ADS2021Midterm-wc

开始时间	2022/04/15 15:30:00	**结束时间**	2022/04/29 12:00:00	**答题时长**	45分钟
答卷类型	标准答案	**总分**	100		

判断题

得分：暂无　总分：20

1-1 For the recurrence equation $T(N) = aT(N/b) + f(N)$, if $af(N/b) = Kf(N)$ for some constant $K > 1$, then $T(N) = \Theta(f(N))$. (2分)

○ T ● F

1-2 A skew heap is a heap data structure implemented as a binary tree. Skew heaps are advantageous because of their ability to merge more quickly than balanced binary heaps. The worst case time complexities for Merge, Insert, and DeleteMin are all $O(N)$, while the amortized time complexities for Merge, Insert, and DeleteMin are all $O(logN)$. (2分)

● T ○ F

1-3 When measuring the relevancy of the answer set, if the precision is high but the recall is low, it means that most of the relevant documents are retrieved, but too many irrelevant documents are returned as well. (2分)

○ T ● F

1-4 For an AVL tree, the balance factors of all the non-leaf nodes are 0 iff the tree is a complete binary tree. (2分)

○ T ● F

1-5 Finding the minimum key from a splay tree will result in a tree with its root having no right subtree. (2分)

○ T ● F

1-6 The time bound of the FIND operation in a B+ tree containing N numbers is $O(\log N)$, no matter what the degree of the tree is. (2分)

● T ○ F

1-7 In a red-black tree, the number of internal nodes in the subtree rooted at x is no more than $2^{bh(x)} - 1$ where $bh(x)$ is the black-height of x. (2分)

○ T ● F

1-8 In a Turnpike Reconstruction Problem, given distance set D = { 2, 2, 4, 4, 6, 8 }, x1=x4 = (0, 2, 4, 8) is the only solution provided that x1 = 0. (2分)

○ T ● F

1-9 For a binomial queue, delete-min takes a constant time on average. (2分)

○ T ● F

1-10 If a problem can be solved by dynamic programming, it must be solved in polynomial time. (2分)

○ T ● F

单选题

得分：暂无　总分：40

2-1 When solving a problem with input size N by divide and conquer, if at each step, the problem is divided into 9 sub-problems and each size of these sub-problems is $N/3$, and they are conquered in $O(N^2logN)$. Which one of the following is the closest to the overall time complexity? (4分)

● A. $O(N^2log^2N)$
○ B. $O(N^2logN)$
○ C. $O(N^2)$
○ D. $O(N^3logN)$

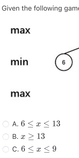
2-2 Among the following groups of concepts, which group is not totally relevant to a search engine? (4分)

● A. thresholding, dynamic programming, precision
○ B. word stemming, compression, recall
○ C. distributed index, hashing, inverted file index
○ D. stop words, posting list, dynamic indexing

2-3 Insert { 28, 12, 18, 36, 42, 30 } one by one into an initially empty AVL tree. The pre-order traversal sequence of the resulting tree is: (4分)

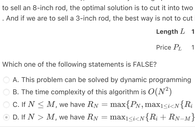
○ A. 36, 18, 12, 30, 28, 42
○ B. 12, 18, 30, 42, 36, 28
● C. 28, 18, 12, 36, 30, 42
○ D. 28, 12, 18, 36, 30, 42

2-4 After deleting 10 from the red-black tree given in the figure, which one of the following statements must be FALSE? (4分)



○ A. 11 is the parent of 6, and 14 is red
○ B. 8 is the parent of 15, and 7 is black
● C. 8 is the parent of 15, and there are 2 red nodes in the tree
○ D. 11 is the parent of 15, and there are 2 red nodes in the tree

2-5 Merge the two leftist heaps in the following figure. Which one of the following statements is FALSE? (4分)



○ A. the null path length of 6 is the same as that of 2
○ B. 1 is the root with 3 being its right child
● C. Along the left most path from top down, we have 1, 2, 4, and 5
○ D. 6 is the left child of 2

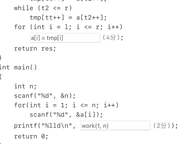
2-6 A queue can be implemented by using two stacks S_A and S_B as follows: (4分)

- To enqueue x, we push x onto S_A.
- To dequeue from the queue, we pop and return the top item from S_B. However, if S_B is empty, we first fill it (and empty S_A) by popping the top item from S_A, pushing this item onto S_B, and repeat until S_A is empty.

Assuming that push and pop operations take $O(1)$ worst-case time, please select a potential function ϕ which can help us prove that enqueue and dequeue operations take $O(1)$ amortized time (when starting from an empty queue).

