Willis Allstead

4/13/17

CS 326

HW5

1. Following these rules:

|  |  |  |
| --- | --- | --- |
| Var | Size (from pdf) | Range |
| s | 2 | 0-1 |
| c | 1 | 2-2 |
| t | 2 | 4-5 |
| d | 1 | 6-6 |
| r | 8 | 8-15 |
| i | 4 | 16-19 |
| Padding | 4 | 20-23 |

We need a total of 24 bytes \* 10 = 240 bytes to store this. This is including the padding needed to do this in reality.

* 1. Structural equivalence: a,b,c,d
  2. Strict name equivalence: a,b
  3. Loose name equivalence: a,b,c

1. The error comes from only passing a Cell \* to the AllocateCell function. It means that the allocation is attempting to allocate space for the pointer and not the structure it points to. We can fix this by passing a Cell \*\* instead:

void AllocateCell (Cell\*\* q)

{

\*q = (Cell \*) malloc ( sizeof(Cell) );

}

int main ()

{

Cell \* c;

AllocateCell (c);

c->a = 1;

free (c);

}

1. Address 1000 + ([3]\*10\*8) + ([7]\*8) = Address 1296