Name: CS520

Midterm Exam

7 March 2018, 80 minutes, 21 questions, 100 points

The exam is closed book and notes.
Please keep all electronic devices turned off and out of reach.
Note that a question may require *multiple* checked boxes for a correct answer. Checking *some* but not *all* of the required boxes will result in a *partial* answer worth only 2 of the 5 points. Checking any box that shouldn't be checked results in an *incorrect* answer, worth zero.

1. [0 pts]
   - ◯ Hold my exam in your office. I will pick it up prior to May 11.
   - ◯ Shred my exam. I never want to see it again.
   - √ Return my exam to my Kingsbury mailbox.

2. How would the following C string (i.e. null terminated) `"210"` be represented using ASCII in the memory of a Big Endian machine with a byte-addressable memory? The bytes are shown below in increasing memory address order, left to right. The ASCII code for `'0'` is `0x30`. The ASCII code for `'1'` is `0x31`. The ASCII code for `'2'` is `0x32`. [5 pts]
   - ◯ `0x00 0x32 0x31 0x30`.
   - ◯ `0x30 0x31 0x32`.
   - ◯ `0x32 0x31 0x30`.
   - ◯ `0x30 0x31 0x32 0x00`.
   - √ none of the above.

3. Interpret `0xC0` and `0x11` as 8-bit two's complement integers and add them together to produce an 8-bit two's complement integer. The result is: [5 pts]
   - √ `0xD1`.
   - ◯ `0xAF`.
   - ◯ `0x47`.
   - ◯ `0x2F`.
   - ◯ none of the above.

4. Consider the following C function: [5 pts]

```
signed int f(void)
{
  int i = -257;
  return *(signed char *) &i;
}
```

   On a machine with a byte-addressable memory, and with 32-bit integers, the function will:
   - ◯ always return 0.
   - √ always return -1.
   - ◯ return -1 if the machine is little-endian and 0 otherwise.
   - ◯ return 1 if the machine is little-endian and 0 otherwise.
   - ◯ none of the above.

5. Which of the following 16-bit two's complement values will overflow when they are moved to an 8-bit two's complement container? [5 pts]

   ◯ `0xFFAA`.

   ◯ `0x000F`.

   √ `0x00A7`.

   √ `0x808A`.

6. What would -23 look like as a 32-bit two's complement integer in the memory of little-endian machine? The bytes are shown below in increasing memory address order, left to right. [5 pts]

   √ `0xE9 0xFF 0xFF 0xFF`.

   ◯ `0xE8 0xFF 0xFF 0xFF`.

   ◯ `0x80 0x00 0x00 0x17`.

   ◯ `0xFF 0xFF 0xFF 0xE8`.

   ◯ `0x17 0x00 0x00 0x80`.

   ◯ none of the above.

7. Which of the following statements about IEEE single-precision floating-point are true? [5 pts]

   ◯ It stores the exponent as a two's complement value.

   √ There are two infinity values, positive infinity and negative infinity.

   ◯ Denormalized values have a stored exponent with all 1 bits and a stored significand that is not all 1 bits.

   ◯ NaN values have a stored exponent with all 1 bits and a stored significand that is all 0 bits.

8. Interpret `0x41540000` as IEEE single-precision floating-point. What is its decimal value? [5 pts]

   ◯ NaN.

   ◯ `+26.5`.

   ◯ `+53.0`.

   ◯ positive infinity.

   √ `+13.25`.

   ◯ none of the above.

9. Interpret `0xF0811111` and `0xF1822222` as IEEE single-precision floating-point values. Which of the following statements are true? [5 pts]

   √ `0xF0811111` is greater than `0xF1822222`

   √ They are both negative values.

   ◯ They are both denormalized values.

   ◯ They are both NaN values.

10. Interpret `0x42A00001` and `0x40A00002` as IEEE single-precision floating-point. Add the two values together. What is the result? [5 pts]

    ◯ `0x42AA0000`.

