**CSCI 520**

**Programming Assignment #9**

To be done in the lab

12:00Noon-1:40PM, Friday, March 31, JOUR 102

In this assignment, you are given three sorting algorithms. You are asked to do the following:

1. Modify the bubble-sort function such that the outer loop stops iterating as soon as there have been no swaps in the inner loop. That is, as soon as the array is already sorted.
2. Modify each function such that you count the number of comparisons between array elements that they perform. Each function will output the final sorted array and the number of comparisons the function has performed.
3. Write a C++ program that receives as input an array of integers and runs each sorting algorithm (a separate function call on the original input) and outputs the final sorted array and number of comparisons each algorithm performs.

One input to your program is the following:

18 20 11 25 17 27 9 13 24 8 28 14 5 19 23 6 29 4 30 22 10 16 21 3 5 7 26

Selection Sort:

void selection(int a[], int l, int r)

{ for (inti = l; i< r; i++)

{ int min = i;

for (int j = i+1; j <= r; j++)

if (a[j] < a[min]) min = j;

exch(a[i], a[min]);

}

}

Insertion Sort:

void insertion(int a[], int l, int r)

{ inti;

for (i = r; i> l; i--) compexch(a[i-1], a[i]);

for (i = l+2; i<= r; i++)

{ int j = i; int v = a[i];

while (v < a[j-1])

{ a[j] = a[j-1]; j--; }

a[j] = v;

}

}

Bubble Sort:

void bubble(int a[], int l, int r)

{ for (inti = l; i< r; i++)

for (int j = r; j >i; j--)

compexch(a[j-1], a[j]);

}

**CSCI 520**

**Programming Assignment #9**

Turn in your work in the drop box for Assignment 9(in Week 9) in eCollege 520 course by 11:59pm on Thursday,April 13.

In this assignment, you are given three sorting algorithms. You are to do the following:

1. Modify the bubble-sort function such that the outer loop stops iterating as soon as there have been no swaps in the inner loop. That is, as soon as the array is already sorted.
2. Modify each function such that you count the number of comparisons between array elements that they perform. Each function will output the final sorted array and the number of comparisons the function has performed.

In each of the following questions you are given two functions A and B. You need to find an array for each case such that function A outperforms function B (i.e. number of comparisons function A performs / number of comparisons function B performs is minimized: the smaller this number higher your score). You may write a separate C++ program (or program segment, or function) for each case. Make sure that **THE ORGINAL ARRAY YOU FIND (INPUT) (this input is to be hard-coded in your submission)**, **the final sorted array**, **number of comparisons**, and **ratio of number of comparisons** are printed for each case.

Note that to be able to answer this question you should understand how each algorithm works, and works best on what kind of input, while other alternate algorithms do not perform similarly well (perform much worse as much as possible).

1. A=Selection Sort, B=Insertion Sort
2. A=Selection Sort, B=Modified Bubble Sort
3. A=Insertion Sort, B=Selection Sort
4. A=Insertion Sort, B=Modified Bubble Sort
5. A=Modified Bubble Sort, B=Insertion Sort
6. A=Modified Bubble Sort, B=Selection Sort

Selection Sort:

void selection(int a[], int l, int r)

{ for (inti = l; i< r; i++)

{ int min = i;

for (int j = i+1; j <= r; j++)

if (a[j] < a[min]) min = j;

exch(a[i], a[min]);

}

}

Insertion Sort:

void insertion(int a[], int l, int r)

{ inti;

for (i = r; i> l; i--) compexch(a[i-1], a[i]);

for (i = l+2; i<= r; i++)

{ int j = i; int v = a[i];

while (v < a[j-1])

{ a[j] = a[j-1]; j--; }

a[j] = v;

}

}

Bubble Sort:

void bubble(int a[], int l, int r)

{ for (inti = l; i< r; i++)

for (int j = r; j >i; j--)

compexch(a[j-1], a[j]);

}