InClass 11 Exercise – Pointers and Variable Addresses

Assignment Overview

Pointers are variables that contain the memory address of data, i.e. typically of variables or elements of arrays. They involve indirect access to data and require the programmer to think in terms of a “double hop” to data, i.e. values "in" variables. Pointers add a new level of abstraction and difficulty to programming. Most programming languages do not provide such low-level access to data. This is one of the main aspects that make C++ and C such powerful languages. Unfortunately, neither the language nor compilers do any checking on values used as addresses. Making sure that pointers have valid values is the sole responsibility of programmers.

As one student found out with the last Homework, if you access memory not inside your program accidentally, then your virus protection software may block your program from running! You can accidentally do this by using an index past the end of an array, or mis-setting a pointer variable and trying to set the memory at that address to a value or just using that value in a calculation.

This assignment has you 1) output the addresses of variables and the values of pointers (addresses), 2) use pointers to access elements of arrays, and 3) use and write functions that have parameters of arrays and pointers and addresses of variables. I hope that you think about what you are doing to accomplish these things to get the proper output and that you really consider what the output means. If you don’t look at the addresses in the output, you aren’t understanding the material.

Most professional programmers consider pointers one of the most difficult topic in programming. I highly recommend you attend class or at the least you really need to watch the recordings for this week.

Complete the Code

Copy and paste the following code into CodeBlocks and
Fill in the Blanks:

- Fill in the places where there is a ; semicolon by itself, i.e. you need to put in a statement where there is a null statement.
- Fill in places with incomplete lines, i.e. with blanks _____

```
#include <iostream>
#include <iomanip>
#include <fstream>

using namespace std;

void readInts(string, int*, int *);
void readIntArray(ifstream &inStream, int iarr[], int l);
```

```
void printIntArray(int *, int);
void swapFirstLast(int [], int);
void swapFirstSecond(int []);
void swapInts(int*, int*);

int main()
{
// declare an int const LEN = 10
    ;
// declare 2 double pointers dptr1 and dptr2
    ;
// declare 2 double variables dnum1, and dnum2, initializing
//    them to 2.5 and 100.1
    ;
// declare a double array called darr of length LEN, initializing
//    to 1.5 2.2 3.7 4.5 5 and LEN, with the rest of the elements 0
    ;

// declare 2 int pointers iptr1 and iptr2
    ;
// declare 2 int variables
     ;
// declare an int array called iarr of length LEN, initializing
//    to 42 23 43 71 and 101, with the rest of the elements 0
    ;
    string filename;
    ifstream inStream;

// to dptr1 assign the adresss of dnum1
    ;
    cout << fixed << setprecision(4);
// print the value of dptr1 as an int and in hexadecimal
    cout << "doubles " << __________ << "(long) = "
         << __________ << "(hex)\n";
// print the addresses of dptr1 and  dptr2
    cout << "dptr addresses [" << (long)___________ << "]  ["
         << _____________ << "]\n";
// print the addresses of dnum1 and  dnum2
    cout << "dnum addresses [" << ___________ << "]  ["
         << _____________ << "]\n";
// print the address of the beginning of darr
    cout << "darr address [" << ____________ << "]\n\n";

// print the addresses of iptr1 and iptr2
    cout << "ints\n" << "int ptrs [" << _____________
         << "]  [" << ____________ << "]\n";
// print the addresses of inum1 and inum2
```

```
    cout << "inum addresses [" << ____________ << "]  ["
         << _____________ << "]\n";
// print the address of the beginning of iarr
    cout << "iarr address [" << _____________ << "]\n\n";

    cout << "Values of the double variables:    ";
// print dnum1 and dnum2
    cout << dnum1 << "  " << dnum2 << endl;
// assign dptr1 the address of dnum1
    ;
// make dptr2 point to the same thing as dptr1
    ;
// use * of dptr2 to set the VALUE of dnum1 to 3.14159
    ____________ = 3.14159;
    cout << "  Values of double variables again: ";
// print dnum1 and dnum2 again
    cout << dnum1 << "  " << dnum2 << endl;

// print the values of what dptr1 and dptr2 point at
    cout << "value pointed at by dptr1 & dptr2: "
         << ____________ << "  " << ___________ << endl;

    cout << "int array:\n";
// set iptr1 equal to the beginning address of iarr
    ;
    for (int i=0; i < LEN ; i++)
    {
// use iptr1 to print each of the 10 numbers in the
//     int array and increment the pointer
        cout << "  " << _________________ << endl;
    }

    cout << "==============\n\n";
    cout << "double array:\n";
// set dptr1 equal to the beginning address of darr
    ;
    for (int i=0; i < LEN ; i++)
    {
// use dptr1 to print each of the 10 numbers in the
//     double array and increment the pointer
        cout << "  " << ___________ << endl;
    }

// set iptr2 to the address of inum2
    ;
// call void readInts( string promptString, int* int1, int* int2)
//     - read values for inum1 and use iptr2 to read a value for inum2
```

```
    readInts("Enter 2 ints to return: ", &inum1, iptr2);
    cout << "ints read by readInts(): " << inum1 << ", "
// 2 different values can fill in this blank. You need to know both.
         << _____________ << endl << endl;

    cout << "enter a file name to read from: ";
    cin >> filename;
    inStream.open(filename.c_str());
    if ( !inStream )
    {
        cout << "Error!\n";
        return 1;
    }

// call void readIntArray(ifstream infile, int intArray[], int len)
    ;
    cout << "PRINT int ARRAY 1:\n";
// call void printIntArray(int arr[], LEN)
    ;

    cout << "swap first and last elements in the array\n";
// call void swapFirstLast(int arrINTs[], LEN)
    ;
    cout << "PRINT int ARRAY 2:\n";
// call void printIntArray(int arr[], LEN)
    ;

    cout << "swap first and second elements of the array\n";
// call void swapFirstSecond(int arrints[])
    ;
    cout << "PRINT int ARRAY 3:\n";
// call void printIntArray(int arr[], LEN)
    ;

    cout << "Point iptr1 to second and iptr2 to "
         << "third numbers in the iarr\n";
    ;
    ;
    cout << "Swap ints pointed at by iptr1 and 2.\n";
// call void swapInts(int *p1, int *p2)
    ;
    cout << "PRINT int ARRAY 4:\n";
// call void printIntArray(int arr[], LEN)
    ;

    return 0;
}
```

```
void readInts(_____________________________)
{
    cout << prompt ;
    cin >> ____________ >> __________;
    return;
}

void readIntArray(_________________________)
{
    for (int i = 0; i < l ; i++)
    {
        in >> _____________;
    }
    return;
}

void printIntArray(_______________________)
{
    int i;
    for ( i = 0; i < len-1 ; i++)
    {
        cout << ___________________ << ", ";
    }
    cout << ___________________ << endl << endl;
    return;
}

void swapFirstLast(_____________________)
{
    int ___________________;
    ;
    ;
    return;
}

void swapFirstSecond(___________________)
{
    int _______________;
    ;
    ;
    return;
}

void swapInts(_____________________)
{
    int _____________;
```

```
    ;
    ;
    return;
}
```

