

中山大学数据科学与计算机学院本科生实验报告

课程名称：算法设计与分析 任课教师：张子臻

年级	2017级	专业（方向）	软件工程
学号	17343100	姓名	仕润昊
电话	13280152626	Email	1056627011@qq.com
开始日期	2019/3/15	完成日期	2019/3/15

1.实验题目

Sicily上共完成10道题目：1020,1021,1027,1035,1046,1051,1198,1093,1438,1783

Detail of r17343100			List of solved problems		
Nickname			Compare	<input type="text" value="r17343100"/>	and <input type="text" value=""/>
Signature			<input type="button" value="GO"/>		
Rank	5514		1020	1021	1027
Solved No.	16		1035	1046	1051
			1093	1154	1198
			1351	1438	1628
			1641	1775	1783
			1939		

✓	1020	Big Integer	4345	12671	34.29
✓	1021	Couples	6131	32827	18.68
✓	1027	MJ, Nowhere to Hide	4940	15225	32.45
✓	1035	DNA matching	3885	12405	31.32
✓	1046	Plane Spotting	2599	7036	36.94
✓	1051	<u>Biker's Trip Odomete</u>	2750	5174	53.15
✓	1198	Substring	5081	17390	29.22
✓	1093	Air Express	2940	9348	31.45
✓	1438	Shopaholic	2792	5741	48.63
✓	1783	Large is Better	902	1687	53.47

- ① 1020 Big Integer
- ② 1021 Couple
- ③ 1027 MJ, Nowhere to Hide
- ④ 1035 DNA matching
- ⑤ 1046 Plane Spotting
- ⑥ 1051 Biker's Trip Odomete
- ⑦ 1198 Substring
- ⑧ 1093 Air Express
- ⑨ 1438 Shopaholic
- ⑩ 1783 Large is Better

1020 Big Integer

题意：输入 n 个整数 b_i ($1 \leq i \leq n$)，以及一个大整数 x ，输出一个 n 元组($x \bmod b_1, x \bmod b_2, \dots, x \bmod b_n$)

约束： $n \leq 100$, $1 < b_i \leq 1000$ ($1 \leq i \leq n$) 大整数 x 的位数 $m \leq 400$ 并且非负

可以通过公式或者简单的除法

1021 Couple

题意： n 对夫妇站成一个圈,如果一对夫妇相邻,那么可以将他们消去,问最终是否可以把所有夫妇都消掉

约束： $n \leq 100000$ ，多组数据

简单的贪心

1027 MJ, Nowhere to Hide

题意：给出 n 个BBS_ID以及对应的IP_Address，对于每个IP_Address都有两个BBS_ID，其中先出现的称为main_ID，后出现的称为MJ_ID，现在需要找到MJ_ID和main_ID的对应关系

约束： $0 \leq n \leq 20$ 且 n 为偶数，多组数据，以 n 为0结束

注意：

1) 每组样例的后面都有一个空行（包括最后一个样例）

2) n 为0只代表输出结束，而不是最后一个样例

可以通过map // try again

1035 DNA matching

题意：给 n 个DNA单链，问最多能组成多少对DNA双链，每一个DNA单链只能使用一次，并且不能够翻转。两个DNA单链能够形成一对DNA双链的条件是对应位置A/T，C/G配对

约束： $1 \leq n \leq 100$ ，DNA单链的长度 $m \leq 100$ ，多组数据

贪心，用multiset 加速

1046 Plane Spotting

题意：题目意思较为复杂需要认真看题

给出长度为 n 的飞机出现次数的序列 p_i （ p_i 表示第 i 时刻，出现 p_i 个飞机）

现在需要选择一个长度至少为 min_quarter 的连续的区间，即连续子序列，或者子串。

两个区间的优劣判断方式为：

- 1) 平均每时刻飞机数多的更优
- 2) 如果平均每时刻飞机数相同，那么区间长度更长的更优
- 3) 如果平均每时刻飞机数相同，且区间长度相同，那么区间开始时间更前的更优

现在要求前 k 好的区间

约束： $1 \leq n \leq 300$ ， $1 \leq \text{min_quarter} \leq 300$ ， $1 \leq p_i \leq 200$ ($1 \leq i \leq n$)

