第一周直播课（上午）
作业讲解

洛谷网校
基础衔接提高计划
—扶苏—

洛谷
www.luogu.com.cn

课前声明

欢迎各位开始洛谷基础-提高衔接计划的学习。

本周为第一周。考虑到由易到难的课程特征，本周的作业内容较简单。

本周作业讲解将会分析每道必做作业题目的思路，并进行部分作业题目的代码讲解。

如果有不懂的其他内容，可以 QQ 群进行提问。

洛谷

课前声明

提问多 ≠ 学的好

【第一周】鸟巢

题目链接 https://www.luogu.com.cn/problem/T476835

给定长度为 n 的数组 $a_1, a_2, \cdots, a_n$，和非负整数 k。

要求从数组中选出一些元素，使得这些元素的最大值和最小值之差 $\leq k$。在这种情况下，求选定元素的最大总价值。

$$\mathbf{1 \leq n \leq 1000}, 0 \leq k \leq 1000, 1 \leq a_i \leq 1000$$

【第一周】鸟巢

解析

注意到 $n \leq 1000$，完全可以接受 $O(n^2)$ 的算法。

很直观的一个想法是，把 $a_1 \cdots a_n$ 由小到大排序，然后选定一个最小值，不断向右试探最大值。

这是很容易使用 $O(n^2)$ 算法实现的。
二维循环，第一维 i 枚举选择哪一个最小值，第二维 j 向右不断试探。
当过程中发现 `a[j] > a[i] + k`，则把第二维 `break` 掉。否则不断更新答案即可。

名词解释-更新答案：在算法执行过程中，对当前已经找到的答案（通常存在一个变量里）进行修改或替换。

【第一周】鸟巢

代码实现

```
int main() {
    cin >> n >> k;
    for (int i = 1; i <= n; ++i) {
        cin >> a[i];
    }
    sort(a + 1, a + n + 1);
    int ans = 0;
    for (int i = 1; i <= n; ++i) {
        int cur = 0;
        for (int j = i; a[j] - a[i] <= k && j <= n; ++j) {
            cur += a[j];
        }
        ans = max(ans, cur);
    }
    cout << ans << endl;
}
```

【第一周】植物园

题目链接 https://www.luogu.com.cn/problem/T476836

给定 $n \times m$ 的 01 矩阵，找到所有满足「0 的数量和 1 的数量相同」的子矩阵中面积最大的那个，输出它的面积。

$$1 \leq n, m \leq 10$$

【第一周】植物园

解析

注意到 $n, m \leq 10$，因此很可能不需要太高深的算法，直接暴力做就可以。

常用的在二维矩阵中暴力枚举某个子矩阵的套路：枚举这个矩阵的左上顶点 (x_i, y_i) 和右下顶点 (x_j, y_j)。这个步骤可以很轻松地使用一个四重循环做到。

之后再加上一个二重循环 $y: x_i \sim x_j, y: y_i \sim y_j$，统计这个矩形内 `0` 和 `1` 各有多少即可。如果个数相同，则尝试更新答案。

左上顶点 (x_i, y_i) 和右下顶点 (x_j, y_j) 描绘的矩阵的面积：

$$(x_j - x_i + 1) \times (y_j - y_i + 1)$$

【第一周】植物园

代码实现

```
int check(int xi, int yi, int xj, int yj) {
    int cnt0 = 0, cnt1 = 0;
    for (int x = xi; x <= xj; x++) {
        for (int y = yi; y <= yj; y++) {
            if (a[x][y] == '0') cnt0++;
            else cnt1++;
        }
    }
    return cnt0 == cnt1;
}

int ans = 0;
for (int xi = 1; xi <= n; xi++)
    for (int yi = 1; yi <= m; yi++)
        for (int xj = xi; xj <= n; xj++)
            for (int yj = yi; yj <= m; yj++) {
                if (check(xi, yi, xj, yj)) {
                    ans = max(ans, (xj - xi + 1) * (yj - yi + 1));
                }
            }
```

