Login name _____ **Name** _____

Signature _____ **Student ID** _____

Final
CSE 131B
Winter 2003

Page 1	**_____ (20 points)**
Page 2	**_____ (25 points)**
Page 3	**_____ (21 points)**
Page 4	**_____ (40 points)**
Page 5	**_____ (30 points)**
Page 6	**_____ (25 points)**
Page 7	**_____ (25 points)**
Page 8	**_____ (14 points)**
Subtotal	**_____(200 points)**
Page 9 Extra Credit	**_____ (10 points)**
Total	**_____**

1. Consider the following psuedocode:

```
x : integer;            -- global

procedure set_x ( n : integer )
      x := n;

procedure print_x()
      output( x );      -- print the value of x

procedure one()
      x : integer;
      set_x( 1 );
      print_x();

procedure two()
      set_x( 2 );
      print_x();

set_x( 0 );
one();
print_x();
two();
print_x();
```

What does the program output if the language uses <u>static</u> scoping? (4 points)

_____
_____
_____
_____

What does the program output if the language uses <u>dynamic</u> scoping? (4 points)

_____
_____
_____
_____

Give an example of a converting type cast/conversion (underlying bit pattern needs to be changed). (4 points)

Give an example of an implicit type coercion (type conversion without an explicit cast). (4 points)

Give an example of a non-converting type cast/conversion (underlying bit pattern does not change). (4 points)

2. Consider the following pseudocode:

```
TYPE A = INTEGER;
TYPE B = POINTER TO A;

VAR a, b  : A;
VAR c     : INTEGER;
VAR d     : B;
VAR e     : POINTER TO INTEGER;
VAR f     : B;
```

(15 points spread out over the following three questions.)

Which variables are considered equivalent under strict name equivalence?

_____ group 1 _____ group 2 (opt) _____ group 3 (opt) _____ group 4 (opt)

Which variables are considered equivalent under loose name equivalence?

_____ group 1 _____ group 2 (opt) _____ group 3 (opt) _____ group 4 (opt)

Which variables are considered equivalent under structural equivalence?

_____ group 1 _____ group 2 (opt) _____ group 3 (opt) _____ group 4 (opt)

Using the Right-Left rule write the definition of a variable named XXX that is a pointer to a function that take a pointer to an double as the single parameter and returns a pointer to an array of 9 elements where each element is a pointer to a struct Pub. (6 points)

Given the array declaration

C

```
int a[2][3];
```

Oberon-like

```
VAR a : ARRAY 2, 3 OF INTEGER
```

Mark with an **A** the memory locations where we would find (4 points)

```
      a[1][2]                          a[1,2]
a:
```

```
low memory                                                    high memory
```

3. What major issue distinguishes a macro compared to an inline function? (4 points)

What gets printed? (9 points)

```
VAR a : INTEGER;

PROCEDURE foo1( VAR x : INTEGER );
BEGIN
  x := 77;

  OUTPUT a;              ____________________
END foo1;

PROCEDURE foo2( VAR y : INTEGER );
BEGIN
  y := 66;

  OUTPUT a;              ____________________

  foo1( y );

END foo2;

BEGIN
  a := 55;

  foo2( a );

  OUTPUT a;              ____________________
END.
```

How does a leaf subroutine differ from a traditional closed subroutine, specifically in the SPARC arch.? List 2 ways they differ. (8 points)

1.

2.

4. Identify <u>where</u> each of the following program parts live in the Java runtime environment as discussed in class. (24 points)

```
public class Foo {
  private Foo a;

  private static int b;

  public Foo() {

    a = this;
    ++b;
  }

  public static void main( String[] args ) {

    int c = 5;

    Foo d;

    d = new Foo();
    d.method( c );
  }

  private void method( int e ) {

    int f;
    f = e;
  }
}
```

a _____

b _____

Foo() _____

this _____

main() _____

args _____

c _____

d _____

where d is pointing _____

method() _____

e _____

f _____

Assume there is a Java class named Fubar that defines

static method s_foo() that returns an int	`static int s_foo() { ... }`
static variable sv that is of type int	
instance method i_foo() that returns an int	`int i_foo() { ... }`
instance variable iv that is of type int	
instance variable ref that is a reference to a Fubar object and initialized	`Fubar ref = new Fubar();`

State whether the following initializations are legal (no compiler error)? <u>Explain why or why not</u>. (16 points)

```
private static int sv = Fubar.s_foo();
```

```
private static int sv = ref.i_foo();
```

```
private int iv = ref.i_foo();
```

```
private int iv = ref.s_foo();
```

5. Given the following program, order the printf() lines so that the values that are printed when run on a Sun SPARC Unix system are displayed from smallest value to largest value. (20 points)

```
void foo( int, int );   /* Function Prototype */

int a = 911;

int main( int argc, char *argv[] ) {

        int b = 420;                                                    prints
        int c;                                                  _____ smallest value

        foo( argc, b );                                         _____

/*  1 */ (void) printf( "argc --> %p\n", &argc );              _____
/*  2 */ (void) printf( "foo --> %p\n", foo );
/*  3 */ (void) printf( "malloc --> %p\n", malloc(50) );       _____
/*  4 */ (void) printf( "b --> %p\n", &b );
/*  5 */ (void) printf( "a --> %p\n", &a );                    _____
/*  6 */ (void) printf( "c --> %p\n", &c );
}                                                               _____

void foo( int d, int e ) {                                      _____

        int f = 404;                                            _____
        static int g;

                                                                _____
/*  7 */ (void) printf( "d --> %p\n", &d );                       prints
/*  8 */ (void) printf( "f --> %p\n", &f );                    _____ largest value
/*  9 */ (void) printf( "e --> %p\n", &e );
/* 10 */ (void) printf( "g --> %p\n", &g );
}
```

