E1

Problems List

#	Title	Total	AC Rate
1	合并	609	26.60%
2	只有1和2的世界	489	32.92%
3	小花过河	770	20.26%
4	真正的勇士	832	18.27%

E1-1 合并

题面

> **问题描述**
> 小花有一个非递减顺序的数组A，小树有一个非递减顺序的数组B.
> 由于小花和小树的关系很好，他们希望合并手中的数组，得到新的非递减顺序的数组C.
> 请你输出这个数组C.
> **输入描述**
> 第一行包含两个整数N,M，分别表示数组A和数组B的长度。
> 第二行包含N个整数，表示非递减数组A。
> 第三行包含M个整数，表示非递减数组B。
> **输出描述**
> 输出一个数组，长度为N+M，表示合并后的数组。

题目要求

合并两个非递减数组，得到一个新的非递减数组。

思路

通过两个指针，分别指向两个数组的首元素，然后将两个指针指向的元素进行大小的比较，将相对较小的元素放入形成的新队列。时间复杂度$O(n)$

核心代码

```
int a_index = 0, b_index = 0; // 分别表示a数组和b数组的下标
    for(int i=0;i<m+n;i++) {
        if(b_index >= m) {  // 判断b_index的边缘条件
            c.push_back(a[a_index]);
            a_index++;
            continue;
        }
        if(a_index >= n) {  // 判断a_index的边缘条件
            c.push_back(b[b_index]);
            b_index++;
            continue;
        }
        if(a[a_index] <= b[b_index]) {
            c.push_back(a[a_index]);
            a_index ++;
        } else {
            c.push_back(b[b_index]);
            b_index ++;
        }
    }
```

E1-2 只有1和2的世界

题面

> **问题描述**
> 在小花的世界里，只有数字1和2。
> 现在来了一个外星人，它希望得到数字N，它可以通过使用若干个数字1和若干个数字2相加的方法得到数字N。
> 现在它想知道，有多少种不同的方法可以得到数字N?
> **输入描述**
> 输入一个整数$N(1 \leq N \leq 45)$.
> **输出描述**
> 输出一个整数，表示有多少种不同的方法得到数字N.

题目大意

选择若干个数字1和数字2，然后相加得到数字N，求问有多少选择策略。其中，数字1和数字2的位置是需要区分的。

思路

思维题。

设 $f[i]$ 为得到数字i的方法数。

得到递推公式：$f[i] = f[i-1] + f[i-2]$。

表示从和为$i-1$的情况中选择再选择一个数字1得到和为i的情况，以及从和为$i-2$的情况中选择再选择一个数字2得到和为i的情况。

核心代码

```
    f[0] = 0;
    f[1] = 1; // 1
    f[2] = 2; // 1+1和2

    for(int i=3;i<=n;i++) {
        f[i] = f[i-1] + f[i-2];
    }
```

E1-3 小花过河

题面

> **问题描述**
> 小花希望去河岸对面找自己的好朋友小树玩，在两岸之间一共有N块岩石。
> 小花以河岸S为起点，以河对岸E为终点。其中，S、E与河岸中的岩石呈一条直线排列。
> 小花从河岸S出发，每一步跳向相邻的岩石，直到到达河岸E.
> 这一天，小树给小花出了一道难题：如果移去河道中的M块岩石，使得小花在跳跃过程中的最短跳跃距离尽可能长，那么这个最短跳跃距离的最大值应该是多少呢?
> **输入描述**
> 第一行包含一个整数L，表示河岸S到河岸E之间的距离。保证 $1 \leq L \leq 10^9$.
> 第二行包含两个整数N,M，分别表示两岸之间岩石的数量，以及移去的M块岩石数。其中$L \geq N \geq M \geq 0$.

> 接下来一行，包含N个整数，整数$D_i(0 < D_i < L)$表示第i块岩石与河岸S之间的距离。并且满足D_i按照从小到大的顺序给出，并且不会存在$D_i = D_j(i \neq j)$.
> $(0 \leq N, M \leq 10^5)$
> **输出描述**
> 一个整数，表示移去M块岩石后，最短跳跃距离的最大值。

题目大意

在一个宽度为L的河流内，删除M块岩石，使得最短的跳跃距离最大。

思路

- 二分答案
- 本题的答案在一个区间内，并且如果直接进行答案的枚举或者删除方案的枚举，都会造成超时。
- 假设最短的跳跃距离为x，则需要验证是否存在一种删除岩石的方案，使得最短跳跃距离为x。并且，验证可以在$O(n)$的时间复杂度内完成。
    - 验证方法：从左到右遍历所有石头，如果当前石头间的间隔小于x，则删除一块石头。最后检查删除石头的块数是否小于限定的M块。
- 并且，答案的搜索具有单调性。如果跳跃距离为x无法满足删除的方案，则最短跳跃距离的最大值一定小于x。

