Brenden Nelson
ID # 11635716

350 HW #2

(1) pseudo-code for Partition(A, p, q)

```
pivot = A[q] //higher
i = (p-1) //smaller
for (j = p; j <= q-1; j++)
{       if A[j] < pivot)
        {
            i++

            switch A[i] + A[j]
        }
}

switch A[i+1] + A[q]
}
```

(2) InsertSort of A w/ 1% probability to be monotonically decreasing

$T_{avg1\%}(n) = .01(T_w(n)) + .99(T_{avg}(n))$

$T_{avg}(n) = O(n^2)$ and $T_w(n) = O(n^2)$,

$\Theta(n^2) \leq T_{avg1\%}(n) \leq \Theta(n^2) + O(n^2)$

Therefore, we can conclude $T_{avg1\%}(n) = \Theta(n^2)$

(3) IqSort

<u>quick alg</u>

1) Partition //time complexity $= O(n)$
2) quicksort low // time complexity $= T_{quick}(r-1)$
3) Insert sort high // time complexity $= T_{ins}(n-r)$

$$T_{Iqbest}(n) = O\overbrace{\left(((r-1)\log(r-1))\right)}^{\text{step 2}} + O\overbrace{(n-r)}^{\text{step 3}} + O\overbrace{(n)}^{\text{step 1}}$$

Therefore $T_{Iqbest}(n) = O(n)$. // array is sorted

$$Iqworst(n) = \overbrace{O(r-1)^2}^{\text{step 2}} + \overbrace{O(n-r)^2}^{\text{step 3}} + \overbrace{O(n)}^{\text{step 1}}$$

Therefore $T_{Iqworst}(n) = O(n^2)$. // array keeps decreasing

$$IqAvg(n) = \frac{1}{n}\sum_{r=1}^{n}\left(\overbrace{T_{Avg}Q(r-1)}^{\text{step 2}} + \overbrace{T_{Avg}I(n-r)}^{\text{step 3}} + \overbrace{O(n)}^{\text{step 1}}\right)$$

$$IqAvg(n) = \frac{1}{n}\sum_{r=1}\left(O(n\log n) + O(n^2) + O(n)\right)$$

Scanned by TapScanner

$T_q Avg(n) \le O(n) + \frac{1}{n} \sum_{r=1}^{n} a \cdot n\log n + b \cdot n^2$

therefore $T_{avg}(n) \le O(n) + O(n^2)$, we can conclude that $T_{avg}(n) = O(n^2)$.

(4) MixSort

quick Alg

1) Partition // time complexity $= O(n)$
2) mixsort recursion on low // time complexity $= T_{mix}(r-1)$
3) insertSort on high // time complexity $= T_{ins}(n-r)$

$$Mix_{best}(n) = \underbrace{O(r-1)}_{step\,2} + \underbrace{O(n-r)}_{step\,3} + \underbrace{O(n)}_{step\,1}$$

Therefore, $Mix_{best}(n) = O(n)$ // input already sorted

$$Mix_{worst}(n) = \underbrace{O(r-1)^2}_{step\,2} + \underbrace{O(n-r)^2}_{step\,3} + \underbrace{O(n)}_{step\,1}$$

Therefore, $Mix_{worst}(n) = O(n^2)$ // array decreases

$$mixAvg(n) = \frac{1}{n}\sum_{r=1}^{n}\left(mixAvg(r-1) + t_{insAvg}(n-r) + O(n)\right)$$
$$= O(n) + \frac{1}{n}\sum_{r=1}^{n}\left(O(r-1)^2 + O(n-r)^2\right)$$
$$mixAvg(n) \le O(n) + \frac{1}{n}\sum_{r=1}^{n} a \cdot (r-1)^2 + b \cdot (n-r)^2$$

I guess $mixAvg(n) = O(n^2)$. That is $mixAvg(n) \le c \cdot n^2$ for some constant $c > 0$ for all n.

I.H. For $\forall i < n$, $mixAvg(i) \le c \cdot i^2$

$$mixAvg(n) \le \frac{1}{n}\left\{\sum_{r=1}^{n} mixAvg(r-1) + O(n-r)^2\right\} + \frac{1}{n}\,\cancel{a \cdot n^2}\ a \cdot n$$
$$\le \frac{1}{n}\left\{\sum_{r=1}^{n} O(r-1)^2 + O(n-r)^2\right\} + a \cdot n$$

⋮

$$\le c \cdot n^2 \text{ when } c > 7a.$$