Login name _________________ **Name** ______________________

Quiz 4
CSE 131B
Winter 2003

Signature ___________________ **Student ID** __________________

1. State which calling convention / parameter passing mode is being used and what gets printed:

C

```
int x = 911;

void foo1( int a, int *b ) {        Parameter passing mode for a ________________________
  a = 75;
  *b = 99;                          Parameter passing mode for b ________________________
}

int main( void ) {
  int y = 404;

  foo1( x, &y );

  printf( "x = %d; y = %d\n", x, y );          Output: x = _______; y = _______

  return 0;
}
```

Oberon

```
VAR x : INTEGER;
VAR y : INTEGER;

PROCEDURE foo1 ( VAR a : INTEGER; b : INTEGER );
BEGIN
    a := 75;                        Parameter passing mode for a ________________________
    b := 99;
END foo1;                           Parameter passing mode for b ________________________

BEGIN
    x := 911; y := 404;

    foo1( x, y );
    OUTPUT "x = ", x, "; y = ", y, "\n";        Output: x = _______; y = _______
END.
```

Fill in the blanks of the equivalent C program to simulate the above Oberon parameter passing modes (that exposes what the compiler is actually doing to implement these parameter passing modes):

```
int x, y;

void foo1(_____________ a, _____________ b ) {
  ________ = 75;
  ________ = 99;
}

int main( void ) {
  x = 911; y = 404;

  foo1( _____________ , _____________ );
  printf( "x = %d; y = %d\n", x, y );
  return 0;
}
```

C#

```
using System;

class test7 {
   static int x = 911;

   public static void foo7( out int a ) {
     a = 75;

     Console.WriteLine( "x = " + x );
   }

   public static void Main() {

     foo7( out x );

     Console.WriteLine( "x = " + x );
   }
}
```

Examine the above C# program. C# (along with Ada) supports **out** parameters. We discussed two different implementations the compiler might use to support out parameters.

If a particular implementation of a C# compiler used call by reference to implement out parameters, what would the above program output?

```
Output:      x = ________

             x = ________
```

If a particular implementation of a C# compiler used call by result to implement out parameters, what would the above program output?

```
Output:      x = ________

             x = ________
```

The SPARC architecture (like most RISC architectures) does not have a multiply/divide/modulus instruction. Instead the SPARC architecture has **.mul** (multiplication), **.div** (division), and **.rem** (remainder/modulus) leaf subroutines defined.

Write the unoptimized SPARC assembly instructions to perform the following integer operation saving the result in local register %l4 (assume we have variable **x** mapped to local register %l4):

```
x = 55 * 7777;
```