CS326 – Programming Languages, Concepts and Implementation

Homework 5

(Due April 11)

1. (25 pts) Suppose we are compiling for a machine with 1-byte characters, 2-byte shorts, 4-byte integers, and 8-byte reals, and with alignment rules that require the address of every primitive data element to be a multiple of the element's size. Suppose further that the compiler is not permitted to reorder fields. How much space will be consumed by the following array? Explain.

```
A : array [0..9] of record
                        s : short
                        c : char
                        t : short
                        d : char
                        r : real
                        i : integer
```

2. (25 pts) For the following code 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.

```
Type    T = array [1..10] of integer
        S = T
a : T
b : T
c : S
d : array [1..10] of integer
```

3. (25 pts) We are trying to run the following C program:

```
typedef struct
{
   int        a;
   char *     b;
} Cell;


void AllocateCell (Cell * q)
{
   q = (Cell *) malloc ( sizeof(Cell) );
}
```

```
void main ()
{
   Cell * c;
   AllocateCell (c);
   c->a = 1;
   free(c);
}
```

The program produces a run-time error. Why?

Rewrite the functions `AllocateCell` and `main` so that the program runs correctly.

4. (25 pts) Consider the following C declaration, compiled on a 32-bit Pentium machine (with array elements aligned at addresses multiple of 4 bytes).

```
struct
{
   int n;
   char c;
} A[10][10];
```

If the address of `A[0][0]` is 1000 (decimal), what is the address of `A[3][7]`? Explain how this is computed.

5. (Extra Credit - 10 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).