1051 Biker's Trip Odomete

题意：给车前轮直径，转圈数和时间，求自行车行驶距离和平均速度

单位换算：

$\text{Pi} = 3.1415927$

$5280 \text{ feet} = 1 \text{ mile}$

$12 \text{ inches} = 1 \text{ foot}$

$60 \text{ minutes} = 1 \text{ hour}$

$60 \text{ seconds} = 1 \text{ minute}$

$201.168 \text{ meters} = 1 \text{ furlong}$

1198 Substring

题意：给 n 个字符串，现在需要将他们拼接起来形成一个字典序最小的字符串

约束： $1 \leq n \leq 8$

1093 Air Express

题意：给出4个重量区间和各个区间的单位重量运输价，问对于一个背包，需要添加多少重量使得运输代价最小。

约束：所有的数都为正数，且不超过1000

1438 Shopaholic

题意：有个购物狂在商场中购物，他想要买n个商品，商场在做促销，每买三个商品，最便宜的一个可以免费，现在问最多可以省多少钱。

约束： $n \leq 2 * 10^4$ $price \leq 2 * 10^4$

1783 Large is Better

题意：给出一个数，你可以对这个数做出以下调整：交换任意两个相邻的数，没有次数限制，交换的两个数中不能有0，求可以得到的最大的数。

2.实验目的

本次习题主要考察的是算法的思想：1) 枚举 2) 贪心

数据结构在加速算法上有重要意义，比如说map、multiset等

同时也要熟悉相关的函数：next_permutation、sort等

3.程序设计

1020 Big Integer

我们知道，在取模运算中，有如下规则

$$(a + b) \% p = (a \% p + b \% p) \% p \quad (1)$$

$$(a - b) \% p = (a \% p - b \% p) \% p \quad (2)$$

$$(a * b) \% p = (a \% p * b \% p) \% p \quad (3)$$

$$a^b \% p = ((a \% p)^b) \% p \quad (4)$$

一个数字 $abcdef\dots z$ 我们可以分解为

$$abcdef\dots z = ((a * 10 + b) * 10 + c\dots) * 10 + z$$

所以 $abcdef\dots z \% k$ 等价于

$$\begin{aligned} & abcdef\dots z \% k \\ &= (((a * 10 + b) * 10 + c\dots) * 10 + z) \% k \\ &= (((a * 10 \% k + b \% k) * 10 \% k + c \% k\dots) * 10 \% k + z \% k) \% k \end{aligned}$$

所以对一个大数取模，我们可以通过下面的函数求得

```
//str是大数，求str%k,ans即为结果
int ans = 0;
for (int i = 0; i < str.length(); i++) {
    ans = ((ans * 10)% k + (str[i]-'0')% k ) % k;
}
```

1021 Couple

本题类似于括号匹配的问题，我们可以把一个环拆成一个链，按照数字从小到大的顺序入栈，如果栈顶跟刚入栈的数字具有匹配关系（夫妻关系），则出栈。最后只需要检验栈是否为空即可

涉及主要操作的栈的运算如下。

```
stack<int> s;
for (int i = 1; i <= 2*n; i++) {
    if (s.empty()) {
        s.push(i);
    }
    else if(s.top() == r[i]) { // 如果栈顶跟刚入栈的数字具有匹配关系,出栈
        s.pop();
    }
    else {
        s.push(i);
    }
}
```

1027 MJ, Nowhere to Hide

本题的数据量很小，只是简单的匹配IP，可以通过直接二重循环解决。L为字符串最大长度，时间复杂度是 $O(n * m * L)$ 。

```
vector<pair<string, string> > ans;
for (int i = 0; i < n; i++) {
    for (int j = i + 1; j < n; j++) {
        if (v[i].second == v[j].second && flag[i] == false && flag[j] == false) {
            flag[i] = flag[j] = true;
            string temp = v[j].first + " is the MaJia of ";
            ans.push_back(pair<string, string>(temp, v[i].first));
            break;
        }
    }
}
```

当然如果我们认真观察，不难发现，IP具有唯一性，字符串查找用哈希函数是再好不过的了，我们可以直接通过STL中的map函数来进行匹配。这样就降低了时间复杂度，为 $O(n * \log(n) * L)$ 。

1035 DNA matching

这个题是相当简单的一个字符串匹配问题，使用贪心算法即可，并且数据量很小，可以直接用二重循环做，时间复杂度为 $O(N * N * L)$ 。

```
for (int i = 0; i < v.size(); i++) {
    bool flag = true;
    for (int j = i + 1; j < v.size(); j++) {
        if (visit[i] == false && visit[j] == false && complement(v[i], v[j])) {
            count++;
            visit[i] = true;
            visit[j] = true;
        }
    }
}
```

虽然本题可以用二重循环做，但是对于更复杂的情况，我们就需要考虑提升算法效率。我们可以用multiset这个数据结构。

multiset<T> 容器就像 set<T> 容器，但它可以保存重复的元素。这意味我们总可以插入元素，当然必须是可接受的元素类型。默认用 less<T> 来比较元素，但也可以指定不同的比较函数。在元素等价时，它必须返回 false。例如：

```
std::multiset<string, std::greater<string>> words>{"dog", "cat", "mouse"}, std::greater<string>
());
```

