Fall, 2024 Name: _____

(Please *don't* write your id number!)

COP 3402 — Systems Software

Midterm Exam

Directions for this Test

This test has 10 questions and pages numbered 1 through 5.

This test will be for the rest of the class time (about half of the time for today's class) and is be closed book.

However, you may use one (1) page of notes on one (1) side of a standard 8.5 by 11 inch sheet of paper. These notes can either be hand-written or printed, but if printed, then the font must be a 9-point or larger font. These notes must be turned in with the exam.

If you need more space, use the back of a page. Note when you do that on the front.

Before you begin, please take a moment to look over the entire test so that you can budget your time.

For Grading

Question:	1	2	3	4	5	6	7	8	9	10	Total
Points:	10	10	10	10	10	10	10	10	10	10	100
Score:											

All questions on this exam are related to the course outcome [Concepts].

1. (10 points) Which of the following are kinds of systems software? (Circle **each** answer letter that is correct.)

   A. A spreadsheet, such as Microsoft Excel.

   B. A compiler, such as a Java compiler.

   C. An operating system, such as Linux.

   D. A social networking app., such as Instagram.

   E. An interpreter, such as a Python interpreter.

   F. A photo editor, such as Adobe Photoshop.

2. (10 points) Why is it important that a Virtual Machine (VM) supports subroutines? (Circle the **one** correct answer's letter.)

   A. Because there was a trend, starting in 1990 at Stanford University, of having subroutines in VMs, and that trend is continuing to this day.

   B. Because subroutines guarantee job security for Computer Scientists, since they are very confusing to understand and maintain.

   C. Because without support from the VM, a program's code would collapse into a pile of bits.

   D. Because subroutines allow programmers to hide details of algorithms from client code, making the program easier to understand and maintain.

3. (10 points) What is the purpose of having a `.c` file, say `f.c`, include the `.h` file with the same base name (i.e., `f.h`)? (You may assume that the **`#include`** of the `.h` file occurs before any other code in the `.c` file.) (Circle the letter of **each** correct answer.)

   A. It allows the front end of the C compiler to know about the beginning of the module.

   B. It allows the C compiler to check the consistency of the declarations in the `.h` file with the definitions in the `.c` file.

   C. It allows the `.c` file to implement the functions declared in the `.h` file in any order, even if they are mutually recursive (call each other).

   D. It allows the author of the program to earn a bonus for creating a very large program, since the `.c` file includes the `.h` file, which in turn can include the `.c` file, and so on, indefinitely.

4. (10 points) In the Simple Stack Machine (SSM) VM's instruction cycle, what is the order in which events happen? (Circle the **one** correct answer's letter.)

   A. The execution takes place in this order: (1) the PC is incremented, (2) the instruction at the address given by the PC is read, then (3) the instruction's semantics (effect) is done by the VM.

   B. The execution takes place in this order: (1) the instruction at the address given by the PC is read, (2) the PC is incremented, then (3) the instruction's semantics (effect) is done by the VM.

   C. The execution takes place in this order: (1) the instruction read in the previous cycle has its semantics (effect) simulated by the VM, (2) the PC is incremented, then (3) the instruction at the address given by the PC is read.

   D. The execution takes place in this order: (1) the instruction at the address given by the PC is read, (2) the instruction's semantics (effect) is done by the VM, then (3) the PC is incremented.

5. (10 points) What would be a straighforward and useful definition of the SSM VM's memory? (Assume that `word_type`, `uword_type`, and `bin_instr_t` are declared in included `.h` files, as in the files provided for the homework.) (Circle the **one** correct answer's letter.)

   A.
   ```
   static union {
       word_type words[MEMORY_SIZE_IN_WORDS];
       uword_type uwords[MEMORY_SIZE_IN_WORDS];
       bin_instr_t instrs[MEMORY_SIZE_IN_WORDS];
   } memory;
   ```

   B.
   ```
   static struct {
       word_type words[MEMORY_SIZE_IN_WORDS];
       uword_type uwords[MEMORY_SIZE_IN_WORDS];
       bin_instr_t instrs[MEMORY_SIZE_IN_WORDS];
   } memory;
   ```

   C.
   ```
   static union {
       word_type words[MEMORY_SIZE_IN_WORDS];
       unsigned short uwords[MEMORY_SIZE_IN_WORDS];
       unsigned long lwords[MEMORY_SIZE_IN_WORDS];
   } memory;
   ```

   D.
   ```
   static struct {
       word_type words[MEMORY_SIZE_IN_WORDS];
       unsigned short uwords[MEMORY_SIZE_IN_WORDS];
       unsigned long lwords[MEMORY_SIZE_IN_WORDS];
   } memory;
   ```

6. (10 points) How does an assembly language program that only contains a single NOP instruction (and no other instructions) execute in the Simple Stack Machine VM? (Circle the **one** correct answer's letter.)

    A. Since the program has no EXIT instruction, the VM will keep on interpreting each word as an instruction, until the PC's value is large enough value to violate an invariant.

    B. The VM does nothing, as a NOP instruction does nothing. This is to be expected, as most ISAs contain a NOP instruction that is used when nothing should be done; such an instruction is often used as a placeholder.

    C. It causes the VM to shrug its shoulders, as the NOP instruction does not specify anything to do.

    D. Since the program has no EXIT instruction, the program will not do anything and will feel sad about not being able to do anything.

7. (10 points) Select all of the following that are valid reasons for using several different files is useful in a C program. (Circle **each** answer letter that is correct.)

    A. Using several different files allows a group to more easily work together on a large project.

    B. Using several different files allows design decisions made in one file's implementation to be hidden in that file, improving maintenance.

    C. Using several different files creates more work for compiler writers, improving job prospects and salary for Computer Science graduates.

    D. Using several different files allows design decisions made in one file's implementation to be hidden in that file, improving maintenance.

    E. Using several different files allows each file to be compiled separately, which will shorten the time needed to build an entire program after some files are already built.

8. (10 points) What would be a straightforward implementation of the registers in the SSM that make a clear connection between the code of the VM and the behavior as specified in the SSM Manual. (Circle the **one** correct answer's letter.)

    A. Implement the registers using an arrays of integers.

    B. Implement the registers using an array of floating point numbers.

    C. Implement the registers using several separate character variables.

    D. Implement the registers by using the UCF Registrar, who is a person with an office in Millican Hall.

9. (10 points) In the SSM VM's implementation, how does the (main) program check to see if it is called with the `-p` option? (Circle the **one** correct answer's letter.)

    A. It checks that it was passed an array containing 3 strings, and that the string at index 1 is the string `-p`.

    B. It checks that it was called with a text message telling it to print the program, and that text message is interpreted by a large language model, like ChatGPT, which then prints the program.

    C. The VM always prints the program that is in the BOF file passed to it as an argument.

    D. The main program consists of two functions, one of which is called when the user wants to use the `-p` option, which does the printing, and the operating system (OS) knows which function to call based on what the user wants done.

10. (10 points) In the SSM VM, what determines the initial value stored in the stack pointer (`$sp`) register when the program starts running? (Circle the **one** correct answer's letter.)

    A. It is always 4096.

    B. It is the value of the `stack_bottom_addr` field of the binary object file's header.

    C. It is the smallest power of two that is at least twice as large as the value of the `text_length` field of the binary object file's header.

    D. It is the smallest power of two that is at least twice as large as last the value of the sum of the `data_start_addr` field and the `data_length` field in the binary object file's header.