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.

Note: In all problems on this exam, assume that all bits which are not initialized by the programmer are set to 0.

1. Suppose our **.data** segment begins with

```
u: .byte 11
   .byte 12
   .byte 13
   .byte 14
v:  .long 8
```

Consider the instruction

```
movl $v, %edx
```

(a) (10) Using official language from our course, state the addressing modes for the source and destination operands in this instruction.

(b) (15) What machine language would the assembler produce from this instruction?

2. Consider the JVM assembly language code for **Minimum.class** in our notes. Parts (a) and (b) are independent of each other. Do not consider the assumptions in one to apply to the other.

(a) (5) Suppose, due to a compiler error, the instruction at offset 27 of **main()** had been generated as

```
istore_1
```

and we were to run

```
java Minimum 12 15
java Minimum 28 25
```

What output would result?

(b) (10) Suppose the first statement in **Min()**, just before the **if**, had been:

```
U = U + V;
```

Show the resulting first four assembly language instructions for **Min()**.

3. Answer either H (hardware), O (OS), B (both) or N (neither):

(a) (5) What writes to the PTR?

(b) (5) What reads from the PTR?

(c) (5) What writes to the page table?

(d) (5) What reads from the page table?

4. (15) Consider the **addone()** example in our MIPS unit, but with **addone()** being replaced by **addsome()**. The latter has a second argument which is the amount to add to the variable pointed to by the first. In other words,

```
int x,n;

main()

{  x = 7; n = 5;
   addsome(&x,n);
   printf("%d\n",x);
}
```

would print out 12. Fill in the missing code (at most two instructions) in this excerpt of the output of **gcc -S**:

```
# addsome(&x,n)
la     $4,x
# missing code goes here
la     $25,addsome
jal    $31,$25
```

5. (15) Look at the OS code excerpt on p.8 of the OS unit. Suppose the designers of the OS had a field named **pc** in the TSS **struct**s, 300 bytes past the start of the TSS. Suppose also that they wished to record in the TSS the value of the PC at the time of the interrupt. Fill in the blanks of the following code to do this recording, just before the first **movl** on p.8:

```
movl ______, %ebp
movl ______(%ebp), %ebp
movl _____, 300(_____)
```

6. (10) Suppose we get a seg fault at the declaration of a local variable:

```
int x;
```

There are no other locals. Show the likely assembly language instruction which triggered the error.

Solutions:

1.a. immediate, register

1.b. 0xba04000000

2.a. Y would never be set, leaving it as 0. Both output values would be 0.

2.b.

```
iload_0
iload_1
iadd
istore_0
```

3. O, B, O, B (partial credit was given for H in (b))

4.

```
la $5, n
lw $5, 0($5)
```

5. %esp, 12, %ebp, %ebx

6. (See the example in our subroutines unit.)

```
movl $0, -4(%ebp)
```