这条语句定义了一个以 string 为元素的 multiset，它以 greater<string> 为构造函数的第二个参数。构造函数的第一个参数是一个初始化列表，它为此容器指定了三个初始元素。和 set 一样，如果它的两个元素相等，那么它们就是匹配的。在一个有比较运算符 comp 的 multiset 中，如果表达式 !(a comp b) && !(b comp a) 为 true，那么元素 a 和 b 就是相等的。

使用multiset可以将程序的复杂度降为 $O(nm \log n)$ 。

1046 Plane Spotting

本题就是一个排序题，需要定义比较函数Cmp，根据题目的复杂规则我们可以定义Cmp函数，可以使用sort函数也可以使用优先队列。

Cmp定义如下。我们可以使用priority_queue<node,vector<node>,Cmp> q; 来建立优先关系。

```
struct Cmp {
    bool operator()(node a, node b) {
        if (a.average < b.average) {
            return true;
        }
    }
}
```

```

else if (a.average > b.average) {
    return false;
}
else {
    if ((a.end - a.from) < (b.end - b.from)) {
        return true;
    }
    else if ((a.end - a.from) > (b.end - b.from)) {
        return false;
    }
    else {
        if (a.end > b.end) {
            return true;
        }
        else{
            return false;
        }
    }
}
}
};

```

1051 Biker's Trip Odomete

本题就是非常非常简单的数学算术题目，最后输出保留精度为2位。

```
cout << fixed << setprecision(2) << ans;
```

1198 Substring

贪心算法，但不是单纯的比较子字符串的字典序大小，而是比较子字符串 $a + b > b + a$ 的大小，决定a和b的相对顺序。

所以我们可以定义结构Cmp，再使用优先队列（也可以用sort），最后出队即可。

相关证明

证明：若 $a <_{cmp} b$, $b <_{cmp} c$, 那么 $a <_{cmp} c$

证：

$$\begin{aligned}
 & a \leq_{cmp} b \quad \&\& \quad b \leq_{cmp} c \\
 \longleftrightarrow & ab \leq ba \quad \&\& \quad bc \leq cb \\
 & \longrightarrow ac \leq ca \\
 \longleftrightarrow & a \leq_{cmp} c
 \end{aligned}$$

```

struct Cmp {
    bool operator()(string a, string b) {

```

```

        return (a + b > b + a);
    }
};

int main() {
    int T;
    cin >> T;
    while (T--) {
        int n;
        cin >> n;
        priority_queue<string, vector<string>, Cmp> q;
        for (int i = 0; i < n; i++) {
            string temp;
            cin >> temp;
            q.push(temp);
        }
        while (!q.empty()) {
            cout << q.top();
            q.pop();
        }
        cout << endl;
    }
}

```

1093 Air Express

本题目就是简单的枚举，每次最多枚举四种情况，比较得出哪种情况花费最少。

```

if (f == 0) {
    temp_cost[0] = cost[0] * n;
    for (int i = 1; i < 4; i++) {
        add_pounds[i] = weight[i - 1] + 1 - n;
        temp_cost[i] = cost[i] * (weight[i-1] + 1);
    }
    int min = temp_cost[0];
    int index = 0;
    for (int i = 1; i < 4; i++) {
        if (min > temp_cost[i]) {
            min = temp_cost[i];
            index = i;
        }
    }
    print(n, min, add_pounds[index]);
}
else if (f == 1) {
    temp_cost[1] = cost[1] * n;
    for (int i = 2; i < 4; i++) {
        add_pounds[i] = weight[i - 1] + 1 - n;
        temp_cost[i] = cost[i] * (weight[i - 1] + 1);
    }
    int min = temp_cost[1];
}

```



```

    int index = 1;
    for (int i = 2; i < 4; i++) {
        if (min > temp_cost[i]) {
            min = temp_cost[i];
            index = i;
        }
    }
    print(n, min, add_pounds[index]);
}
else if (f == 2) {
    temp_cost[2] = cost[2] * n;
    for (int i = 3; i < 4; i++) {
        add_pounds[i] = weight[i - 1] + 1 - n;
        temp_cost[i] = cost[i] * (weight[i - 1] + 1);
    }
    int min = temp_cost[2];
    int index = 2;
    for (int i = 3; i < 4; i++) {
        if (min > temp_cost[i]) {
            min = temp_cost[i];
            index = i;
        }
    }
    print(n, min, add_pounds[index]);
}
else{
    print(n, cost[3] * n, 0);
}

```

1438 Shopaholic

可以直接使用贪心算法，将所有花费从大到小排序，然后从第三个数开始，隔两个取一个，将所有取出数字相加即为最省结果。

可以直接使用sort函数，将结构体cmp（也可以用greater）作为参数。

```

bool cmp(int a, int b)
{
    return a>b;
}

