CS 210 - Introduction to Computer Science

Fall 2016 - Exam 2 Review Sheet

Exam 2 will be on Tuesday, October 18, 2016. The exam is open book and open notes.

The exam will consist of questions on the material in Chapters 6, 7 and 11 with the exception of section 7.8 on multidimensional arrays, 7.10 on graphics, and section 11.2 on binary files. The exam will have three sections. In section 1 you will be asked to answer short answer questions related to all of the topics in the reading material. In section 2 you will be given a program or a portion of a program an asked to explain what the program does. In section 3 you will be asked write a program.

Topics covered include the following:

- Parameter passage by reference and by value
- Programs using pointers
- One-dimensional arrays
  - Searching and sorting (bubble sort and select sort)
  - Passing arrays as parameters
- Opening, using, and closing text files

When you are asked to write code, you will not need to write comments, include directives, or output formatting beyond producing newlines in appropriate places.

Short Answer

1. What is a stack and what is it commonly used for?

2. It is permissible to do arithmetic on pointers. You can increment or decrement a pointer and you can add or subtract a value from a pointer but you cannot multiply or divide using a pointer. Why not?

3. Since a pointer is an address to a memory location why is it important that we declare the type of data that a pointer points to as in `int *iPtr` and `double *dPtr`?

4. Why are arrays, by default, passed by reference instead of by value like other variables?

5. Since arrays are passed by reference, how can we insure that a function does not change an array that we don't want to be changed?

6. Explain how the use of functions in a program saves memory.

7. What is the purpose of the open and close statements with regard to files.

8. Supposed I declare a variable and a pointer as:

```
int a = 15;
int *aPtr = &a;
```

If the variable *a* is stored in memory at location 1234, show what is printed by each of the following:

`printf("%d\n",a);`	`printf("%d\n",&a);`	`printf("%d\n",aPtr);`

`printf("%d\n",*aPtr);`	`printf("%d\n", &*aPtr);`	`printf("%d\n",*(aPtr));`

9. Suppose I define an array as shown below. Show what each of the printf statements print. Use XXX if the print statement attempts to print an illegal value.

```
int i, a[10];
for(i=0;i<10;i++)
   a[i] = i+2;
i = 3;
```

printf("%d\n", a[i]);	printf("%d\n", a[a[i]]);	printf("%d\n", a[a[a[2]]]);

10. Write a short sequence which will set the pseudo-random number generator seed to the current time and store 1000 pseudo-random numbers in an array. The number should be ints that range from $100 \leq x < 200$. Assume that includes have been written for STDIO, STDLIB, and TIME.

Sample Programs

1. The following sequence uses a function called MinMax to find the minimum and maximum of an array. Write the function.

```
{int a[] = {1, 5, 2, 6, 8, 23, 45};
 int min, max;
 int SIZE = 7;
 MinMax(a, &min, &max, SIZE);
 printf("The min is %d, the max is %d", min, max);
}
```

2. Write a method which will accept an `int` array called *a* and will return the largest difference between any two adjacent elements. For example, the following sequence will print 4 since the largest difference is 7 – 3 = 4.

```
int[] a = {1, 1, 3, 7, 8, 9};
printf("%d\n", FindMaxDiff(a, 6));
```

3. Show how to create a 1-dimensional array of doubles and fill it with the number x where x is the product of PI and the array index. The array should have 3,482 elements.

4. Write a method which receives a 1-dimensional array named *a[]* and returns the *index* of the first nonzero element in the array. If there are no nonzero elements in the array your method should return -1.

5. Create an array of 100 random ints which range in value from 0 to 1000. Calculate and print the minimum and maximum values stored in the array.

6. Create two arrays of doubles called x and y which have the following data: x = {0, 2.2, 3.4, -9.5, 10.3, 7.5, 2.1}, y = {0, -2.3, -8.5, -2.1, 4.4, 6.5, 0.9}. Treat values in the x and y vectors which have the same index as representing a point on a two dimensional grid. Use the distance formula to find the two points which are the farthest apart.

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

7. The Sieve of Eratosthenes is an ancient simple method of finding prime numbers. To use the sieve to find all of the prime number between 1 and say 1000, we make an array of all integers from 1 to 1000. Begin with the number 2 and delete all multiples of 2. Next take 3 and delete all of the higher multiples of 3. Continue eliminating multiples until you get to the square root of 1000. The numbers left in the array will be the primes from 1 to 1000. Write a program to do the Sieve of Eratosthenes and print the resulting prime numbers.

8. Fill in the memory map below for the following program:

```
#include<stdio.h>
int Confused(int y,  int *z);
int main()
  {int x = 9, y = 14, z = 2;
   printf("%d %d %d\n", x, y, z);
   z = Confused(x, &y);
   printf("%d %d %d\n", x, y, z);
  }
//
int Confused(int y,  int *z)
  {int a = 13, b;
   printf("%d %d %d\n", a, y, *z);
   b = 8;
   *z = 12;
   return a + b + *z;
  }
```

Confused	Main	Data

Printed Results

_____
_____
_____
_____
_____
_____
_____

9. The main program below create and array named *a* with 1000 random ints and a second array *aRev* of the same size but empty. Write a function called Reverse which accepts the two arrays as parameters and returns with *aRev* having all of the values of *a* but in reverse order.

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

int main()
   {int i, a[50], aRev[50];
    srand((unsigned) time(NULL));
    for(i=0;i<50;i++)
       a[i] = rand();
    Reverse(a, aRev);
    for(i=0;i<50;i++)
       printf("%d, %d\n", a[i], aRev[i]);
    return 0;
}
```

10. A file named `Number.txt` contains an unknown number of ints with one int per line. Write a program which opens the file and prints all of the ints which are greater than zero to a second file named `Positive.txt`. Your program should print the number of positive numbers to the console.