ADA Homework 3
B03902089 林良翰

Problem 1.(1)

Sad case :

$$t_i = \left\{ f + (s+1) \times k \;\middle|\; k \textit{ is integer} \right\}$$

Problem 1.(2)

f = 1, if s = 3, we can create the table :

time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
require			O				O	O	O	O			O				O
clean fish	F	T	F	F	F	T	F	F	T	F	F	T	F	F	F	T	F
fish num	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1

```
==========
cleanTimeList = [ ]
function findCleanTime(t_list, s, cleanTimeList) :
      for i = 0 to size of t_list-1 :
            if  i < 1 :
                  add t_list[i]-1 to cleanTimeList
            else :
                  if cleanTimeList.last + (s+1) ≤ t_list[i] :
                        add (t_list[i]-1) to cleanTimeList
                  end if
            end if
      end for
      return cleanTimeList
end function
```

Problem 1.(3)

If f = 3, s = 5, we can make a special table :

time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
require				O	O					O		O	O	O			O		
3																			
2																			
1																			
clean	T	T	T	F	F	F	T	T	T	F	F	T	T	F	F	F	F	F	F

By observing the table, we need to verify every t_i whether it has enough fish by traversing the last few elements of cleanTimeList

```
==========
cleanTimeList = [ ]
n = 0
for i = 0 to size of t_list-1 :
```

```
        if i < 1 :
                for j = 1 to f :
                        add t_list[i]-j to cleanTimeList[n]
                        n++
                end for
        else :
                curFish = 0
                for j = 1 to f :
                        if cleanTimeList[n-j] + s ≥ t_list[i]
                                curFish++
                        else : break for
                end for
                for j = 1 to f-curFish :
                        add t_list[i]-j to cleanFishList[n]
                        n++
                end for
        end if
end for
```

Problem 2.(1)

By definition, define “F” to be the influence :

$$F = \frac{1}{T} \sum_{i=1}^{N-1} b_i \sum_{j=i+1}^{N} p_j$$

problem 2.(2)

original	new
$b_{i-1}\ (p_i+p_{i+1}+p_{i+2}+p_{i+3}\ldots p_n)$	$b_{i-1}\ (p_{i+1}+p_i+p_{i+2}+p_{i+3}\ldots p_n)$
$b_i\ (p_{i+1}+p_{i+2}+p_{i+3}\ldots p_n)$	$b_{i+1}\ (p_i+p_{i+2}+p_{i+3}\ldots p_n)$
$b_{i+1}\ (p_{i+2}+p_{i+3}\ldots p_n)$	$b_i\ (p_{i+2}+p_{i+3}\ldots p_n)$

The difference =

$$\Delta F = \frac{b_{i+1}p_i - b_i p_{i+1}}{T}$$

Problem 2.(3)

Define f(S) to be the influence of the sequence S

It is ensured that $f(S_{swap(k,k+1)}) < f(S)$, suppose that $f(S_{swap(k,k+2)}) > f(S)$:

In the other word, sequence <…k+2, k+1, k,…> is more influent

By original definition :

$$f(S_{swap(k,\,k+1)}) < f(S)$$

$$f(S_{swap(k+1,\,k+2)}) < f(S)$$

$$\Rightarrow f(S_{swap(k,\,k+2)}) < f(S)$$

contradiction!

Problem 2.(4)

Design the merge sort which complexity is $O(n\log n)$

And we use the compare function which return difference (problem2.2) ΔF

Finally we will find the sequence.

problem 2.(5)

```cpp
#include <cstdio>
#include <vector>
#include <algorithm>
using namespace std;

class element {
    int p;  // price
    int b;  // influence level
    // constructor
    element(int _p, int _b) {
        p = _p;
        b = _b;
    }
};

bool operator<(const element &n1, const element &n2)
{
    if (n2.b * n1.p - n1.b * n2.p < 0) return true;
    else return false;
}

int main()
{
    vector<element> seq;
    int p, b;
    //
    // read the original sequence
    //
    sort(seq.begin(), seq.end());   // time complexity O(nlogn)
    //
    // print result
    //
    return 0;
}
~
```

Problem 3.(1)

If Paul choose one some element out of $\{a_{[1]1},\ a_{[3]1},\ a_{[5]1}\ldots,\ a_{[2n+1]1} \mid n\ is\ integer\}$

⇒ Paul doesn't use optimal strategy

⇒ Contradiction!

⇒ Paul must choose the element in : $\{a_{[1]1},\ a_{[3]1},\ a_{[5]1}\ldots,\ a_{[2n+1]1} \mid n\ is\ integer\}$

We can also prove that John have to choose the element in :

$\{a_{[2]1},\ a_{[4]1},\ a_{[6]1}\ldots,\ a_{[2n]1} \mid n\ is\ integer\}$

problem 3.(2)
We just concentrate on a pile of cards i_n
Paul and John all choose n_i, then :
⇒ Paul will get a_{i1}, a_{i2}, …, $a_{i(ni/2)}$

⇒ John will get $a_{i(ni/2+1)}$, …, a_{ini}

sum up every piles :
Paul will get at least a score of

$$\sum_{i=1}^{N} \sum_{j=1}^{\frac{n_i}{2}} a_{ij}$$

John will get at least a score of

$$\sum_{i=1}^{N} \sum_{j=\frac{n_i}{2}+1}^{n_i} a_{ij}$$

problem 3.(3)
warning : problem too difficult

problem 3.(4)
warning : problem too difficult