```

```

sort(v, v + n ,cmp);
int sum = 0;
if (n < 3) {
    cout << 0 << endl;
    continue;
}
for (int i = 2; i < n; i += 3) {
    sum += v[i];
}
cout << sum << endl;

```

1783 Large is Better

这个题目就是以0为分隔，对每个子字符串进行从大到小的排序。

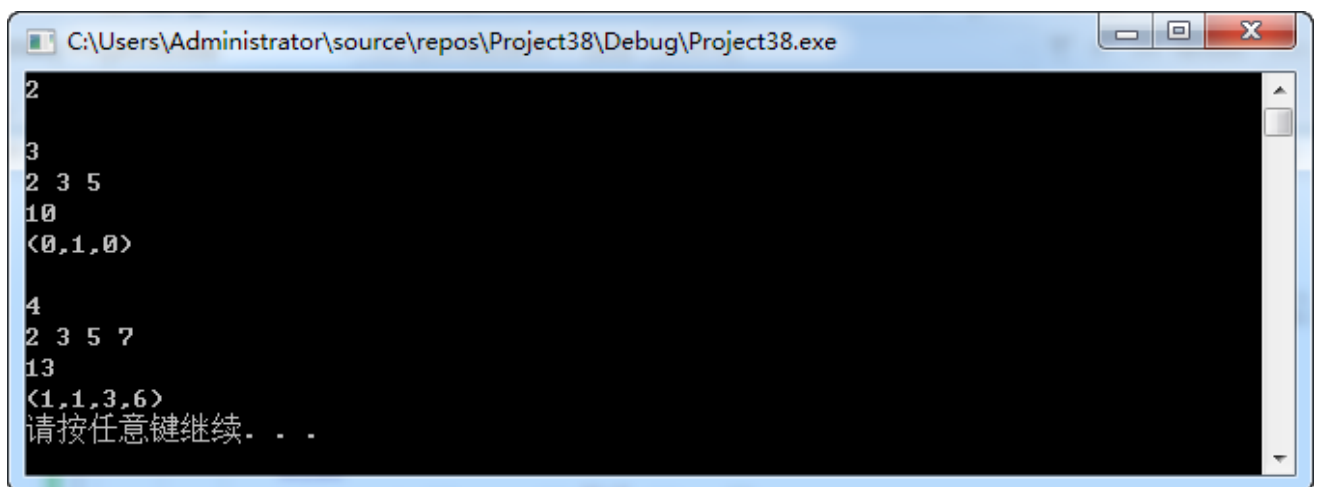
```

for (int i = 0; i < num_arr.length(); i++) {
    if (num_arr[i] == '0') {
        if (count != 0) {
            num.push_back(num_arr.substr(i - count, count));
            num.push_back(num_arr.substr(i, 1));
        }
        else {
            num.push_back("0");
        }
        count = 0;
    }
    else {
        count++;
    }
}
num.push_back(num_arr.substr(num_arr.length() - count, count));
for (int i = 0; i < num.size(); i++) {
    sort(num[i]);
}

```

4.程序运行与测试

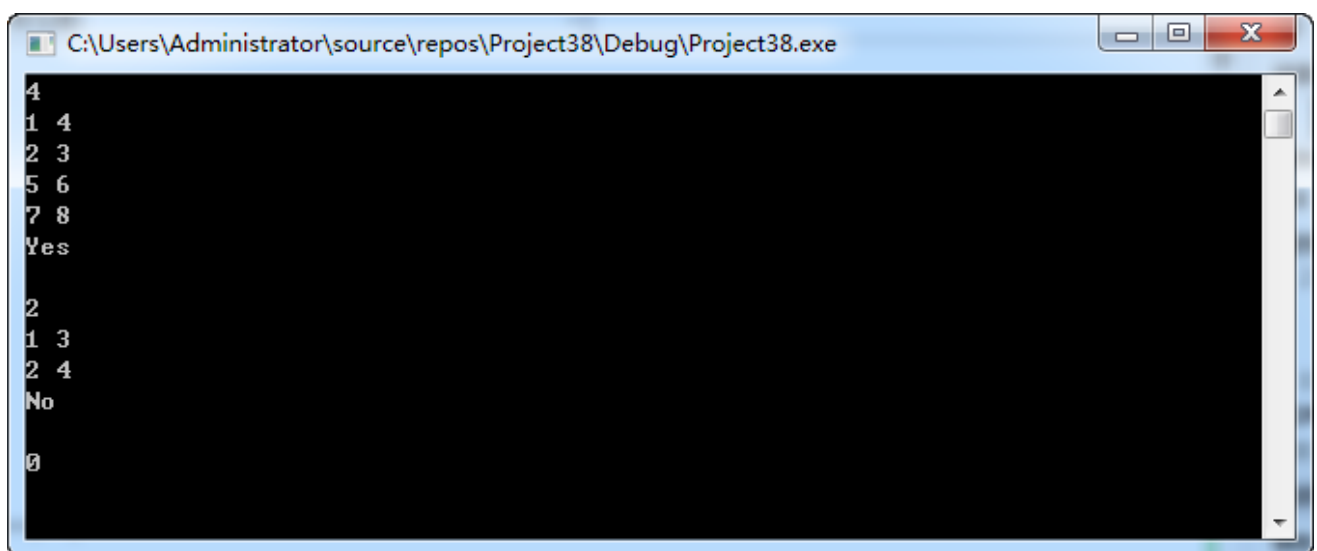
1020 Big Integer



```
C:\Users\Administrator\source\repos\Project38\Debug\Project38.exe
2
3
2 3 5
10
(0,1,0)

4
2 3 5 7
13
(1,1,3,6)
请按任意键继续...
```