【第一周】三子棋

题目链接 https://www.luogu.com.cn/problem/T476837

给定 $n \times n$ 棋盘，判断其中是否有一种棋子在行、列、斜线（包括左上-右下和右上-左下）上连续连成了 3 个。

$$1 \leq n \leq 30$$

【第一周】三子棋

解析

对这样的图案判断题，最好的办法是找好图案的基准点。基准点一旦找好便不再变动，后面遍历所有格子都只用这一个基准点计算。

对于本题而言，最好的基准点是三个棋子中间的那一个。因此我们只需要枚举 $n \times n$ 棋盘中的每一个点，并假设这个点就是（可能存在的）三个棋子中间的那个。此时，便只需判断四个方向上是否有其他的两枚棋子。

- 上一格、下一格
- 左一格、右一格
- 左上一格、右下一格
- 右上一格、左下一格

【第一周】三子棋

代码实现

```
int check(int x, int y) {
    char p = a[x][y];
    if (a[x - 1][y] == p && a[x + 1][y] == p) return 1;
    if (a[x][y - 1] == p && a[x][y + 1] == p) return 1;
    if (a[x - 1][y - 1] == p && a[x + 1][y + 1] == p) return 1;
    if (a[x - 1][y + 1] == p && a[x + 1][y - 1] == p) return 1;
    return 0;
}

for (int i = 1; i <= n; i++)
    for (int j = 1; j <= n; j++) {
        if (a[i][j] != '.' && check(i, j)) {
            cout << a[i][j] << endl;
            return 0;
        }
    }
cout << "ongoing" << endl;
```

【第一周】数论

题目链接 https://www.luogu.com.cn/problem/T476841

给定正整数 n，每次取其末位，平方取个位后再放入 n 的头部。求 q 次操作中是否能将其变为 m。

$$1 \le n, m \le 10^9, 1 \le q \le 10^6$$

【第一周】数论

解析

用数字去存 n 和 m，可能在塞入头部的部分比较困难。我们把 n 和 m 都存成字符串就好了。

如果要把数字 x 塞入 s 的头部，可以写：

```
s = std::to_string(x) + s
```

参考资料：https://zh.cppreference.com/w/cpp/string/basic_string/to_string

【第一周】数论

代码实现

```
int count_bit(lint n) {
    if (n == 0) return 1;
    int cnt = 0;
    while (n) { cnt++; n /= 10; }
    return cnt;
}

int count_bit(lint n) { return to_string(n).size(); }

for (int i = 1; i <= q; i++) {
    lint x = n % 10;
    n /= 10, x *= x, x %= 10;
    n = x * pow(10, count_bit(n)) + n;
    // cout << n << endl;
    if (n == m) {
        flag = 1;
        break;
    }
}
```

【第一周】巧克力工厂

题目链接 https://www.luogu.com.cn/problem/T476842

给定一个整数 K，要求仅选择一个 2 的幂的整数 D。可以将 D 进行若干次平分（每次平分 D 或 $\frac{D}{2}$ 或 $\frac{D}{4}$…）。最终需要将部分拆分出的数进行组合形成 K，求 D 的最小值和最少的拆分次数。

$$1 \leq K \leq 10^6$$

【第一周】巧克力工厂

解析

考虑到 D 和其拆分出的数均为二的非负整数幂，因此考虑 K 的二进制表示。

对于第一问，显然 D 需要 $\geq K$，除此之外没有制约条件（D 通过拆分可以组成任何 $\leq D$ 的整数）。因此 D 为 $\geq K$ 的最小的二的非负整数幂。

对于第二问，考虑直接模拟。可以发现，即使一块 D 切成两块 $\frac{D}{2}$，最后的 K 也只会用到这两块 $\frac{D}{2}$ 中的最多一块（否则无需再切）。所以可以不断令 D 除以 2。过程中，如果 $K \geq D$，则让 $K \leftarrow K - D$，直到 K 归零为止。过程中做好切割次数统计即可。

【第一周】巧克力工厂

代码实现

```
lint D = 1;
while (D < K) D *= 2;
cout << D << ' ';
int cnt = 0;
while (K) {
    if (K >= D) {
        K -= D;
    }
    if (K == 0) break;
    cnt++;
    D /= 2;
}
cout << cnt << endl;
```

