PROBLEM SET 2

Zhiqiang Xie 77892769

Problem 1

Raw	First	Next	Last
COW	SEA	TAB	BAR
DOG	TEA	BAR	BIG
SEA	MOB	EAR	BOX
RUG	TAB	TAR	COW
ROW	DOG	SEA	DIG
MOB	RUG	TEA	DOG
BOX	DIG	DIG	EAR
TAB	BIG	BIG	FOX
BAR	BAR	MOB	MOB
EAR	EAR	DOG	NOW
TAR	TAR	COW	ROW
DIG	COW	ROW	RUG
BIG	ROW	NOW	SEA
TEA	NOW	BOX	TAB
NOW	BOX	FOX	TAR
FOX	FOX	RUG	TEA

Problem 2

The number in **bold** text is comparing to those in `monospace`, and they'll all be moved.

4	9	2	3	5	7	8	1	6
4	9	2	3	5	7	8	1	6
`4`	**9**	2	3	5	7	8	1	6
`4`	`9`	**2**	3	5	7	8	1	6
2	`4`	`9`	**3**	5	7	8	1	6
2	3	4	`9`	**5**	7	8	1	6
2	3	4	5	`9`	**7**	8	1	6
2	3	4	5	7	`9`	**8**	1	6
`2`	`3`	`4`	`5`	`7`	`8`	`9`	**1**	6
1	2	3	4	5	`7`	`8`	`9`	**6**
1	2	3	4	5	6	7	8	9

Problem 3

Python

```python
def sortStack(stack1):
    """
    :type stack1: List[int]
    :rtype: List[int]
    """
    stack2 = []
    while not stack1.empty():
        tem = stack1.pop()
        while (not stack2.empty()) and tem > stack2.top():
            stack1.push(stack2.pop())
        stack2.push(tem)
    return stack2
```

Problem 4

Python

```python
def findKthSmallest(array, k):
    """
    Revised quick sort algorithm to find the k'th smallest number of an array
    in expected O(n) time
    :type array: List[int]
    :type k: int
    :rtype: int
    For concise, 1 <= k <= len(array) and no duplicate are supposed
    """
    pivot = array[0]
    tem_index = 1
    for i in range(1, len(array)):
        if array[i] < pivot:
            array[i], array[tem_index] = array[tem_index], array[i]
            tem_index += 1
        array[0], array[tem_index-1] = array[tem_index-1], array[0]
    if k == tem_index:
        return pivot
    elif k > tem_index:
        return findKthSmallest(array[tem_index:], k - tem_index)
    else:
        return findKthSmallest(array[:tem_index], k)
```

The expected time complexity $T(n) = \Theta(n) + T(n/2) \in O(n)$

Problem 5

Python

```python
def searchCorTarget(array):
    """
    Revised binary search to find the i, A[i] == i
    :type array: List[int]
    :rtype: int or None if target is not found
    Count the index from 0 instead of 1
    """
    left, right = 0, len(array) - 1
    while left <= right:
        mid = (left+right) // 2
        if array[mid] == mid:
            return mid
        elif array[mid] < mid:
            left = mid + 1
        else:
            right = mid - 1
    return None
```

The time complexity $T(n) = \Theta(1) + T(n/2) \in \Theta(log(n))$

Problem 6

1. As for base case $n = 2$, we can easily show that CURLY-SORT swaps the two elements if they are not sorted.

   Now we assume CURLY-SORT correctly sorts an array with length of k and $1 \leq k < n$. Particularly, we set $k = 2n/3$. Therefore,

   - We firstly correctly sort $A[i : j - k]$ which contains approximately the preceding $2n/3$ elements.
   - Then we correctly sort $A[i + k : j]$ which contains approximately the last $2n/3$ elements.
   - Since $k \leq (j - i + 1)/3$, we have $(j - k) - (i + k) + 1 = j - i - 2k + 1 >= k = j - (j - k)$

     Therefore, we guarantee that all elements in $A[j - k + 1 : j]$ are larger than anyone in $A[i : j - k]$
   - Finally, we correctly sort $A[i : j - k]$ and we have a sorted $A[i : j]$ with length of n.

   Proved by the principle of induction.

2. $T(n) = 3T(2n/3) + \Theta(1) = \Theta(n^{log_{3/2}3})$ By Master Theorem.
3. As for worst case, $log_{3/2}3 \approx 2.71 > 2$, which means this algorithm is the worst one.

Algorithm	**Time Complexity(worst case)**
Insertion-sort	$\Theta(n^2)$
Merge-sort	$\Theta(nlog(n))$
Quick-sort	$\Theta(n^2)$
CURLY-SORT	$\Theta(n^{2.71})$