● A. $\phi = 2|S_A|$
○ B. $\phi = |S_A|$
○ C. $\phi = 2|S_B|$
○ D. $\phi = |S_B|$

2-7 Given the following game tree, the red node will be pruned with α-β pruning algorithm if and only if ___. (4分)



○ A. $6 \le x \le 13$
○ B. $x \ge 13$
○ C. $6 \le x \le 9$
● D. $x \ge 9$

2-8 Which of the following binomial trees can represent a binomial queue of size 141? (4分)

● A. $B_0\ B_2\ B_3\ B_7$
○ B. $B_0\ B_2\ B_3\ B_5\ B_6$
○ C. $B_0\ B_8$
○ D. $B_0\ B_1\ B_4\ B_5\ B_7$

2-9 Rod-cutting Problem: Given a rod of total length N inches and a table of selling prices P_L for lengths $L = 1, 2, \cdots, M$. You are asked to find the maximum revenue R_N obtainable by cutting up the rod and selling the pieces. For example, based on the following table of prices, if we are to sell an 8-inch rod, the optimal solution is to cut it into two pieces of lengths 2 and 6, which produces revenue $R_8 = P_2 + P_6 = 5 + 17 = 22$. And if we are to sell a 3-inch rod, the best way is not to cut it at all. (4分)

Length L	1	2	3	4	5	6	7	8	9	10
Price P_L	1	5	8	9	10	17	17	20	23	28

Which one of the following statements is FALSE?

○ A. This problem can be solved by dynamic programming
○ B. The time complexity of this algorithm is $O(N^2)$
○ C. If $N \le M$, we have $R_N = \max\{P_N, \max_{1 \le i < N}\{R_i + R_{N-i}\}\}$
● D. If $N > M$, we have $R_N = \max_{1 \le i < N}\{R_i + R_{N-M}\}$

2-10 In dynamic programming, we derive a recurrence relation for the solution to one subproblem in terms of solutions to other subproblems. To turn this relation into a bottom up dynamic programming algorithm, we need an order to fill in the solution cells in a table, such that all needed subproblems are solved before solving a subproblem. Among the following relations, which one is impossible to be computed? (4分)

○ A. $A(i,j) = min(A(i-1,j), A(i,j-1), A(i-1,j-1))$
○ B. $A(i,j) = F(A(min\{i,j\}-1, min\{i,j\}-1), A(max\{i,j\}-1, max\{i,j\}-1))$
○ C. $A(i,j) = F(A(i,j-1), A(i-1,j-1), A(i-1,j+1))$
● D. $A(i,j) = F(A(i-2,j-2), A(i+2,j+2))$

程序填空题

得分：暂无　总分：40

5-1 In a permutation A, if there exists a pair of numbers A_i and A_j satisfied $i < j$ and $A_i > A_j$, then A_i and A_j are called an inverted pair.

Giving a permutation of N, please find the number of the inverted pairs in it.

Hint: The function $work(l, r)$ returns the number of inverted pairs in $A_lA_{l+1} \dots A_r$ and makes $A_lA_{l+1} \dots A_r$ being sorted in increasing order.

```
#include <algorithm>
#include <cstdio>
using namespace std;
const int N = 100010;
int tmp[N], a[N];
long long work(int l, int r) {
    if (l == r)
        return 0;
    int mid = (l + r) >> 1;
    long long res = 0;
    res += work(l, mid);
    res += work(mid+1,r)          (4分);
    int t1 = l, t2 = mid + 1, tt = l;
    while (t1 <= mid && t2 <= r)          (4分){
        if (a[t1] < a[t2])
            tmp[tt++] = a[t1++];
        else {
            res += mid - t1 + 1          (6分);
            tmp[tt++] = a[t2++];
        }
    }
    while (t1 <= mid)
        tmp[tt++] = a[t1++];
    while (t2 <= r)
        tmp[tt++] = a[t2++];
    for (int i = l; i <= r; i++)
        a[i]=tmp[i]          (4分);
    return res;
}
int main()
{
    int n;
    scanf("%d", &n);
    for(int i = 1; i <= n; i++)
        scanf("%d", &a[i]);
    printf("%lld\n", work(1, n)          (2分));
    return 0;
}
```

5-2 An n-digit number that is the sum of the n-th powers of its digits is called an n-narcissistic number. For example, 153 is a 3-narcissistic number since $153 = 1^3 + 5^3 + 3^3$, and 1634 is a 4-narcissistic number since $1634 = 1^4 + 6^4 + 3^4 + 4^4$.

Please complete the following program that prints the sum of all n-narcissistic numbers.

```
#include <cstdio>
long long cost[10], ans;
int cnt[10], t[10], n;
void dfs(int rest, int now, long long current) {
    if (rest == 0) {
        long long temp = current;
        for (int i = 0; i < 10; i++)
            t[i] = 0;
        while (temp > 0) {
            ++t[temp % 10];
            temp /= 10;
        }
        bool flag = 1;
        for (int i = 0; i < 10; i++)
            if (cnt[i] != t[i]) {
                flag = 0;
                break;
            }
        if (flag) {
            ans += current          (4分);
        }
        return;
    }
    if ( now > 9          (4分)) {
        return;
    }
    for (cnt[now] = 0; cnt[now] <= rest; cnt[now]++)
        dfs(rest - cnt[now], now + 1,  current + cost[now] * cnt[now]          (4分));
    cnt[now]          (4分) = 0;
}
int main()
{
    scanf("%d", &n);
    for (int i = 0; i < 10; i++) {
        cost[i] = 1;
        for (int j = 0; j < n; j++)
            cost[i] *= i;
    }
    dfs( n, 0, 0          (4分));
    printf("%lld\n",ans);
    return 0;
}
```