    ◯ `0x42A00003`.

    ◯ `0x40A00003`.

    √ `0x42AA0001`.

    ◯ none of the above.

11. Interpret `0x80444444` as an IEEE single-precision floating point value. Convert it to IEEE double-precision floating point. What is the result? [5 pts]

- √ `0xB801111100000000`.
- ◯ `0xB818888880000000`.
- ◯ `0x8000000000000000`.
- ◯ `0x8008888880000000`.
- ◯ none of the above.

12. Represent this UTF-16 value, `0x3333`, in UTF-8. What is the result? [5 pts]

- ◯ `0x33 0x33`.
- ◯ `0xEC 0xB3 0x8C`.
- √ `0xE3 0x8C 0xB3`.
- ◯ `0xE0 0x83 0x8C 0xB3`.
- ◯ none of the above.

13. Represent this UTF-32 value, `0x00020202`, in UTF-16. What is the result? [5 pts]

- ◯ `0x0002 0x0202`.
- ◯ `0xD880 0xDE02`.
- √ `0xD840 0xDE02`
- ◯ `0xD842 0xDC22`.
- ◯ none of the above.

14. Which of the following statements are true? [5 pts]

- ◯ The UTF-16 encoding of an ASCII character is always one byte long.
- √ The UTF-8 encoding of an ASCII character is always one byte long.
- ◯ The UTF-8 encoding of a Unicode character is always shorter than the UTF-16 encoding of the character.
- ◯ The UTF-8 encoding of a Unicode character is always shorter than the UTF-32 encoding of the character.

15. Which of the following UTF-8 sequences contain at least one error? [5 pts]

- ◯ `0xF1 0x80 0x80 0x81`.
- √ `0xF0 0x80 0x81`.
- √ `0xE0 0x81`.
- √ `0x81`.

16. Which of the following statements about an ELF file are true? [5 pts]

- ◯ The .text section contains the names of the sections.
- ◯ All ELF files are the same length.
- √ An ELF file starts with an ELF header that includes a field that identifies whether the file is in Big Endian or Little Endian format.
- ◯ The section header array begins immediately after the ELF header.

17. Consider the following RISC-V program: [5 pts]

```
top:
  sbreak
  sbreak
  beq x9,x5,top
```

What is the encoding of the `beq` instruction?

- ◯ `0xFE4186E3`.
- ◯ `0xFE541863`.
- √ `0xFE548CE3`.
- ◯ `0xFE928CE3`.
- ◯ none of the above.

18. Which of the following statements about linking are true? [5 pts]

- √ If a symbol is an outsymbol in one input file to a linker, and it does not have a matching input symbol in another input file, then it is an outsymbol in the output file of the linker.
- √ A linker combines input object files into a single output object file.
- √ If a symbol is an insymbol in both input files to a linker, then it is duplicate symbol error.
- ◯ If a symbol is an outsymbol in both input files to a linker, then it is duplicate symbol error.

19. Decode the RISC-V instruction `0x00100073`. How would this instruction be displayed by the Program 3 disassembler? [5 pts]

- ◯ `fence`.
- ◯ `addi x1,x0,73`.
- √ `sbreak`.
- ◯ `addi x1,x0,115`.
- ◯ none of the above.

20. Which of the following statements about PC-relative addresses are true? [5 pts]

- √ The RISC-V machine uses PC-relative addresses in its branch instructions.
- √ At execution time they are added to the program counter to get the actual address.
- ◯ They can contain only positive values.
- ◯ The PC in PC-relative stands for "politically correct."

21. Which of the following statements are true? [5 pts]

- √ An assembler has two passes because a reference to a label may come before its definition.
- ◯ The first pass of an assembler only exists to determine the length of the input file.
- √ An assembler reads an assembly language source file as input and writes an object file as output.
- √ The second pass of an assembler determines the address of a referenced label by looking it up in a symbol table created by the first pass of the assembler.