1. (25 pts) The Boolean type is typically represented on a byte in memory, although it can have only two possible values. Why not represent the Boolean type on a single bit?

2. (25 pts) Write a small fragment of code that shows how unions can be used in C to interpret the bits of a value of one type as if they represented a value of some other type (non-converting type cast).

3. (25 pts) Consider the following:

```c
typedef struct
{
    int x;
    char y;
} Rec1;
typedef Rec1 Rec2;
typedef struct
{
    int x;
    char y;
} Rec3;
Rec1  a, b;
Rec2  c;
Rec3  d;
```

Specify which of the variables `a, b, c, d` are type equivalent under (a) structural equivalence, (b) strict name equivalence, and (c) loose name equivalence.

4. (25 pts) Consider the following program, written in C:

```c
typedef struct
{
    int x;
    int y;
} Foo;

void allocate_node (Foo * f)
{
    f = (Foo *) malloc ( sizeof(Foo) );
}
```
void main ()
{
    Foo * p;
    allocate_node (p);
    p->x = 2;
    p->y = 3;
    free(p);
}

Although the program compiles, it produces a run-time error. Why?

Rewrite the two functions allocate_node and main so that the program runs correctly.

5. (Extra Credit - 10 pts) A compiler for a language with static scoping typically uses a LeBlanc-Cook symbol table to track the bindings of various names encountered in a program. Describe the mechanism that should be used by a compiler to ensure that the names of record fields used inside a with statement are bound to the correct objects. Specify what should be placed in (a) the symbol table, (b) the scope stack, and (c) the run-time program stack.