Final
CSE 131
Winter 2008

Page 1  _________ (24 points)
Page 2  _________ (28 points)
Page 3  _________ (37 points)
Page 4  _________ (24 points)
Page 5  _________ (9 points)
Page 6  _________ (12 points)
Page 7  _________ (12 points)
Page 8  _________ (40 points)

Subtotal  _________ (186 points)

Page 9  _________ (12 points)

Extra Credit

Total  _________
1. Given the following CUP grammar snippet (assuming all other Lexing and terminals are correct):

Stmt ::= Designator T_ASSIGN {: System.out.println("A"); :} Expr {: System.out.println("B"); :} T_SEMI
| Expr {: System.out.println("C"); :} AddOp {: System.out.println("D"); :} Designator
| Designator {: System.out.println("E"); :}

Expr ::= Expr {: System.out.println("C"); :} AddOp {: System.out.println("D"); :} Designator
| Designator {: System.out.println("E"); :}

Designator ::= T_ID {: System.out.println("F"); :}
| T_PLUS {: System.out.println("+"); :}

AddOp ::= T_PLUS {: System.out.println("+"); :}

What is the output on the screen when the follow statement is given as input:

\[ a = b + c; \]

Rewrite the Expr rule so expressions like \( a + b + c \) will evaluate as \( a + (b + c) \). Do not include the actions.

Which of the following would be correct if we wanted to add the minus sign (-) as an operator with higher precedence than the current plus sign (+)？ _____

<table>
<thead>
<tr>
<th>A</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Expr ::= Expr Op Designator</td>
<td>Expr T_PLUS Exprl</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Op ::= T_MINUS</td>
<td>T_PLUS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Expr ::= Expr Op Designator</td>
<td>Expr T_MINUS Exprl</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Op ::= T_PLUS</td>
<td>T_MINUS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Expr ::= Expr T_PLUS Exprl</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Exprl ::= Exprl T_MINUS Designator</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Expr ::= Expr T_MINUS Exprl</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Exprl ::= Exprl T_PLUS Designator</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2. In object-oriented languages like Java, determining which overloaded method code to bind to (to execute) is done at run time rather than at compile time (this is known as dynamic dispatching or dynamic binding). However, the name mangled symbol denoting a particular method name is determined at compile time. Given the following Java class definitions, specify the output of each print() method invocation.

```java
public class Overloading_Final_Exam {
    public static void main (String [] args) {
        Lois famGuy1 = new Lois();
        Peter famGuy2 = new Lois();
        Peter famGuy3 = new Brian();
        Brian famGuy4 = new Brian();
        Peter famGuy5 = new Peter();

        famGuy1.print(new Lois());
        ((Peter)famGuy1).print(new Lois());

        famGuy2.print(new Lois());
        ((Lois)famGuy2).print(new Lois());

        ((Brian)famGuy3).print(new Brian());
        famGuy3.print(new Brian());
        ((Brian)famGuy3).print(new Lois());

        famGuy4.print(new Brian());
        ((Peter)famGuy4).print(new Brian());
        ((Peter)famGuy4).print(new Lois());
        ((Peter)famGuy4).print(new Peter());

        famGuy5.print(new Brian());
        famGuy5.print(new Lois());
        famGuy5.print(new Peter());
    }
}
```
3. In your Project 2, how did you (and your partner if you had a partner) handle allocating and accessing global variables? Be specific how your project implemented this!

Which part of the entire compilation sequence clear through to program execution is responsible for:

a) resolving undefined external references with defined global references in other modules

b) ensuring the bss segment is set up and zero-filled

c) translating C source code into assembly target code

d) getting the executable image from disk into memory

e) translating assembly source code into object target code

f) creating an executable from multiple object files

e) expanding #defines and #includes

Using Reduced-C syntax, define a pointer to an array of 5 ints named `foo` such that `(*foo)[4] = 42` is a valid expression. This will most likely take two lines of code.