【第一周】神秘实验

题目链接 https://www.luogu.com.cn/problem/T476844

给定 n 次实验，每次实验有三个数据 A, B, C。要求将这些实验按照以下规则排序：

1. 比较三个变量的总和，总和高者靠前；
2. 如果总和相同，则比较变量 A 和变量 B 的总和，总和高者靠前；
3. 如果仍相同，则比较变量 A 和变量 B 的最大值，最大值高者靠前；
4. 如果仍相同，则实验并列。

需要输出每次实验的排名，如遇 x 实验并列，则它们排名相同，并留空后面的 $x-1$ 个名次。

【第一周】神秘实验

解析

排序题。定义结构体容纳 a, b, c, id。前半部分直接使用自定义函数排序即可。

排序后，另开 rk 数组，用于计算排名。

首先从头遍历结构体数组，并按照 `id` 同步赋值 rk 数组。之后，再从头遍历一次，检查结构体数组中相邻两项的值是否相同，如果相同则将二者的排名也更改为相同。

【第一周】神秘实验

代码实现

```
struct student {
    int id;
    int c, m, e;
} a[10005];

int cmp(student a, student b) {
    if (a.c + a.m + a.e != b.c + b.m + b.e)
        return a.c + a.m + a.e > b.c + b.m + b.e;
    if (a.c + a.m != b.c + b.m) return a.c + a.m > b.c + b.m;
    return max(a.c, a.m) > max(b.c, b.m);
}

int same(student a, student b) {
    if (a.c + a.m + a.e != b.c + b.m + b.e) return 0;
    if (a.c + a.m != b.c + b.m) return 0;
    if (max(a.c, a.m) != max(b.c, b.m)) return 0;
    return 1;
}
```

【第一周】神秘实验

代码实现

```
for (int i = 1; i <= n; i++) {
    cin >> a[i].c >> a[i].m >> a[i].e;
    a[i].id = i;
}
sort(a + 1, a + n + 1, cmp);
for (int i = 1; i <= n; i++) {
    rk[a[i].id] = i;
}
for (int i = 2; i <= n; i++) {
    if (same(a[i], a[i - 1])) {
        rk[a[i].id] = rk[a[i - 1].id];
    }
}
for (int i = 1; i <= n; i++) {
    cout << rk[i] << endl;
}
```

[语言月赛 202402] 陨石

题目链接 https://www.luogu.com.cn/problem/B3938

给定一个农场，里面有 n 间牛棚，每间牛棚有一个初始防御值。接下来会有 m 块陨石在 t 秒后撞击牛棚，每块陨石会减少相应牛棚的防御值。你可以在 t 秒内使用补给增加牛棚的防御值。每次补给增加 1 点防御值，且每次补给需要 1 秒时间。目标是尽可能多地保护牛棚，使其不被破坏。求不被破坏的牛棚的最多数量。

$$1 \le n \le 5 \times 10^3, 1 \le m \le 10^6, 0 \le t \le 10^6$$

[语言月赛 202402] 陨石

解析

对本题而言，先补后撞和先撞后补没有区别。

由于已经知悉接下来的陨石情况，可以先计算，在不补给的情况下，陨石撞击后所有牛棚的防御值。之后将防御值由大到小排序，优先修补防御值大的牛棚即可。

[语言月赛 202402] 陨石

代码实现

```
for (int i = 1; i <= m; ++i) {
    int v;
    cin >> v;
    a[v] -= 2;
}
sort(a + 1, a + n + 1, cmp);
for (int i = 1; i <= n; ++i) {
    if (a[i] > 0) continue;
    if (1 - a[i] > t) break;
    t -= 1 - a[i], a[i] = 1;
}
int ans = 0;
for (int i = 1; i <= n; ++i) {
    if (a[i] > 0) ans = i;
}
```

[语言月赛 202212] 宇宙密码

题目链接 https://www.luogu.com.cn/problem/B3689

给定一个长度为 n 的初始密码 a，允许至多 k 位数字变化，每个变化的数字可以在原基础上增加或减小 1，并且考虑数字的环绕变化（0 减小 1 变为 9，9 增加 1 变为 0）。

