CS520 Spring 2014 Final Exam

- **This exam has a total of 100 points. There are twenty questions, which are all worth 5 points each.**
- **Answer the questions on separate paper.**
- **For partial credit show your work.**
- **The exam is closed book and notes.**
- **Please keep all electronic devices turned off and out of reach.**
- **When turning in your exam, please fold your papers in half length-wise and write your name on the outside.**
- **I will return your exam to your Kingsbury mailbox unless you tell me not to.**

1. (5 points) Write a C function called `isNaN` that will test a 32-bit IEEE single-precision floating-point value and will return 1 if it is a NaN and return 0 otherwise. The function should only use integer operations. The function should take one `unsigned int` argument. The argument will contain 32 bits that should be interpreted as a floating-point value.

2. (5 points) Why did you need to use a mutex and a condition variable to implement an event loop? How exactly were each of these used?

3. (5 points) Decode the following UTF-8 sequence (shown as hexadecimal) to a Unicode character: `0xEA 0xB3 0x9C`. Show the Unicode character as hexadecimal.

4. (5 points) Write an Intel 64 assembly language function called `storeLong` that will take two arguments: a 64-bit address as the first argument and a 64-bit integer value as the second argument. The function should store the integer value given by the second argument at the address given by the first argument. The function should return the integer value given by the second argument as its return value.

5. (5 points) Write a C function called `isLittleEndian` that will return 1 if the machine executing the function is Little Endian and 0 otherwise.

6. (5 points) In an implementation of a threads package, how would the `condSignal` function be implemented? Provide pseudo code for the function. Also explain what fields are stored in the memory allocated to represent a condition variable. You may assume that the threads package will run on only a single processor.

7. (5 points) Add together the following two IEEE single-precision floating-point values: BF800003 and 3F800003. Show the IEEE single-precision floating-point result in hexadecimal.

8. (5 points) Encode the `load` instruction in the following vm520 assembly language fragment:

```
answer:
     word  1  # this directive allocates one word in memory
     alloc 10 # this directive allocates ten words in memory
skip:
     load r4, answer
```

Show your answer in hexadecimal and show all the hex digits, even if they are zero.

The opcode for the `load` instruction is `0x01`. The format for the `load` instruction has the opcode in the low eight bits, the register in the next four bits, and the PC-relative address in the upper twenty bits.

9. (5 points) In our implementation of an exception mechanism for C programs, explain exactly what the `catchException` function did.

10. (5 points) Why is a virtual memory system similar to a fully associative memory cache?

11. (5 points) What are the **roots** of a garbage collection?

12. (5 points) Von Neumann's design for the IAS computer relied on self-modifying code to do array indexing. Explain.

13. (5 points) Consider how the following two C loops would be accessed by a virtual memory system with 64 virtual pages, 8 physical pages, a page size of 4 ints, 4 TLB entries, and LRU replacement of physical pages and TLB entries:

```
for (i = 0; i < 256; i++)
  a[i] = i;
sum = 0;
for (i = 0; i < 256; i++)
  sum += a[i];
```

Assume only the array is in the virtual memory and the first word of the array is at address zero. The first 8 virtual pages are loaded into the 8 physical pages at the initialization of the virtual memory system. The TLB is initialized to have the VM to PM mapping for the first 4 virtual pages loaded into physical pages. How many page faults would there be? How many TLB misses? How many disk writes?

14. (5 points) Show how -37 (base 10) would be represented in memory as a 16-bit 2's complement integer. Show your answer in hexadecimal and show all the hex digits, even if they are zero. Clearly label the order in which the bytes would lie in memory. Assume the machine is Little Endian.

15. (5 points) Write a C declaration for a variable called `fp` that contains a pointer to a function that takes two arguments of type `long` and that returns a `long`.

16. (5 points) Convert the UTF-16 sequence (shown in hexadecimal) `0xD801 0xDE60` to UTF-32. Show your answer in hexadecimal. Be sure to clearly indicate how many UTF-32 characters are in the result.

17. (5 points) Describe in POSIX threads pseudo-code how to implement the producer-consumer pattern when there are multiple producers and multiple consumers. Assume the producers are placing values in an array and the consumers are removing values from the array. You do not need to worry about termination.

18. (5 points) What information must be in an object file to support linking? Assume the file contains only PC-relative addresses.

19. (5 points) Why does the TLB exist? What problem does it solve? Where is it located?

20. (5 points) Add together the following two 8-bit 2's complement integers (shown in hexadecimal): `0x07` and `0xB1`. Show the 8-bit 2's complement answer in hexadecimal and show all the hex digits, even if they are zero. Does the answer overflow? Please indicate either yes or no. Also check your answer using decimal: What decimal values do the two input values represent? What is the answer in decimal?