Change the following into two instructions that is an improvement over a single multiply instruction

```
  r3 = r2 * 17
```

```
  r3 = r2
      # Explicit zero

  r3 = 17
      # 17 is pushable
```

Optimize the following. `x` represents a memory location.

```
x = r1

r2 = x
```

```
x = r1

r2 = r1
```

Optimize the following

```
r1 = r2 + r4

r3 = r1 + r5

r6 = r2 + r4

r1 = 15 + r6
r6 = ...
```

```
r1 = r2
    # Explicit zero

r3 = r1
    # 15 is pushable

r6 = ...
    # 17 is pushable
```
4. Given the following Reduced-C code, write the equivalent code generated in SPARC Assembly.

```c
/* Reduced-C */

function : bool foo(int & a)
{
    bool b;
    int c = 5;
    a = c;
    return b;
}
```

/* Equivalent SPARC Assembly code */

Using the Right-Left rule (which follows the operator precedence rules) write the definition of a variable named bar that is an array of 8 elements where each element is a pointer to a function that takes a pointer to a struct fubar as a single parameter and returns a pointer to an array of 3 elements where each element is a pointer to a struct Baz. (9 points)
5. The following program to test linked lists in Reduced-C has three bugs that are manifested at run-time. Identify all three of them and what needs to be changed to fix each bug. Each change should require only one minor modification to the code (you should *not* need to do any restructuring of multiple lines of code).

```c
1  structdef LIST {
2    int data;
3    LIST* next;
4  };

5  function : void createList(LIST* list) {
6     new list;
7     list->data = 77;
8     new list->next;
9     list->next->data = 99;
10  }

11  function : void printList(LIST* &list) {
12     while(list != NULL) {
13         cout << list->data << endl;
14         list = list->next;
15     }
16  }

17  function : void deleteList(LIST* list) {
18     if(list->next == NULL) {
19         deleteList(list->next);
20     } delete list;
21  }

22  function : void main() {
23     LIST* head;
24
25     createList(head);
26     printList(head);
27     deleteList(head);
28  }

/*****************************/
Correct output should be:
77
99
(with no memory leaks)
/*****************************/

1)  

2)  

3)  

```
6. Consider the following error messages and programs as defined in the starterCode and Project I:

E1) public static final String error5n_Call =  
    "Number of arguments (%D) differs from number of parameters (%D).";

E2) public static final String error5c_Call =  
    "Non-addressable argument passed to reference parameter %S (type %T).";

E3) public static final String error5v_Call =  
    "Argument of type %T not equivalent to reference parameter %S, of type %T.";

E4) public static final String error5a_Call =  
    "Argument of type %T not assignable to value parameter %S, of type %T.";

function : bool foo( float a )
{
    return true;
}

function : int main() {
    float a;
    int b;
    bool c;

    foo( /* Pass a var here */ );
    return 0;
}

Which error message (E1, E2, E3, E4) should be displayed if you tried to pass any of the variables that might cause a compile error? If no vars would cause an error, state "No Error". __________

function : bool foo( float & a )
{
    return true;
}

function : int main() {
    float a;
    int b;
    bool c;

    foo( /* Pass a var here */ );
    return 0;
}

Which error message (E1, E2, E3, E4) should be displayed if you tried to pass any of the variables that might cause a compile error? If no vars would cause an error, state "No Error". __________

function : bool foo( float & a )
{
    return true;
}

function : int main() {
    int c;

    foo( c + 2.5 );
    return 0;
}

Which error message (E1, E2, E3, E4) should be displayed? If not an error, state "No Error". __________
7. Consider the following two C program files:

```
#include <stdio.h>
extern float a;
extern int c;
extern int foo( float z );
static int c = 420;
int a = 420;

int main( int argc, char *argv[] ) {
    int i = c;
    for ( i = 0; i < c; ++i )
        (void) printf( "%d ", foo( i ) );
    return 0;
}
```

Trying to separately compile each file and then link the resulting object modules

```
gcc –c file1.c file1.c –>file1.o
gcc –c file2.c file2.c –>file2.o
gcc file1.o file2.o file1.o & file2.o –> a.out
```

results in just one error being reported. We discussed some of the problems/complications imposed on the compiler when trying to perform static semantic type checking with separate compilation.

What error will be reported (specify the symbol name and a general description of what the problem is). Hint: The error will be reported in the 3rd gcc call which attempts to link the already compiled and assembled object modules.

Assume we fixed this error so the program will fully compile/link. How many times does the variable `b` in function `foo()` get initialized (once or every time function foo() is called)?

C (unlike C++) requires all global and static variable initializations to be done completely at compile time so the values in the Data segment are known at compile time. Can we change the initialization of `b` in file2.c to be `static int b = bar();` assuming function `bar()` is properly defined? Why or why not?

Can we change the initialization of `i` in file1.c to be `int i = bar();` assuming function `bar()` is properly defined? Why or why not?

Identify two other potential semantic errors in this program that the C compiler and linker did not detect, but lint will identify.