找出所有可能的密码，并按从小到大的顺序输出。

$$1 \le n \le 6, 0 \le a \le 10^n, 0 \le k \le n$$

[语言月赛 202212] 宇宙密码

解析

注意到数字至多只可能有 6 位。即，最坏情况下，最后的答案只可能在 $000000 \sim 999999$ 中取得，至多有 10^6 个可能的答案。

直接从 a 推到最后的答案较为困难，可能使用深度优先搜索等超纲算法。然而，验证某一个数 b 是否是最后的答案是比较简单的。

逐个枚举 $x = 0 \sim 10^n - 1$，并逐个判断及输出即可。

判断中，逐个取 x 的十进制位与原数字 a 的对应位进行比较，并取一个初始化为 0 变量 c 记录「变化」的位的数量。比较 $c \le k$ 即可。

[语言月赛 202212] 宇宙密码

代码实现

```
int check(int x) {
    int cnt = 0, var = a;
    for (int i = 1; i <= n; ++i) {
        if (abs((var % 10) - (x % 10)) == 1 || abs((var % 10) - (x % 10)) == 9) {
            ++cnt;
        } else if (abs((var % 10) - (x % 10)) >= 2)
            return 0;
        var /= 10, x /= 10;
    }
    return cnt <= k;
}

cin >> n >> a >> k;
int limit = pow(10, n);
for (int i = 0; i < limit; ++i) {
    if (check(i)) {
        cout << i << endl;
    }
}
```

[语言月赛 202403] 土块

题目链接 https://www.luogu.com.cn/problem/B3948

给定 n 道多选题，每道题有 4 个选项。题目答案用 $n \times 4$ 的 01 矩阵 a 描述，考生作答用 $n \times 4$ 的 01 矩阵 b 描述。

考生可能会涂串行，从某题开始按顺序依次作答所有题目。计算对于从 0 到 $n-1$ 的每个题号作为第一道题时，考生能获得的总分数。

$$1 \le n \le 1000$$

[语言月赛 202403] 土块

解析

假设已经得到考生涂串行的试卷，考虑如何计算每一题的得分：

考虑将考生试卷作答设为 1，答案设为 -1。将两个数组相加，那么 0 表示这个位置是正确的。

一个完全正确的答案数组一定全部为 0，如果数组存在多余的 1，那么说明多涂。否则是部分正确的。

枚举考生作答起点 x，按照题目中给定的 $x,(x+1) \bmod n,\cdots$ 公式，用上面的方法比较 $a[(x+i) \bmod n]$ 和 $b[i]$ 即可。

[语言月赛 202403] 土块

代码实现

```
int check(int a[4], int b[4]) {
    int c[4], sum = 0;
    for (int i = 0; i < 4; i++)
        sum += b[i];
    if (sum == 0) return 0;
    for (int i = 0; i < 4; i++)
        c[i] = a[i] + b[i];
    for (int i = 0; i < 4; i++)
        if (c[i] == -1) return 0;
    for (int i = 0; i < 4; i++)
        if (c[i] == 1) return 3;
    return 6;
}
```

```
cin >> n;
for (int i = 0; i < n; i++)
    for (int j = 0; j < 4; j++)
        cin >> a[i][j];
for (int i = 0; i < n; i++)
    for (int j = 0; j < 4; j++)
        cin >> b[i][j], b[i][j] *= -1;
for (int x = 0; x < n; x++) {
    int ans = 0;
    for (int i = 0; i < n; i++)
        ans += check(a[(x + i) % n], b[i]);
    cout << ans << " ";
}
cout << endl;
```

找筷子

题目链接 https://www.luogu.com.cn/problem/P1469

给定 n 根筷子的长度，其中只有一根筷子是落单的，其他筷子均成对。找出这根落单筷子的长度。

$$1 \leq n \leq 10^7 + 1$$

找筷子

解析

异或，逐个二进制位判断，每个二进制位若相同，则取 0，不同则取 1。

$$0 \oplus 1 = 1, 1 \oplus 0 = 1, 0 \oplus 0 = 0, 1 \oplus 1 = 0$$

C++：x ^ y

注意到异或运算 $\oplus$ 的性质：$x \oplus x = 0$。

由于只有一只筷子落单，其余筷子都是成双的。因此将所有输入的筷子异或起来即可得到答案。

找筷子

代码实现

```
cin >> n;
for (int i = 1; i <= n; ++i) {
    int x;
    cin >> x;
    ans ^= x;
}
cout << ans << endl;
```

