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. In fill-in questions, the number of blanks shown has no relation to the number of characters in the answer. Note that earlier questions tend to be easier.

1. (5) Look at the instruction JNS NXT on p.83 of Neveln. What would the machine code 0F8904000000 change to if the instruction were changed to JS GCD? (Note that both the operation and operand are changed.)

3. (5) The –gstabs option of **as** is similar to what option of **gcc**?

4. (10) Fill in the blank: One aspect of the Intel hardware which we mentioned in class could be accessed via assembly language but not via C is _____.

5. (10) Choose the correct statement: If in Prog. 6.2 we were to use JC (or JNC) instead of ADC, the code would be (i) faster and larger, (ii) faster and smaller, (iii) slower and smaller, (iv) slower and larger,

6. (10) State the two reasons discussed in class as to why jump instructions typically specify the location of the place to be jumped to in terms of a distance rather than an absolute address.

7. (10) An example of an instruction discussed in class as being present in the Intel CPUs but ironically not in the more modern SPARC is _____.

8. (10) When multiplying two 16-bit quantities, MUL places the product in DX:AX. It would have been more natural to have the product go into EAX or some other 32-bit register. Why was this not done?

9. (10) Write assembly code, using AT&T syntax (this is required), which will replace the current contents of EAX by its absolute value. For instance, if EAX currently contains -12, that value will be changed to +12. (Note: Ideally, your code should not be longer than, say, 5 or 6 lines; feel free to copy the contents of EAX to other registers, say EBX, but don't use memory; I am just asking for a program fragment here, not a whole source code with lines line .text etc.)

10. Below is the output obtained when **as** was applied with the -a option to a certain source file. Assume that the .data and .text segments are loaded into memory at the locations indicated above. Assume there is no instruction prefetching.

- (a) (5) In line 20, if the 23 in the source code had been 28, what would the machine code BA17000000 change to?
- (b) (10) Consider the third iteration of the loop. At the very end of the execution of the instruction in line 23, state which values will be in the PC, MAR and MDR.
- (c) 10) Answer (b) for line 22.

```
1
2                       # calculate the first 25 Fibonacci numbers
3
4                       .data
5
6                       .globl fib
7 0000 01000000         fib:  .long 1
8 0004 01000000               .long 1
```

```
 9                          .rept 23
10                          .long 0
11                          .endr
12 0008 00000000
12      00000000
12      00000000
12      00000000
12      00000000
13                      .text
14
15                      .globl _start
16                      _start:
17 0000 BB010000            movl $1, %ebx  # EBX will be fib_i-2
17      00
18 0005 B9010000            movl $1, %ecx   # ECX will be fib_i-1
18      00
19 000a B8080000            movl $fib+8, %eax  # EAX will point to fib_i
19      00
20 000f BA170000            movl $23, %edx  # loop counter
20      00
21 0014 89CE            top:  movl %ecx, %esi  # make a copy of old fib_i-1
22 0016 01D9                  addl %ebx, %ecx  # calculate fib_i
23 0018 8908                  movl %ecx, (%eax)  # store fib_i
24 001a 89F3                  movl %esi, %ebx  # old fib_i-1 is new fib_i-2
25 001c 83C004                addl $4, %eax
26 001f 4A                    decl %edx  # decrement loop counter
27 0020 75F2                  jnz top
28 0022 BE000000        done: movl $0, %esi  # dummy for breakpoint
28      00
```

Solutions:

1. 0F880B000000

3. -g

4. The flags.

5. (iv)

6. Shorter instructions, position-independent code.

7. MUL

8. Backwards compatibility.

9.

```
movl $0,%ebx
subl %eax,%ebx
jns eaxneg
jmp done
movl %ebx,%eax
```

```
eaxneg: movl %ebx, %eax
done:
```

Many other solutions are possible.

10.a.

BA1C000000

10.b.

PC: 1A, MAR: 10, MDR: 5

10.c.

PC:18, MAR: 16 (or 17), MDR: 1D9 (or D9)