1021 Couple



```
C:\Users\Administrator\source\repos\Project38\Debug\Project38.exe
4
1 4
2 3
5 6
7 8
Yes

2
1 3
2 4
No

0
```

1027 MJ, Nowhere to Hide

```
C:\Users\Administrator\source\repos\Project38\Debug\Project38.exe

8
inkfish 192.168.29.24
zhi 192.168.29.235
magicpig 192.168.50.170
pegasus 192.168.29.235
iamcs 202.116.77.131
finalBob 192.168.29.24
tomek 202.116.77.131
magicduck 192.168.50.170
tomek is the MaJia of iamcs
finalBob is the MaJia of inkfish
magicduck is the MaJia of magicpig
pegasus is the MaJia of zhi

4
mmmmmm 172.16.72.126
kkkkkk 192.168.49.161
llllll 192.168.49.161
nnnnnn 172.16.72.126
llllll is the MaJia of kkkkkk
nnnnnn is the MaJia of mmmmmm

0
```

1035 DNA matching

```
C:\Users\Administrator\source\repos\Project38\Debug\Project38.exe

2
3
ATCG
TAGC
TAGG
1
2
AATT
ATTA
0
请按任意键继续. . .
```

1046 Plane Spotting

```
C:\Users\Administrator\source\repos\Project38\Debug\Project38.exe
3
10 5 5
1 5 0 2 1 4 2 5 0 2
Result for run 1:
4-8
2-8
6-10
1-8
2-6
10 3 5
10 3 1 4 2 6 3 0 8 0
Result for run 2:
1-6
1-7
1-9
5 5 5
1 2 3 4 5
Result for run 3:
1-5
请按任意键继续. . .
```

1051 Biker's Trip Odomete

```
C:\Users\Administrator\source\repos\Project38\Debug\Project38.exe
26 1000 5
Trip #1: 1.29 928.20
27.25 873234 3000
Trip #2: 1179.86 1415.84
26 0 1000
```

1198 Substring

```
C:\Users\Administrator\source\repos\Project38\Debug\Project38.exe
1
3
a
ab
ac
aabac
请按任意键继续. . .
```

1093 Air Express

```
C:\Users\Administrator\source\repos\Project38\Debug\Project38.exe

9 10
49 5
99 3
2
Set number 1:
8
Weight <8> has best price $50 <add 2 pounds>
10
Weight <10> has best price $50 <add 0 pounds>
90
Weight <90> has best price $200 <add 10 pounds>
100
Weight <100> has best price $200 <add 0 pounds>
200
Weight <200> has best price $400 <add 0 pounds>
0

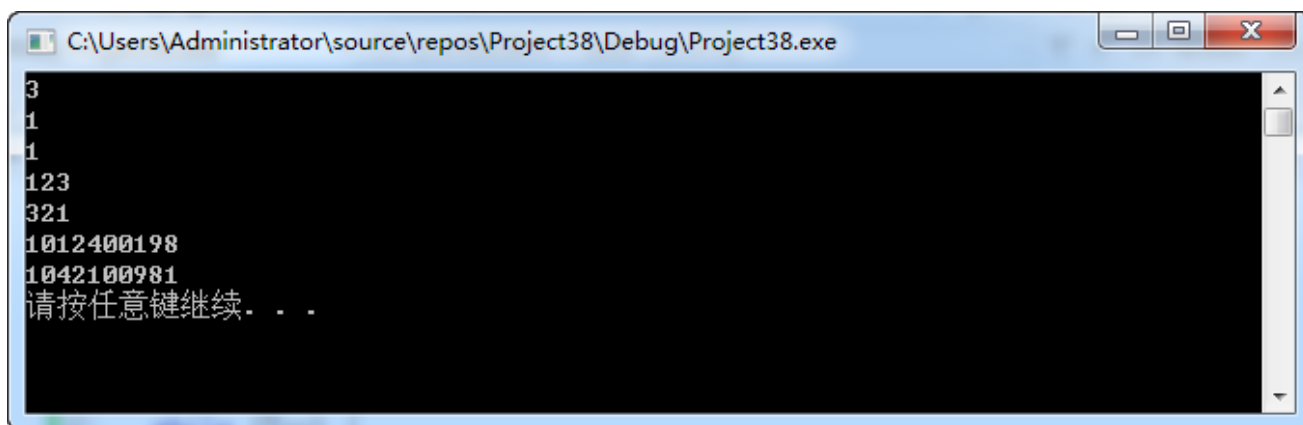
10 10
20 20
30 30
100
Set number 2:
1
Weight <1> has best price $10 <add 0 pounds>
12
Weight <12> has best price $240 <add 0 pounds>
29
Weight <29> has best price $870 <add 0 pounds>
50
Weight <50> has best price $5000 <add 0 pounds>
0
```