Output

```
doubles 7274152(long) = 0x6efea8(hex)
dptr addresses [7274164]  [7274160]
dnum addresses [7274152]  [7274144]
darr address [7274064]

ints
int ptrs [7274060]  [7274056]
inum addresses [7274052]  [7274048]
iarr address [7274008]

Values of the double variables:    2.5000  100.1000
  Values of double variables again: 3.1416  100.1000
value pointed at by dptr1 & dptr2: 3.1416  3.1416
int array:
  42
  23
  43
  71
  101
  0
  0
  0
  0
  0
==============

double array:
  1.5000
  2.2000
  3.7000
  4.5000
  5.0000
  10.0000
  0.0000
  0.0000
  0.0000
  0.0000
Enter and 2 ints: 34 22
ints read by readInts(): 34, 22

enter a file name to read from: in
PRINT int ARRAY 1:
2, 4, 6, 8, 11, 13, 15, 17, 19, 1

swap first and last elements in the array
PRINT int ARRAY 2:
1, 4, 6, 8, 11, 13, 15, 17, 19, 2
```

```
swap first and second elements of the array
PRINT int ARRAY 3:
4, 1, 6, 8, 11, 13, 15, 17, 19, 2

Point iptr1 to second and iptr2 to third numbers in the
iarr
Swap ints pointed at by iptr1 and 2.
PRINT int ARRAY 4:
4, 6, 1, 8, 11, 13, 15, 17, 19, 2
```

Input File

```
2
4
6
8
11
13
15
17
19
1
```