In the following Java and C/C++ programs, which version will generally be faster / require less space if we create several instances of a Foo object (or in the C/C++ version we call function foo() several times) and access variable **a** many times? Explain why (time and space) for both the Java and the C/C++ versions. (10 pts)

Java Version 1

```
public class Foo {
  static int[][] a = new int[100][100];
   ...
}
```

Java Version 2

```
public class Foo {
  int[][] a = new int[100][100];
   ...
}
```

C/C++ Version 1

```
void foo() {
  int a[100][100] = {0};
   ...
}
```

C/C++ Version 2

```
void foo() {
  static int a[100][100];
   ...
}
```

6. Given the following Oberon program, emit the **unoptimized** SPARC assembly language code that should be generated for this program. Assume the global variables x, y, and z are allocated in the Data segment (given). Assume no optimizations – treat each instruction separately without any knowledge of any previously computed/loaded/stored values that may still be in a register from a previous instruction. Draw a line between each group of assembly language instructions that represent the emitted code generated for each instruction and label them with the instruction number. (25 points)

```
VAR x, y, z : INTEGER;

PROCEDURE foo( a : INTEGER; VAR b : INTEGER ) : INTEGER;
  VAR i, j : INTEGER;        (**** Local Stack variables – Do Not Need to Initialize to 0. ****)
BEGIN
  i := a + 5;           (* 1 *)
  j := b - a;           (* 2 *)
  b := i + 7;           (* 3 *)
  RETURN b;             (* 4 *)
END foo;

BEGIN
  INPUT x;              (* 5 *)
  INPUT y;              (* 6 *)
  z := foo( x, y);      (* 7 *)
END.
```

```
        .global foo, main

        .section ".data"
        .align 4
x:      .word 0
y:      .word 0
z:      .word 0

        .section ".text"
foo:
        save    %sp, -(92 + 8) & -8, %sp
```

```
main:
      save    %sp, -96, %sp
```

7. Given the following pseudocode

```
read n;

for ( i = 0; i < n; ++i ) {
   a[i] = n * i;

   if ( n > 20 )
      b[i] = a[i];
}
```

the `if (n > 20)` is a loop invariant. Restructure this code to move it out of the loop to save up to 3 instructions (cmp, conditional branch, nop) per loop iteration? Also perform any other explicit code improvements such as redundant loads/stores (unnecessary memory accesses) and strength reduction if possible. No assembly – just all high-level code like the above. You may add/change code as part of your code improvements. The idea is the resulting code is faster to execute for any arbitrary value of **n** read in at runtime. (15 points)

```
read n;
```

Consider the following code:

```
ld   [r1], r2
add  r1, 20, r1
ld   [r1], r3     ! Destination of this load is a source operand in next instr.
------            ! Stall 1 cycle (L1 cache hit) while load into r3 completes.
add  r2, r3, r4
```

Show how to shorten the time required for this code by moving the update of r1 forward into the delay slot of the second load. Assume r1 is still live (needed) at the end of this code. Make whatever other alterations to individual instructions to maintain correctness. The idea is to reduce this chunk of code from 5 cycles to 4 cycles. No assembly – just this virtual register pseudo-assembly code like the above. (10 points)

8. Consider the following code:

```
r5 = r2 x r4      ! Assume general multiply takes 5 instruction cycles.
-----             ! Leave the multiply alone.
-----
-----
-----
r6 = r5 + r1
r1 = r1 + 20
```

Show how to shorten the time required for this code by moving the update of r1 backward into one of the delay slots of the multiply. Assume all the registers used here are still live (needed) at the end of this code. Make whatever other alterations to individual instructions or additional/new instructions in the delay slots to maintain correctness. You may use other registers not used here. The idea is to reduce this chunk of code from 7 cycles to 6 cycles. No assembly – just this virtual register pseudo-assembly code like the above. (10 points)

Why do computer programmers confuse Halloween with Christmas? (2 point)

Tell me something you learned in this class that is extremely valuable and that you think you will be able to use for the rest of your programming/computer science career. (2 point)

9. Extra Credit (10 points)

What is the value of each of the following expressions?

```
char *a = "End this, please!";          /*  char a[] = "End this, please!";  */
```

```
"I loved Compilers B!"[6]   _____________

a[1]                        _____________

*a                          _____________

*(a+12)                     _____________

*&a[5]                      _____________

0["This Blows Me Away!"]    _____________
```

Given the following ANSI/ISO C variable definitions, identify which expressions will produce a static semantic compiler error. Hint: Think modifiable l-value.

A) No compiler error
B) Compiler error

```
int i = 5;
float f = 1.5;
int *iPtr = &i;
float *fPtr = &f;

*iPtr = (int) *fPtr;      ______

(float *) iPtr = fPtr;    ______

fPtr = &(i + f);          ______

++( (float *) iPtr );     ______
```

Scratch Paper