[语言月赛 202309] pip install

题目链接 https://www.luogu.com.cn/problem/B3860

给定 N 个编号为 1 到 N 的软件包，每个软件包有若干依赖的软件包。包 i 的依赖关系用 K_i 个整数表示。

需要安装编号为 1 的包及其所有依赖包，求总共需要安装的包的数量。

$$1 \le N \le 5000, 0 \le K_i < N$$

[语言月赛 202309] pip install

解析

使用递归函数和一个 v 数组，其中 $v[i]$ 表示 i 号软件包是否被安装过。

递归函数只有一个参数 x，表示现在要安装 x 号软件包。首先标记 $v[x]=1$，之后查询 x 的所有依赖包。如果未安装，则递归安装。

最后统计 v 中有多少个 1 即可。

[语言月赛 202309] pip install

代码实现

```
void f(int x) {
    v[x] = 1;
    for (int i = 1; i <= k[x]; i++) {
        if (!v[x]) f(a[x][i]);
    }
}

int ans = 0;
for (int i = 1; i <= n; ++i) {
    if (v[i]) ++ans;
}
```

[语言月赛 202210] 军训

题目链接 https://www.luogu.com.cn/problem/B3675

n 行 m 列的同学，每个同学有两个特征值 $a_{i,j}$、$b_{i,j}$。定义一行同学的不整齐度为 $a_{i,j}$ 方差与 $b_{i,j}$ 方差之和。

现在希望对若干行进行交换，使得最后的方阵中，从第 1 行至第 n 行不整齐度依次递增。

求一种次数不超过 n^2 的交换方案。

[语言月赛 202210] 军训

解析

细节繁多，需要仔细处理。

首先，按公式计算出每一行的 $a_{i,j}$ 和 $b_{i,j}$ 方差（分别记作 A_i, B_i），并加和（$C_i = A_i + B_i$）。

考虑使用交换实现序列单调递增的方法。

序列单调递增 ➔ 序列排序 ➔ 使用交换排序 ➔ 冒泡排序

仿照冒泡排序的步骤，输出交换的方案即可。

注意读入时 m, n 顺序不符合常规情况。

[语言月赛 202210] 军训

代码实现

```
cin >> m >> n;
for (int i = 1; i <= n; i++) {
    double S = 0; double aval = 0.0;
    for (int j = 1; j <= m; j++) { cin >> a[i][j]; S += (double)a[i][j]; }
    S /= (double)m;
    for (int j = 1; j <= m; j++)
        aval += ((double)a[i][j] - S) * ((double)a[i][j] - S);
    aval /= (double)m; val[i] += aval;
}
for (int i = 1; i <= n; i++) {
    double S = 0; double bval = 0.0;
    for (int j = 1; j <= m; j++) { cin >> b[i][j]; S += (double)b[i][j]; }
    S /= (double)m;
    for (int j = 1; j <= m; j++)
        bval += ((double)b[i][j] - S) * ((double)b[i][j] - S);
    bval /= (double)m; val[i] += bval;
}
```

[语言月赛 202210] 军训

代码实现

```
for (int i = 1; i <= n; i++) {
    for (int j = 1; j < i; j++) {
        if (val[j] > val[i]) {
            vx.push_back(i);
            vy.push_back(j);
            swap(val[i], val[j]);
        }
    }
}
cout << (int)vx.size() << endl;
for (int i = 0; i < (int)vx.size(); i++) {
    cout << vx[i] << " " << vy[i] << endl;
}
```

