A) Multiple Choice / True False

(*Circle* the letter beside the best answer, do not fill in the blanks)

1. Compare the array and pointer (dynamic) implementations of a queue in C. What happens in each case if you enqueue too many elements to the queue without removing any (dequeue)?

   a) The array implementation will run out of memory (crash), and the pointer implementation will report that the queue is full.
   b) The pointer implementation will run out of memory (crash), and the array implementation will report that the queue is full.
   c) They will both run out of memory (crash).
   d) They will both report that the queue is full.

2. The pupose of Dijkstra's algorithm related to graphs is to

   a) Traverse the graph in depth-first order
   b) Build a graph given an adjacency list
   c) Find and eliminate cycles in the graph
   d) Find the most efficient path through a graph

3. When building a binary search tree the most efficient tree will result if the value at the root of each subtree (and the root of whole tree) happens to be in the middle (median) of all the values in nodes below it, because then the tree will be perfectly balanced. Suppose you build a binary search tree and the value at the root of each subtree happens to be at the 25% percentile – in other words 25% of the values in nodes below it are smaller, and 75% are larger. Then the computational complexity of searching this tree will be

   a) $O(\log_2(n))$
   b) $O(n \log(n))$
   c) $O(n)$
   d) Between $O(\log_2(n))$ and $O(n)$

4. Suppose you declare a variable but don't initialize it. Which of the following may result in an unpredictable/undefined variable value? Choose all that apply.

   a) A global variable with data type 'double'
   b) A local variable with data type 'bool'
   c) A local variable that's an array of integers size 100
   d) A variable declared with the keyword 'static'

B) Fill in the Blanks

5. Suppose the computational complexity of a data processing algorithm is **O(n^2)** and you measure the CPU time it takes to process 100 items. If the measured time is 580 ms, how much CPU time would you expect the algorithm to require for 400 items?

   ____

   Bonus (more challenging): What about if the computational complexity was **O(n log(n))** where you know the base of the logarithm is 2?

   ____

6. What is the big-O computational complexity for the following pseudocode algorithm?

```
for (int i = 0; i < 99; i++) {
    for (int j = 0; j < n; j++)
        for (int k = 0; k < n; k++) {
            sum++;
        }
    }
}
```

   ____

7. Draw the directed graph for the following graph definition data file (format is as discussed in class).
   9
   A B C D E F G H I
   A 2 B 25 H 45
   B 0
   C 2 B 10 D 70
   D 0
   E 2 D 65 F 20
   F 0
   G 2 H 40 F 31
   H 0
   I 8 A 5 B 10 C 15 D 20 E 25 F 30 G 35 H 40

8. Define the term Abstract Data Type. Be brief.

9. List the three main steps which take place when compiling C source files to produce an executable file (.exe).

   (1) ________________________________

   (2) ________________________________

   (3) ________________________________

10. Examine the function declaration below. Under each parameter write C or N to indicate whether the data for that parameter is Copied (function can't change original value) or Not copied (function can change the original value). The first one is an example. Note: Assume 'Student' is a struct that has been previously declared.

```
void manyArgs(float val, char str[], Student st, char ch, int *arr);
                    C        ___          ___        ___      ___
```

11. Given the following structure and variable declarations, write how to print the 'jewels' field of the variable ppl (assume the variable is properly initialized).

```
typedef struct {
    int swords;
    int jewels;
    int potions;
} ObjectCount;

typedef struct {
    char nickName[31];
    int health;
    ObjectCount *objCnt;
} Player;

Player *ppl;
```

Answer: ________________________

C) Programming

12. Make sure you've completed all exercises in all of the course slides from week 1 to week 13. Do it without looking at my posted solutions.

13. Suppose the data type Stack has been declared as in the course slides (week 9). Write a main function to:
    - Declare and initialize a variable with data type Stack.
    - Input numbers from the user and push them on the stack until the user enters -1.
    - Pop all the numbers and print them.

14. Modify the binary search tree code from Week 12 so that findOrInsert() returns NULL if the item was not found (inserted). It should return the existing item pointer if found as before. Now write a program that does these two steps:
    - Input strings from the user to build the tree until the user enters "X".
    - Input more strings from the user, these are strings to search for. In each case print "Node was found" or "Node was not found (inserted)" as appropriate. Quit on "X".

15. Write a C function that accepts a string (array of char) as a parameter and encodes each letter in the string using exclusive-or with 0x5A. The exclusive-or operator in C is the same as in Java: ^
    A useful property of this secret code is that you can encode again to decode a message! Test your program by writing a main function that inputs a message, encodes it and prints the result (it may look funny), then decodes it and prints the result.

16. Write a C program that does the following in the order shown:
    - Input 6 double floating-point numbers from the user and store them in a dynamic array of size 6. Store the current size of the array in a variable to avoid hard-coding the size. Make the dynamic array itself a global/module variable.
    - Sort the numbers in ascending order using the C library **qsort** function and print the sorted array. Remember you'll need to write an appropriate comparison function.
    - Enlarge the dynamic array to double its previous size.
    - Input 6 more numbers from the user and store them in the second half of the array.
    - Use the heap sort algorithm from my slides to sort the entire array again, then print it out. You may have to add data types for some variables in the heap sort code. You'll also need to implement 'swap', the easiest way is to just do the swap (3 lines) without calling a function.