1438 Shopaholic

```
C:\Users\Administrator\source\repos\Project38\Debug\Project38.exe

1
6
400 100 200 350 300 250
400
请按任意键继续. . .
```

1783 Large is Better



5.实验总结与心得

本次实验主要目的是练习枚举与贪心算法，算是比较简单的练习。通过这十道题目，我对贪心算法有了更深一步的认识，并且更加熟练地掌握了STL中的相关函数，比如贪心经常用的priority_queue、sort、map、multiset等等，对算法复杂度的分析也有了更深一步的认识。

在刚学程序设计的时候，如果让我做一道大数求模的题我可能会手足无措，但是现在我已经有了算法分析的基本思路，可以比较轻松的解决这类问题。同时，也加深了我对算法的效率问题的理解，可以逐步分析，将算法的复杂度逐渐降低，对今后的程序设计有很大的帮助。

附录、提交文件清单

```
// sicily 1093
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <queue>
#include <stack>
#include <cstring>
#include <iomanip>
using namespace std;

int v[20010];

bool cmp(int a, int b)
{
    return a>b;
}

int main() {
    int T;
    cin >> T;
    while (T--) {
        int n;
        cin >> n;
        for (int i = 0; i < n; i++) {
            cin >> v[i];
        }
    }
}
```

```

        sort(v, v + n ,cmp);
        int sum = 0;
        if (n < 3) {
            cout << 0 << endl;
            continue;
        }
        for (int i = 2; i < n; i += 3) {
            sum += v[i];
        }
        cout << sum << endl;
    }
}

```

// sicily 1093

```

#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <queue>
#include <stack>
#include <cstring>
#include <iomanip>
using namespace std;

```

```
int weight[3];
```

```
int cost[4];
```

```

int get_min_w(int n) {
    if (n <= weight[0]) {
        return 0;
    }
    else if (n > weight[0] && n <= weight[1]) {
        return 1;
    }
    else if (n <= weight[2]) {
        return 2;
    }
    else {
        return 3;
    }
}

```

```
void print(int w,int p,int add) {
```

```

//  cout << "Weight(" << w << ") has best price $" << p << "(add " << add << " pounds)" << endl;
    cout << "Weight (" << w << ") has best price $"
        << p << " (add " << add << " pounds)\n";
}

```

```
int temp_cost[4];
```

```
int add_pounds[4];
```



```

int main() {
    int count = 0;
    while (cin >> weight[0]) {
        count++;
        cin >> cost[0];
        cin >> weight[1] >> cost[1] >> weight[2] >> cost[2] >> cost[3];
        cout << "Set number " << count << ":" << endl;
        int n;
        while (true) {
            cin >> n;
            if (n == 0) {
                break;
            }
            int f = get_min_w(n);
            for (int i = 0; i < 4; i++) {
                add_pounds[i] = 0;
            }
            if (f == 0) {
                temp_cost[0] = cost[0] * n;
                for (int i = 1; i < 4; i++) {
                    add_pounds[i] = weight[i - 1] + 1 - n;
                    temp_cost[i] = cost[i] * (weight[i-1] + 1);
                }
                int min = temp_cost[0];
                int index = 0;
                for (int i = 1; i < 4; i++) {
                    if (min > temp_cost[i]) {
                        min = temp_cost[i];
                        index = i;
                    }
                }
                print(n, min, add_pounds[index]);
            }
            else if (f == 1) {
                temp_cost[1] = cost[1] * n;
                for (int i = 2; i < 4; i++) {
                    add_pounds[i] = weight[i - 1] + 1 - n;
                    temp_cost[i] = cost[i] * (weight[i - 1] + 1);
                }
                int min = temp_cost[1];
                int index = 1;
                for (int i = 2; i < 4; i++) {
                    if (min > temp_cost[i]) {
                        min = temp_cost[i];
                        index = i;
                    }
                }
                print(n, min, add_pounds[index]);
            }
            else if (f == 2) {
                temp_cost[2] = cost[2] * n;
                for (int i = 3; i < 4; i++) {

