lab 05 Pointers

Instructions: It is ***vital*** that you understand pointers and, consequently, the different memory spaces: static, stack, and heap. Please answer the following questions:

Note: When a question asks for the *value* of a variable, if it is a known number write the number. If it is a memory address write what the memory address is pointing to (Ex: The variable x holds the memory address of the variable y.) If it cannot be determined, write undefined.

1. (15 pts) Consider the following code:

```c
#include <stdio.h>

int main(int argc, char *argv[]) {
    int x = 32;
    int *y = &x;
    int z = *y;
    printf("z = %d\n", z);
    return 0;
}
```

   (a) What is the output?

   (b) What is the *value* of y?

   (c) In what memory (static, stack, or heap) do the variable x, y, z exist during runtime?

2. (15 pts) Consider the following code:

```c
#include <stdio.h>

int main(int argc, char *argv[]) {
    int *x = (int *)200;
    long z = (long)x;
    printf("z = %d\n", z);
    return 0;
}
```

   (a) What is the output?

   (b) In what memory (static, stack, or heap) do the variable x, z exist during runtime?

   (c) Why is this code bad practice (even though it compiles/runs without a seg fault)?

3. (15 pts) Consider the following code:

```
#include <stdio.h>

int main(int argc, char *argv[]) {
    int *x = new int[100];
    x[0] = 500;
    int z = x[25];
    printf("z = %d\n", z);
    return 0;
}
```

(a) What is the output? (Careful, a compile/run will not give you the correct answer.)

(b) In what memory (static, stack, or heap) do the variable x, z exist during runtime?

(c) What is stored on the heap?

(d) What is the memory issue?

4. (15 pts) Consider the following code:

```
#include <stdio.h>

int main(int argc, char *argv[]) {
    int *x = new int[100];
    int *y = x+10;

    for (int i = 0; i < 100; i++) {
        x[i] = i;
    }

    printf("y[10] = %d\n", y[10]);
    return 0;
}
```

(a) Why is the output?:
y[10] = 20

(b) Write a line of code that would free up memory using only x.

(c) Write a line of code that would free up memory using only y.

5. (15 pts) Consider the following code:

```
 1 #include <stdio.h>
 2
 3 int main(int argc, char *argv[]) {
 4     int *x = new int[100];
 5     int *y = new int[100];
 6     int **z = NULL;
 7
 8     for (int i = 0; i < 100; i++) {
 9         x[i] = i;
10         y[i] = 100-i;
11     }
12
13     z = &x;
14     printf("(*z)[10] = %d\n", (*z)[10]);
15     z = &y;
16     printf("(*z)[10] = %d\n", (*z)[10]);
17     return 0;
18 }
```

(a) What is the *value* of x and y?

(b) What is the *value* z on lines 14 and 16?

(c) What does the code fragment "(*z)[10]" mean? (Describe what the code must do the evaluate that code fragment.)

(d) Why are there two different outputs for "(*z)[10]"? Here is the output of the program:
(*z)[10] = 10
(*z)[10] = 90

(e) Write the commands to free memory.

6. (25 pts) Consider the following code:

```
#include <stdio.h>
#include <stdlib.h>

class Student {
   public:
      int mId;
      double mGPA;
      std::string mAddress;
      std::string mBiography;
};

int main(int argc, char *argv[]) {
   Student *students = new Student[100];
   Student ** studentsPtr = new Student*[100];

   srand(100); // Seed random number generator
   for (int i = 0; i < 100; i++) {
      students[i].mId = i+1;

      // Generate a "random" GPA from 0.0-4.0
      students[i].mGPA = 4 * (((double)rand())/RAND_MAX);

      studentsPtr[i] = students+i;
   }

   // This is Bubble Sort:
   for (int i = 0; i < 100; i++) {
      for (int j = 1; j < 100; j++) {
         // Based on GPA
         if (studentsPtr[j-1]->mGPA > studentsPtr[j]->mGPA) {
            Student *temp = studentsPtr[j];
            studentsPtr[j] = studentsPtr[j-1];
            studentsPtr[j-1] = temp;
         }
      }
   }

   for (int i = 0; i < 100; i++) {
      printf("%f\n", studentsPtr[i]->mGPA);
   }

   return 0;
}
```

(a) What is the *value* of students?

(b) What is the *value* of studentsPtr?

(c) Which of the above is being sorted?

(d) Why was the above (in c) chosen over the other? What advantage do you see?

(e) Write the commands to free memory.

How to turn in:
Turn in via GitHub. Ensure the file(s) are in your directory and then:

- $ git add <files>
- $ git commit
- $ git push

Due Date: September 09, 2020 2359

Teamwork: No teamwork, your work must be your own.