Assignment 5

11510257 彭福

Exe.1.

Solution:

Define: database A, database B, integer A(k) or B(k) is the kth smallest value we want to ask.

Let k = . If A() >B(), then we know that A() is larger than n-+ = n numbers in two database at least, and B() is less than n- + = n numbers in two database at least. Therefore, the median number we want is in the smallest number in A and in the largest number in B. And then we do the same thing as above recursively until we find only one number in A and one number in B may be the median number.

Function find\_median(n,A\_start,B\_start):

If n = 1, **return** min(A(A\_start),B(B\_start));

If(A(A\_start + celling(n/2))>B(B\_start + celling(n/2)));

Then B\_start += celling(n/2);

Else

Then A\_start += celling(n/2);

n= celling(n/2);

find\_median(n,A\_start,b\_start);

Exe.2.

Solution:

We can use the algorithm to counting the inversion in the textbook with some modification. The modification is merging twice every time we merge two sequences. The first merge is the original sequence a­1, a2, a3…an and b1, b2, b3… bn to sorting the sequences. And the second merge is a­1, a2, a3…an and 2\*b1, 2\*b2, 2\*b3 … 2\*bn to counting the significant inversions.

Funtion merge\_and\_count(A,B):

**int** a,b=0;

**int** i = 0;

**int** count = 0;

let sequence B1={ 2\*b1, 2\*b2, 2\*b3 … 2\*bn };

let L is the sequence after merging;

**while**(a<length(A) &&b<length(B))

{

**if**(A[a]<B[b])

{ a++;

L[i] = A[a];

i++;

}

**else**

{ b++;

L[i] = B[b];

i++;

}

}

put the remain number into L;

a = 0;

b = 0;

**while**(a<length(A) &&b<length(B1))

{

**if**(A[a]<B1[b])

then a++;

**else**

{ b++;

count+=length(A)-a;

}

}

**return** count;

Function sort\_and\_count(L):

If (length(L)==1)

then **return** 0;

**else**

{

let A be the first half of L;

let B be the second half of L;

count\_A = sort\_and\_count(A);

count\_B = sort\_and\_count(B);

count = merge\_and count(A,B);

}

Return count = count\_A + count\_B+count;