Homework 2

Problem 1

Consider the following C declaration:

```
struct Node {
    char a;
    double b;
    int c;
    int* d;
    struct Node* e;
};
```

1. Show the memory layout of struct `Node`. Label the bytes with the names of the various fields and **clearly indicate the right hand boundary of the data structure with a vertical line**. Use an X to denote space that is allocate in the struct as padding.

2. Rearrange the elements of `Node` to save the most space in memory. Label the bytes with the names of the various fields and **clearly indicate the right hand boundary of the data structure with a vertical line**. Use an X to denote space that is allocate in the struct as padding.

Problem 2

Consider the following C code:

```
double array[3][4][5] =
    {
        {
            {2.4, 4.3, 9.0, 0.9, -2.0 },
            {3.7, 2.6, 0.8, 1.5, 3.3 },
            {-1, 5, 2, 1, 7 },
            {1.9, 3.5, 9.2, 12, 1.0 }
        },
        {
            {6.4, 9.3, 0.2, 1.7, 2.3 },
            {2.2, 6.2, -0.3, 1.1, 0.6 },
            {-0.2, 9.0, 3.6, 1.2, -8.0 },
            {1.5, 5.4, 6.1, 9.2, 2.5 }
        },
        {
            {10.2, 2.5, -2.2, -9.4, 2.1 },
            {1.6, 12.1, 11.2, 3.2, 2.6 },
            {1.0, -0.7, 4.1, 1.2, 12.2 },
            {2.7, 12.2, 2.9, 6, 1 }
        }
    };
```

The memory address of `array` is **x.** Fill the blanks below:

Reference	**Type**	**Value**
array	double *	x
array[2][1][1]	_____	_____
array[0][1]	_____	_____
array[1]	_____	_____
array+1	_____	_____
*(array+1)	_____	_____
&array[0][0][1]	_____	_____
_____	_____	x + 32