Name _____ Per ___

Quadratic Sorts Analysis

```
void insertionSort (int  list[])
{
    int outer, pos, hold;
    int N = list.length;

    for (outer=1; outer <=  N-1; outer++)
    {
        pos = outer;

        while ( (pos > 0) && (list[pos-1] > list[pos]) )
        {
            swap (list, pos-1, pos);
            pos--;
            IamHere();
        }
    }
}
```

```
void selectionSort (int  list[])
{
    int flag;
    int N = list.length;

    for (int  outerPos=0; outerPos < N-1; outerPos++)
    {
        posOfSmallest = findPosOfSmallest(list,
                                outerPos, N-1);
        swap(list, outerPos, posOfSmallest);
    }
}
```

```
int findPosOfSmallest (int list[],
                              int from, int to)
{
    int smallestPos = from, i;
    for (i=from+1, i<=to; i++)
    {
        if (list [i] < list[smallestPos])
            smallestPos = i;
        IamHere();
    }
    return smallestPos;
}
```

Using the above algorithms, based on a list size of N, how many times will the statement **IamHere();** be executed for the sorts in the worst case? Best case? We will do one or two of the boxes below together in class.

Sort Name	*Describe Best Case (i.e., how do we arrange the data elements so that the algorithm does the least amount of work? random? ascending? descending?*	*Best Case Analysis*	*Describe Worst Case* *random?* *ascending?* *descending?*	*Worst Case Analysis*
Selection				
Insertion				