Name: ______________________________

Directions: Work only on this sheet (on both sides, if needed); do not turn in any supplementary sheets of paper. There is actually plenty of room for your answers, as long as you organize yourself BEFORE starting writing. In order to get full credit, SHOW YOUR WORK.

1. (5) Suppose we have an Intel assembly language subroutine which is called from C. Give a single Intel assembly language instruction which will copy the third argument to the EDX register. Assume that the subroutine does not modify the stack at all.

2. (5) Fill in the blank with a term from our course: *Cache memory* is to *virtual memory* as *cache miss* is to __________________.

3. (10) Fill in the blanks with numbers: One difference between the Intel and MIPS CPUs is that during the fetch, decode and execution of an instruction on MIPS, the PC always is incremented by __________________. In the JVM, the corresponding value is usually __________________.

4. (10) Give an Intel assembly language instruction which is analogous to the MIPS instruction

```
sw $10,200($15)
```

5. (10) Consider the subroutines sys_write, sys_read and sys_ioctl in the Tetris game source code. Fill in the blank with a UNIX file name: To get the numbers 3, 4 and 54 in these macros, the author of the code consulted ________________________.

6. (10) When an interrupt occurs on a PC, how many different values will go through the MDR? Exclude from your count memory accesses from the interrupted instruction and from the first instruction in the interrupt service routine. In answer, give both the correct count of values and and a narrative description of what those values represent.

7. Suppose we have one RISC and one CISC machine, and on each one the cache is split into separate data and instruction caches. We have a set of C source files which we compile on each machine, and we run the two sets of executables on both machines. We are concerned with the **miss rates**, i.e. the percentage of accesses of the given type (data access, instruction fetch) which cause cache misses, on the two machines.

(a) (5) Choose the best answer regarding the data miss rate: (i) The miss rate for CISC will probably be higher than for RISC; (ii) the miss rate for RISC will probably be higher than for CISC; (iii) there is no particular reason for either one to be higher.

(b) (5) Choose the best answer regarding the instruction miss rate: (i) The miss rate for CISC will probably be higher than for RISC; (ii) the miss rate for RISC will probably be higher than for CISC; (iii) there is no particular reason for either one to be higher.

8. Look at the JVM assembly language for the function Min, consisting of 10 instructions.

(a) (10) Suppose the function were to determine the maximum of the arguments, rather than the minimum. (This change does not apply to parts (b) and (c) below.) Show, if possible, how to change one or more of the 10 instructions, without adding or deleting any of the 14, in order to make the function find the maximum instead of the minimum. If it is impossible, state clearly what the obstacle is.

(b) (10) Suppose the instruction in offset 2 had been

```
if_icmpge 0
```

(Again, this change does not apply to parts (a) and (c).) Show in hex what the JVM machine code would be for this instruction.

(c) (10) (The following assumptions apply only to this part.) Suppose this assembly language subroutine had been written by hand, instead of being generated by the compiler. Suppose also that the author of the code had forgotten to include the instructions in offsets 6, 11 and 12. (Adjust the rest of the code accordingly: The **if_icmpge** would now have 9 as its target; the **goto** would now be in offset 6 and its target would be 10; etc.) What ill effects, if any, would this have on the correct operation of Min()? State clearly what would go wrong, or if there would be no ill effects, state clearly why not.

9. (10) Consider the Katevenis quote on p.14 of the RISC handout. Suppose that a more efficient optimizer is developed, so that the figure "40% to 60%" now becomes 80%. What would the percentage of lost cycles now be?

Solutions:

1.

```
movl 12(%especially),%edx
```

(Note that since the subroutine is not written in C, there would be no code using EBP, etc.)

2. Page fault.

3. 4,1

4.

```
molv %eax,200(%ebx)
```

5.

```
/usr/include/asm/unistd.h
```

6. 6: Flags register; CS; PC; device number, i, from 8259A; 2 words from c(IDT)+8i.

7a. (iii)

7b. (ii) (due to larger code size)

8a.

```
0  iload_1
1  iload_0
```

8b. Jump distance is -2 = 0xfffe, so instruction is 0xa2fffe.

8c. The code would still work fine. It would not set the value of T, but that would not affect correct operation; the proper value will still be on the top of the stack when the return instruction is executed.

9. 0.1 (10%) + 0.1 (20%) = 3%