Shammah Thao
Csc20-11

Final Exam part 1

- Highlights are for the number that are switching with each other.

1.

Bubble Sort:

54	25	93	17	77	31	44	55	22
25	54	93	17	77	31	44	55	22
25	54	93	17	77	31	44	55	22
25	54	17	93	77	31	44	55	22
25	54	17	77	93	31	44	55	22
25	54	17	77	31	93	44	55	22
25	54	17	77	31	44	93	55	22
25	54	17	77	31	44	55	93	22
25	54	17	77	31	44	55	22	93
25	54	17	77	31	44	55	22	93
25	17	54	77	31	44	55	22	93
25	17	54	31	77	44	55	22	93
25	17	54	31	44	77	55	22	93
25	17	54	31	44	55	77	22	93
25	17	54	31	44	55	22	77	93
17	25	54	31	44	55	22	77	93

17	25	31	54	44	55	22	77	93
17	25	31	44	54	55	22	77	93
17	25	31	44	54	22	55	77	93
17	25	31	44	22	54	55	77	93
17	25	31	22	44	54	55	77	93
17	25	22	31	44	54	55	77	93
17	25	22	31	44	54	55	77	93
17	22	25	31	44	54	55	77	93

Selection Sort:

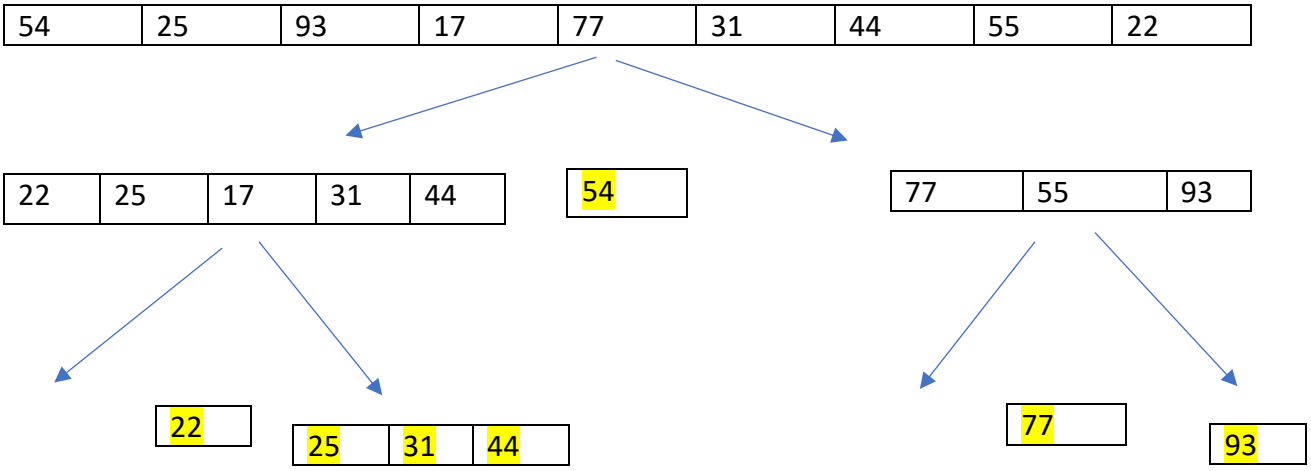
54	25	93	17	77	31	44	55	22
54	25	22	17	77	31	44	55	93
54	25	22	17	77	31	44	55	93
54	25	22	17	55	31	44	77	93
54	25	22	17	55	31	44	77	93
54	25	22	17	44	31	55	77	93
54	25	22	17	44	31	55	77	93
31	25	22	17	44	54	55	77	93
31	25	22	17	44	54	55	77	93
31	25	22	17	44	54	55	77	93
17	25	22	31	44	54	55	77	93
17	25	22	31	44	54	55	77	93

17	22	25	31	44	54	55	77	93
17	22	25	31	44	54	55	77	93

Insertion:

54	25	93	17	77	31	44	55	22
25	54	93	17	77	31	44	55	22
25	54	93	17	77	31	44	55	22
17	25	54	93	77	31	44	55	22
17	25	54	77	93	31	44	55	22
17	25	31	54	77	93	44	55	22
17	25	31	44	54	77	93	55	22
17	25	31	44	54	55	77	93	22
17	22	25	31	44	54	55	77	93
17	22	25	31	44	54	55	77	93

Quick Sort:



17

55

17	22	25	31	44	54	55	77	93

2.

