Homework 2

Zhimiao Lai zl2083 N11140147

1.

1) Divide A[1..n] into three segments: A[1,1third], A[1third+1, 2third], A[2third+1,n].

2) Find three maximum subarrays which are entirely in A[1,1third]], A[1third+1, 2third], A[2third+1,n].

3) Find maximum subarray crossing boundarys:

```
Find-Max-Cross-Subarray(A,low,1third,2third,high)
        Sum=0
        Left-sum= A[1third]
        For i=1third-1 downto low
                Sum=sum + A[i]
                If sum > left-sum
                        Left-sum=sum
                        Max-left=i
        Left-sum-2=-1000
        For i=2third downto 1third+1
                Sum= sum+ A[i]
                If sum>left-sum-2
                        Left-sum-2=sum
                        Max-left-2=i
        If left-sum-2 > left-sum
                Left-sum=left-sum-2
                Max-left=max-left-2
        Sum=0
        right-sum=-1000
        For j=1third+1 upto 2third
                Sum=sum + A[j]
                If sum>right-sum
                        Right-sum=sum
                        Max-right=j
                        If max-right=2third
                                For k=2third+1 upto high
                                        Sum = sum + A[k]
                                        If sum>right-sum
                                                Right-sum=sum
                                                Max-right=k
Return(max-left, max-right, left-sum + right-sum)
```

Total time: $T(n)=3T(n/3)+ \Theta (4n/3)$

$T(n)= \Theta(n\log n)$

2.

a.

In each list, the worst case running time is Θ(k^2). There are n/k lists in total, so the worst case total running time is Θ($k^2 \frac{n}{k}$)= Θ(k)

b.

The depth of merging from 1 element lists is log(n). We start from length k lists, so the depth is log(n)-log(k). And the running time in each level is still Θ(n). So total running time is Θ$\left(nlog\left(\frac{n}{k}\right)\right)$

c.

The standard merge sort running time is Θ($nlogn$). Let

$\Theta(nlogn) = \Theta\left(nk + nlog\left(\frac{n}{k}\right)\right) = \Theta(nk + nlogn - nlogk)$. In the right term, nk already dominates nlogk, and as long as k grows no faster than $logn$, the equation stands. So $k(n) = O(logn)$.

d.

Choose $k = logn$ can minimize the running time.

3.

Input: two length n arrays A and B, each contains binary values of a and b

Output: a length n+1 array C, containing binary values of a+b

```
C=zeros(1, n+1)
For i=n downto 1
C[i+1]=(A[i]+B[i]+c)mod2
If A[i]+B[i]+c >= 2
        c=1
else
                c=0
C[1]=c
```

4.

Bubble sort:

[25, 13, 10, 16, 5, 8, 1]

[25, 13, 10, 16, 5, 1, 8]

[25, 13, 10, 16, 1, 5, 8]

[25, 13, 10, 1, 16, 5, 8]

[25, 13, 1, 10, 16, 5, 8]

[25, 1, 13, 10, 16, 5, 8]

[1, 25, 13, 10, 16, 5, 8]

[1, 25, 13, 10, 5, 16, 8]

[1, 25, 13, 5, 10, 16, 8]

[1, 25, 5, 13, 10, 16, 8]

[1, 5, 25, 13, 10, 16, 8]

[1, 5, 25, 13, 10, 8, 16]

[1, 5, 25, 13, 8, 10, 16]

[1, 5, 25, 8, 13, 10, 16]

[1, 5, 8, 25, 13, 10, 16]

[1, 5, 8, 25, 13, 10, 16]

[1, 5, 8, 25, 10, 13, 16]

[1, 5, 8, 10, 25, 13, 16]

[1, 5, 8, 10, 13, 25, 16]

[1, 5, 8, 10, 13, 16, 25]

Insertion sort:

[25, 13, 10, 16, 5, 8, 1]

[13, 25, 10, 16, 5, 8, 1]

[10, 13, 25, 16, 5, 8, 1]

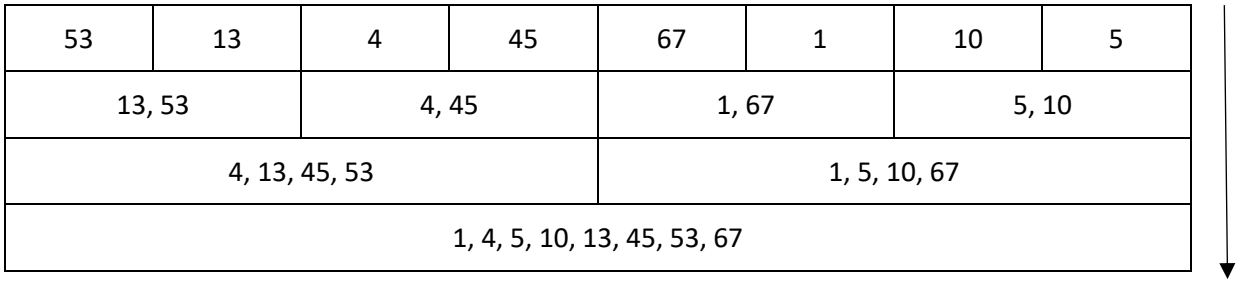
[10, 13, 16, 25, 5, 8, 1]

[5, 10, 13, 16, 25, 8, 1]

[5, 8, 10, 13, 16, 25, 1]

[1, 5, 8, 10, 13, 16, 25]

5.


53
13
4
45
67
1
10
5
13, 53
4, 45
1, 67
5, 10
4, 13, 45, 53
1, 5, 10, 67
1, 4, 5, 10, 13, 45, 53, 67