```

```

        add_pounds[i] = weight[i - 1] + 1 - n;
        temp_cost[i] = cost[i] * (weight[i - 1] + 1);
    }
    int min = temp_cost[2];
    int index = 2;
    for (int i = 3; i < 4; i++) {
        if (min > temp_cost[i]) {
            min = temp_cost[i];
            index = i;
        }
    }
    print(n, min, add_pounds[index]);
}
else{
    print(n, cost[3] * n, 0);
}
}
cout << endl;
}
}

```

// sicily 1783

```

#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <queue>
#include <stack>
#include <cstring>
#include <iomanip>
using namespace std;

void swap(char & a, char & b) {
    char c = a;
    a = b;
    b = c;
}

void sort(string & a) {
    for (int i = 0; i < a.length(); i++) {
        for (int j = 0; j < a.length() - i - 1; j++) {
            if (a[j] < a[j + 1]) {
                swap(a[j], a[j + 1]);
            }
        }
    }
}

int main() {
    int T;
    cin >> T;

    while (T--) {

```

```

string num_arr;
cin >> num_arr;
vector<string> num;
int count = 0;
for (int i = 0; i < num_arr.length(); i++) {
    if (num_arr[i] == '0') {
        if (count != 0) {
            num.push_back(num_arr.substr(i - count, count));
            num.push_back(num_arr.substr(i, 1));
        }
        else {
            num.push_back("0");
        }
        count = 0;
    }
    else {
        count++;
    }
}
num.push_back(num_arr.substr(num_arr.length() - count, count));

for (int i = 0; i < num.size(); i++) {
    sort(num[i]);
}
for (int i = 0; i < num.size(); i++) {
    cout << num[i];
}
cout << endl;
}
}

```

// sicily 1198

```

#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <queue>
#include <stack>
#include <cstring>
#include <iomanip>
using namespace std;

struct Cmp {
    bool operator()(string a, string b) {
        return (a + b > b + a);
    }
};

int main() {
    int T;

    cin >> T;

```

```

    while (T--) {
        int n;
        cin >> n;
        priority_queue<string, vector<string>, Cmp> q;
        for (int i = 0; i < n; i++) {
            string temp;
            cin >> temp;
            q.push(temp);
        }
        while (!q.empty()) {
            cout << q.top();
            q.pop();
        }
        cout << endl;
    }
}

```

// sicily 1051

```

#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <queue>
#include <stack>
#include <cstring>
#include <iomanip>

```

```
using namespace std;
```

```

void print_in_2_45(double ans) {
    cout << fixed << setprecision(2) << ans;
}

```

```

int main() {
    double d, n, t;
    int count = 0;
    while (true) {
        cin >> d >> n >> t;
        if (n == 0) {
            return 0;
        }
        count++;
        double l = d * 3.1415927 / 12 / 5280 * n;
        double v = l / t * 3600;
        cout << "Trip #" << count << ": ";
        print_in_2_45(l);
        cout << " ";
        print_in_2_45(v);
        cout << endl;
    }
}

```

```

    }
}

// sicily 1046
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <queue>
#include <stack>
#include <cstring>
using namespace std;

int arr[1000];

int cal_sum(int * arr, int from, int end) {
    int sum = 0;
    for (int i = from; i < end; i++) {
        sum += arr[i];
    }
    return sum;
}

struct node {
    int from;
    int end;
    double average;
    node(int from, int end, double average) {
        this->from = from;
        this->end = end;
        this->average = average;
    }
};

struct Cmp {
    bool operator()(node a, node b) {
        if (a.average < b.average) {
            return true;
        }
        else if (a.average > b.average) {
            return false;
        }
        else {
            if ((a.end - a.from) < (b.end - b.from)) {
                return true;
            }
            else if ((a.end - a.from) > (b.end - b.from)) {
                return false;
            }
            else {
                if (a.end > b.end) {
                    return true;
                }
            }
        }
    }
};

```

```

        }
        else{
            return false;
        }
    }
}
};

int main() {
    int T;
    cin >> T;
    int N = T;
    while (T--> {
        int n;
        int q_num;
        int mini_quar;
        cin >> n >> q_num >> mini_quar;
        for (int i = 0; i < n; i++) {
            cin >> arr[i];
        }
        priority_queue<node, vector<node>, Cmp> q;
        for (int i = 0; i < n - mini_quar+1; i++) {
            int this_sum = cal_sum(arr, i, i + mini_quar);
            node temp(i + 1, i + mini_quar, double(this_sum) / double(mini_quar));
            q.push(temp);
            for (int j = i+mini_quar; j < n; j++) {
                this_sum += arr[j];
                node temp2(i + 1, j+1, double(this_sum) / double(j-i+1));
                q.push(temp2);
            }
        }
        cout << "Result for run " << N - T << ":" << endl;
        for (int i = 0; i < q_num && !q.empty(); i++) {
            node temp = q.top();
            q.pop();
            cout << temp.from << "-" << temp.end << endl;
        }
    }
}

