Login name ________________

Quiz 5
CSE 131
Winter 2012

Name ______________________

Signature __________________

Student ID _________________

1. Give the order of the phases of compilation in a typical C compiler as discussed in class

A – Target language file (for ex., prog.s)	E – Parser (Syntax Analysis)
B – Source language file (for example, prog.c)	F – Intermediate Representation(s)
C – Parser (Semantic Analysis)	G – Scanner (Lexical Analysis)
D – Code generation (for ex., Assembly)	

______ –> ______ –> ______ –> ______ –> ______ –> ______ –> ______

Give 4 examples of operators/expressions/constructs in this quarter's version of Reduced-C that evaluate to a modifiable l-val. Use words vs. code. For example, "the arrow operator / struct member access via a pointer to struct" – so now you cannot use the arrow operator as one of your answers. (There are probably 5 more.)

1) ____

2) ____

3) ____

4) ____

Now explain the main difference in code gen for an expression that evaluates to a modifiable l-val vs. an r-val. Be specific about code gen – talk about what gets stored and why you need to tag the resulting ExprSTO as a modifiable l-val (vs. an r-val).

What is the 80/20 rule?

Give an example of a programming construct that exemplifies this rule.

What are the values of `a` and `b` after the following Reduced-C statements?

```
bool a = true;
bool b = false && (a = false);
```

Value of `a` is _________ Value of `b` is _________

Use virtual register notation for the following.

Change the following into three instructions which are most likely a time improvement over the single instruction when it comes to actual code generation. `x` represents a memory location.

```
r4 = 420
x = r4
r1 = r5 + r4
x = r1
r4 = r2 - r3
```

____

____

____

What terms describes these particular kinds of peephole optimizations? There are at least two.

1)

2)

What do you do differently in code gen to increment a pointer vs. incrementing an int?

How is accessing a global or static variable different than accessing a local variable or formal parameter in code gen?

Match the compilation process with the various tasks done in the compilation sequence.

A) C++ Preprocessor B) Loader C) Linkage Editor D) C++ Compiler E) Assembler

_____ takes an executable file on disk and makes it ready to execute in memory.

_____ resolves undefined external symbols with defined global symbols in other modules.

_____ zero fills the BSS segment in memory.

_____ puts globally defined symbols in the export list of the resulting object file.

_____ translates assembly code into machine code.

_____ combines all object modules into a single executable file.

What question(s) would you like to see on the final exam?