1) 

2)
8. Given the following program, specify the order of the output lines when run and sorted by the address printed with the `%p` format specifier on a Sun SPARC Unix and Linux system. For example, which line will print the lowest memory address, then the next higher memory address, etc. up to the highest memory address?

```c
#include <stdio.h>
#include <stdlib.h>

void foo1( int *, int ); /* Function Prototype */
void foo2( int, int * ); /* Function Prototype */

int main( int argc, char *argv[] ) {
    int a = 42;
    int b;
    foo2( a, &argc );
    /* 1 */ (void) printf( "1: foo2 --> %p\n", foo2 );
    /* 2 */ (void) printf( "2: a --> %p\n", &a );
    /* 3 */ (void) printf( "3: argv --> %p\n", &argv );
    /* 4 */ (void) printf( "4: b --> %p\n", &b );
    /* 5 */ (void) printf( "5: argc --> %p\n", &argc );
}

void foo1( int *c, int d ) {
    static int e = 404;
    struct foo {int a; int b;} f;
    int g;
    /* 6 */ (void) printf( "6: d --> %p\n", &d );
    /* 7 */ (void) printf( "7: malloc --> %p\n", malloc(50) );
    /* 8 */ (void) printf( "8: e --> %p\n", &e );
    /* 9 */ (void) printf( "9: c --> %p\n", &c );
    /* 10 */ (void) printf( "10: f.b --> %p\n", &f.b );
    /* 11 */ (void) printf( "11: g --> %p\n", &g );
    /* 12 */ (void) printf( "12: f.a --> %p\n", &f.a );
}

void foo2( int h, int *i ) {
    int j[2];
    static int k;
    foo1( j, h );
    /* 13 */ (void) printf( "13: h --> %p\n", &h );
    /* 14 */ (void) printf( /*14: j[1] --> %p\n", &j[1] );
    /* 15 */ (void) printf( "15: i --> %p\n", &i );
    /* 16 */ (void) printf( "16: k --> %p\n", &k );
    /* 17 */ (void) printf( "17: j[0] --> %p\n", &j[0] );
}
```

How can you reduce the stack space required for a large number of local variables of different types on a typical RISC architecture with no optimization compiler flags turned on?

What is the name of Rick's favorite beer? _______________________

Variables declared to be ________________ will not be optimized by the compiler.
9. Extra Credit (12 points total extra credit)

What gets printed when this program is executed?

```c
#include <stdio.h>

int main( void )
{
    char a[] = "CSE131 Rocks!";
    char *p = a + 3;

    printf( "%c", +++p ); ______
    printf( "%c", ++++p ); ______
    printf( "%c", *p++ + 1 ); ______
    p = p + 2;
    printf( "%c", *p = *p + 3); ______
    printf( "%c", --*p++ ); ______
    printf( "%d", p - a ); ______

    return 0;
}
```

For each of the following give the equivalent expression which in most cases is how the C compiler really treats/handles the expression.

```c
int a[5];
struct foo { int a; double b; } * p;

p->a ______

a[2] ______
```

If array a is located at memory location 0x5000, what memory location is a[3]?

If p has the value 0x6000, what memory location is p->b?

Tell me something you learned in this class that is extremely valuable to you and that you think you will be able to use for the rest of your computer science career. (1 point if serious; you can add non-serious comments also)

Crossword Puzzle (next page) (1 point)
A portion of the Operator Precedence Table

<table>
<thead>
<tr>
<th>Operator</th>
<th>Associativity</th>
</tr>
</thead>
<tbody>
<tr>
<td>++ postfix increment</td>
<td>L to R</td>
</tr>
<tr>
<td>-- postfix decrement</td>
<td></td>
</tr>
<tr>
<td>* indirection</td>
<td>R to L</td>
</tr>
<tr>
<td>++ prefix increment</td>
<td></td>
</tr>
<tr>
<td>-- prefix decrement</td>
<td></td>
</tr>
<tr>
<td>&amp; address-of</td>
<td></td>
</tr>
<tr>
<td>* multiplication</td>
<td>L to R</td>
</tr>
<tr>
<td>/ division</td>
<td></td>
</tr>
<tr>
<td>% modulus</td>
<td></td>
</tr>
<tr>
<td>+ addition</td>
<td>L to R</td>
</tr>
<tr>
<td>- subtraction</td>
<td></td>
</tr>
<tr>
<td>= assignment</td>
<td>R to L</td>
</tr>
</tbody>
</table>
Scratch Paper