浣熊的语言

题目链接 https://www.luogu.com.cn/problem/P9553

给定 n 个单词和每个单词的首次学习时间 d_i、k 个复习点 t_i、m 个特殊情况日期 s_i。

每个单词首次学习后，需要在 $d+t$ 天再复习 k 次。

特殊情况日期不能学习，当天的学习需要延后。如果被耽误的是首次学习，则复习也同步延后；如果被耽误的是复习，则不影响其他结果。

求学习和复习完所有单词需要的总天数，以及每一天的学习和复习单词数量。

$$1 \le n \le 10^6, 0 \le m \le 200, 1 \le k, d, t \le 10^3$$

浣熊的语言

解析

直接模拟相对简单。但时间复杂度 $O(nk)$，会超时。

注意到答案只关心每天学习单词数量，而单词多个复习点重叠不需要进行去重，并且所有单词遗忘曲线相同，所以可使用数组累加答案。

使用两个数组 n, o（代表 new，old）记录每天需要学习的单词数量。

1. 对所有的输入，让 $n_{d_i} \leftarrow n_{d_i} + 1$。
2. 令 $n_{s_i+1} \leftarrow n_{s_i}, n_{s_i} \leftarrow 0$
3. 从头遍历所有天数，令 $j = 1 \sim n$，$o_{i+t[j]} \leftarrow n_i$。
4. 令 $o_{s_i+1} \leftarrow o_{s_i}, o_{s_i} \leftarrow 0$，更新学习量非 0 的最晚天数。

浣熊的语言

代码实现

```
for (int i = 1; i <= n; i++) {
    new_word[d[i]]++;
}
for (int i = 1; i <= m; i++) {
    new_word[s[i] + 1] += new_word[s[i]]; new_word[s[i]] = 0;
}
for (int i = 1; i <= ans; ++i) {
    if (new_word[i] == 0) continue;
    for (int j = 1; j <= k; j++) {
        old_word[i + t[j]] += new_word[i]; ans = max(ans, i + t[j]);
    }
}
for (int i = 1; i <= m; ++i) {
    old_word[s[i] + 1] += old_word[s[i]]; old_word[s[i]] = 0;
}
while (new_word[ans] == 0 && old_word[ans] == 0) ans--;
cout << ans << endl;
for (int i = 1; i <= ans; i++) {
    cout << new_word[i] << " " << old_word[i] << endl;
}
```

导弹拦截

题目链接 https://www.luogu.com.cn/problem/P1158

给定两个导弹拦截系统，分别在 $(x_1, y_1), (x_2, y_2)$。

给定 N 枚导弹。当天可以给两个系统分别设置一个工作半径 r_1, r_2，二者此时可以拦截与其距离为 r_1, r_2 的导弹。

规定总代价为 $r_1^2 + r_2^2$。求为了拦截所有导弹，总代价的最小值。

$1 \le N \le 10^5$，所有坐标分量绝对值不超过 1000。

导弹拦截

解析

对于一个导弹，其一定被拦截系统 1 或 2 拦截。考虑枚举系统 1 拦截的导弹，其余的一定被 2 拦截。

将导弹按照到 1 的距离从小到大排序，从头往后枚举即可。

设当前枚举到第 i 颗导弹，导弹 1 的拦截半径为其到该导弹的距离。

由于导弹已经排序，所以 $1 \sim i$ 的导弹都可以被 1 拦截，剩下的导弹 1 不可能拦截，故全部给 2 拦截。这样便可枚举系统 1 拦截导弹的全部情况，相应地枚举 2 拦截导弹的全部情况。

导弹拦截

解析

对于系统 2，可以考虑做如下优化。

f_i 表示导弹拦截 2 拦截 $i \sim n$ 导弹需要的代价，那么显然

$$f_i = \max\{f_{i+1}, d_i\}$$

d_i 代表拦截 i 导弹的代价。

这样便可在 $O(n \log n)$ 的时间里完成本题。

导弹拦截

代码实现

```
struct Point {
    int x, y;
    Point(int x = 0, int y = 0) { this->x = x, this->y = y; }
} v1, v2, a[100005];
int n; int ans = 0x3f3f3f3f; int f[100005];

int getDist(Point a, Point b) {
    int r1 = a.x - b.x, r2 = a.y - b.y; return r1 * r1 + r2 * r2;
}
int cmp(Point x, Point y) { return getDist(x, v1) < getDist(y, v1); }

cin >> v1.x >> v1.y >> v2.x >> v2.y >> n;
for (int i = 1; i <= n; ++i) cin >> a[i].x >> a[i].y;
sort(a + 1, a + n + 1, cmp);
for (int i = n; i; --i) f[i] = max(f[i + 1], getDist(a[i], v2));
for (int i = 1; i <= n; ++i) ans = min(ans, getDist(a[i], v1) + f[i + 1]);
cout << ans << endl;
```