```

```

// sicily 1035
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <queue>
#include <stack>
#include <cstring>
using namespace std;

```

```

bool complement(string a, string b) {
    if (a.length() != b.length()) {
        return false;
    }
    for (int i = 0; i < a.length(); i++) {
        if ((a[i] == 'A' && b[i] == 'T') || (a[i] == 'T' && b[i] == 'A') ||
            (a[i] == 'G' && b[i] == 'C') || (a[i] == 'C' && b[i] == 'G')) {

        }
        else {
            return false;
        }
    }
    return true;
}

bool visit[1000];

int main() {
    int T;
    cin >> T;
    while (T--) {
        int n;
        cin >> n;
        vector<string> v;
        for (int i = 0; i < n; i++) {
            string temp;
            cin >> temp;
            v.push_back(temp);
            visit[i] = false;
        }
        int count = 0;
        for (int i = 0; i < v.size(); i++) {
            bool flag = true;
            for (int j = i + 1; j < v.size(); j++) {
                if (visit[i] == false && visit[j] == false && complement(v[i], v[j])) {
                    count++;
                    visit[i] = true;
                    visit[j] = true;
                }
            }
        }
        cout << count << endl;
    }
}

// sicily 1027
#include <iostream>
#include <string>

#include <vector>

```

```

#include <algorithm>
#include <queue>
#include <stack>
#include <cstring>
using namespace std;
bool flag[1000000];

void swap(pair<string, string> &a, pair<string, string> &b) {
    pair<string, string> c;
    c = a;
    a = b;
    b = c;
}

void compare_sort(vector<pair<string, string> > &v) {
    for (int i = 0; i < v.size(); i++) {
        for (int j = 0; j < v.size() - i - 1; j++) {
            if (v[j].second > v[j + 1].second) {
                swap(v[j], v[j + 1]);
            }
        }
    }
}

int main() {
    int n;
    while (true) {
        cin >> n;
        if (n == 0) {
            return 0;
        }
        vector<pair<string, string> > v;
        for (int i = 0; i < n; i++) {
            string temp1, temp2;
            cin >> temp1 >> temp2;
            flag[i] = false;
            v.push_back(pair<string, string>(temp1, temp2));
        }
        vector<pair<string, string> > ans;
        for (int i = 0; i < n; i++) {
            for (int j = i + 1; j < n; j++) {
                if (v[i].second == v[j].second && flag[i] == false && flag[j] == false) {
                    flag[i] = flag[j] = true;
                    string temp = v[j].first + " is the MaJia of ";
                    ans.push_back(pair<string, string>(temp, v[i].first));
                    break;
                }
            }
        }
        compare_sort(ans);
        for (int i = 0; i < ans.size(); i++) {
            cout << ans[i].first << ans[i].second << endl;
        }
    }
}

```



```

        cout << endl;

    }
}

// sicily 1021
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <queue>
#include <stack>
using namespace std;

int r[200010];

int main() {
    int n;
    while (true) {
        cin >> n;
        if (n == 0) {
            return 0;
        }
        for (int i = 0; i < n; i++) {
            int a, b;
            cin >> a >> b;
            r[a] = b;
            r[b] = a;
        }
        stack<int> s;
        for (int i = 1; i <= 2*n; i++) {
            if (s.empty()) {
                s.push(i);
            }
            else if(s.top() == r[i]) {
                s.pop();
            }
            else {
                s.push(i);
            }
        }
        if (s.empty()) {
            cout << "Yes" << endl;
        }
        else {
            cout << "No" << endl;
        }
    }
}

```

```

// sicily 1020 big integer
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
using namespace std;
//(a + b) % p = (a % p + b % p) % p (1)
//(a - b) % p = (a % p - b % p) % p (2)
//(a * b) % p = (a % p * b % p) % p (3)
//a ^ b % p = ((a % p)^b) % p (4)
int base[1001];
int main() {
    int T;
    cin >> T;
    while (T--) {
        int n;
        cin >> n;
        for (int i = 0; i < n; i++) {
            cin >> base[i];
        }
        string str;
        cin >> str;
        cout << "(";
        for (int k = 0; k < n; k++) {
            int ans = 0;
            for (int i = 0; i < str.length(); i++) {
                ans = ((ans * 10)%base[k] + (str[i]-'0')%base[k] ) % base[k];
            }
            if (k < n - 1)
                cout << ans << ",";
            else
                cout << ans << ")" << endl;
        }
    }
}

```