**(index);

**break**;

**case** 2:

**single\_deallocate**(index);

**break**;

**case** 3:

cout << "Returning to Main Menu..." << **endl**;

cout << **endl**;

**break**;

**default**:

cout << "try again" << **endl**;

**break**;

}

**break**;

**case** 2:

**list\_deall\_mem**();

**break**;

**case** 3:

**full\_deallocate**();

**break**;

**case** 4:

**full\_deallocate**();

**break**;

**default**:

cout << "try again" << **endl**;

**break**;

}

}

**return** 0;

}