Bubble sort:

```
Array Before Bubble Sort
54 25 93 17 77 31 44 55 22
Before comparing
Swapping
Before comparing
Before comparing
Swapping
Before comparing
Swapping
Before comparing
Swapping
Before comparing
Swapping
Before comparing
Swapping
Before comparing
Swapping
--------------
Before comparing
Before comparing
```

8 compares

```
Swapping
Before comparing
Before comparing
Swapping
Before comparing
Swapping
Before comparing
Swapping
Before comparing
Swapping
--------------
Before comparing
Swapping
Before comparing
Before comparing
Swapping
Before comparing
Swapping
Before comparing
Before comparing
Swapping
--------------
Before comparing
Before comparing
Before comparing
Before comparing
Before comparing
Swapping
--------------
Before comparing
Before comparing
Before comparing
Before comparing
Swapping
--------------
Before comparing
Before comparing
Before comparing
```

7 compares

6 compares

5 Compares

4 compares

3 compares

```
Swapping
--------------
Before comparing
Before comparing
Swapping
--------------
Before comparing
--------------
--------------
Array After Bubble Sort
17 22 25 31 44 54 55 77 93
 ----jGRASP wedge2: exit code for process is 0.
```

2 compares

1 compares

Selection sort:

```
Before Selection Sort
54 25 93 17 77 31 44 55 22
Before comparing
Swapping
Before comparing
Before comparing
Swapping
Before comparing
Before comparing
Before comparing
Before comparing
Before comparing
---------------
Before comparing
Before comparing
Before comparing
Before comparing
Before comparing
Before comparing
Before comparing
Swapping
```

8 compares

7 compares

```
---------------
Before comparing
Swapping
Before comparing
Before comparing
Swapping
Before comparing
Before comparing
Before comparing
Swapping
---------------
Before comparing
Before comparing
Swapping
Before comparing
Before comparing
Before comparing
---------------
Before comparing
Swapping
Before comparing
Swapping
Before comparing
Before comparing
---------------
Before comparing
Before comparing
Before comparing
---------------
Before comparing
Swapping
Before comparing
---------------
Before comparing
---------------
After Selection Sort
17 22 25 31 44 54 55 77 93
 ----jGRASP wedge2: exit code for process is 0.
```

6 compares

5 compares

4 compares

3 compares

2 compares

1 compares

Insertion Sort:

```
 ----jGRASP wedge2: pid for process is 3328.
Before Insertion Sort
54 25 93 17 77 31 44 55 22
Before comparing
Swapping
---------------
Before comparing
---------------
Before comparing
Swapping
Swapping
Swapping
---------------
Before comparing
Swapping
---------------
Before comparing
Swapping
Swapping
Swapping
---------------
Before comparing
Swapping
Swapping
Swapping
---------------
Before comparing
Swapping
Swapping
---------------
Before comparing
Swapping
Swapping
Swapping
Swapping
```

1 compares

1 compares

1 compares

1 compares

1 compares

1 compares

1 compares

1 compares

```
Swapping
Swapping
Swapping
---------------
After Insertion Sort
17 22 25 31 44 54 55 77 93
 ----jGRASP wedge2: exit code for process is 0.
```

Quicksort sort:

```
 ----jGRASP wedge2: pid for process is 13416.
Before Quick Sort
54 25 93 17 77 31 44 55 22
[54, 25, 93, 17, 77, 31, 44, 55, 22]
Before comparing
Swapping
----------------------
Before comparing
----------------------
Before comparing
Swapping
----------------------
Before comparing
----------------------
Before comparing
Swapping
----------------------
Before comparing
Swapping
----------------------
Before comparing
----------------------
Before comparing
Swapping
----------------------
Eoll: 5
```

8 compares

```
Pivot Point 54
[22, 25, 17, 31, 44]
Before comparing
----------------------
Before comparing
Swapping
----------------------
Before comparing
----------------------
Before comparing
----------------------
Eoll: 1
Pivot Point 22
[25, 31, 44]
Before comparing
----------------------
Before comparing
----------------------
Eoll: 2
Pivot Point 25
[31, 44]
Before comparing
----------------------
Eoll: 3
Pivot Point 31
[77, 55, 93]
Before comparing
Swapping
----------------------
Before comparing
----------------------
Eoll: 7
Pivot Point 77
After Quick Sort
17 22 25 31 44 54 55 77 93
 ----jGRASP wedge2: exit code for process is 0.
```

4 compares

2 compares

1 compare

2 compares

Based on the comparison of array above for the 4 different kind of sorting. It seems that insertion sorting is the most efficient kind of sorting for this kind of problems. The reason being is that it, it has the less amount of comparison compare to the other sorting. It seems to effectively and quick choose which number is the biggest and swap it with the number that is in its location. Since the problem is small and close to being sorted already, insertion sort is a lot easier to use to compare the numbers. Although quicksort is extremely good its isn't as affective on sorting problem that is relatively small, since it takes times to divide and conquer the problem.