因此，本题采用二分答案的方法，对答案进行枚举，然后该答案的可行性进行验证。

核心代码

```
bool ok(int x) {
    int pre = 0;
    int cnt = 0;
    for (int i = 1;i<=n;i++) {
        if(a[i]-a[pre]<x) { // 假设距离小于x，则可以删除一个石头，保证间隔大于
            cnt++;
        } else { // 当距离大于x时，开启下一轮判断，判断下一个间隔是否小于x
            pre = i;
        }
    }
    if(cnt<=m) return true;
    return false;
}
```

```
14
15  int l = 0, r = L + 1;
16  int ans = 0;
17  while (l < r) {
18      int mid = (l + r) >> 1;
19      if (ok(mid)) {
20          ans = mid;
21          l = mid;
22      } else {
23          r = mid-1;
24      }
25  }
```

扩展题目

- http://poj.org/problem?id=2018
- https://www.luogu.com.cn/problem/P1873

E1-4 真正的勇士

题面

问题描述

小花要玩一场勇者斗恶龙的游戏。

由于恶龙群体的飞速扩大，她所面对的是N个山洞，第i个山洞里有k_i只恶龙。其中，每个恶龙的生命值都是不同的。当小花进入一个山洞时，她必须按照给定的顺序，依次打败每只恶龙。

并且，当且仅当她的勇气值**严格大于**当前恶龙的生命值时，才能打败该恶龙。每当小花打败一只恶龙，她的勇气值则会+1。

已知，小花可以任意选择进入山洞的顺序，但是进入山洞后需要按照顺序打败恶龙。请问，小花最初最少需要具备多少勇气值，才可以开始这场勇者斗恶龙的游戏?

输入描述

第一行给定一个整数N，表示山洞的数量。

接下来N行，第i行的第一个数字为k_i，表示山洞中恶龙的数量。接下来是k_i个整数，表示山洞中恶龙的生命值。

$(N \leq 10^5, \sum k_i \leq 10^5)$

输出描述

> 输出一个整数，表示小花最初最早需要具备多少勇气值。
> **题目大意**
>
> - 有 n 个洞穴，第 i 个洞穴里有 k 个怪物，每杀死一个怪物，则勇气值就加一。
> - 当且仅当你的勇气值严格大于这个怪物的武力值，才能够杀死这个怪物。
> - 求出杀死所有怪物需要的最小初始勇气值。

思路

- **贪心和模拟**。
- 时间复杂度$O(nlogn)$
- 先求出通关每个洞穴的最小初始值，然后对这些初始值进行排序。则进入洞穴的顺序应该由每个洞穴所需要的最小的初始值单调递增的顺序决定。此时求的是勇气值的局部最小值。
    - 假设洞穴A的最小初始勇气值 < 洞穴B的最小初始勇气值。假设先经过洞穴B，能够通过洞穴B必然能够通过洞穴A。假设先经过洞穴A，那么从洞穴A出去后，增加后的勇气值可能能够通过洞穴B。
- 然后从最小初始值的洞穴开始打怪，模拟打怪的过程即可。
- 本题也可以通过**二分答案**求解。

核心代码

```
for(int i=0;i<n;i++) {
    int k;cin>>k;
    vector<int> b;
    for(int j=0;j<k;j++) {
        int num;cin>>num;
        b.push_back(num);
    }
    int init_heart = 0;
    int heart = init_heart;
    for(int j=0;j<k;j++) {
        if(heart <= b[j]) {
            init_heart = b[j] + 1 - j;
            heart = b[j] + 1;
        }
        heart++;
    }
    a.push_back(b);
    limit.push_back(node(i, init_heart));
}

```

```
sort(limit.begin(), limit.end(), cmp);

int init_heart = 0;
int heart = init_heart;
int cnt = 0;
for(int i=0;i<n;i++) {
    int index = limit[i].index;
    for(int j=0;j<a[index].size();j++) {
        if(heart <= a[index][j]) {
            init_heart = a[index][j] + 1 - cnt;
            heart = a[index][j] + 1;
        }
        heart++;
        cnt++;
    }
}
cout<<init_heart<<endl;
```

扩展题目

- http://poj.org/problem?id=3190
- http://poj.org/problem?id=1328
- https://www.luogu.com.cn/problem/P1080

E2

Problems List

#	Title	Total	AC Rate
1	小花的项链	584	26.54%
2	小花打工	327	47.09%
3	小花的圣诞树	392	39.03%
4	小花的糖果堆	470	32.55%

E2-1 小花的项链

题面

> 小花正在和她的好朋友小树一起玩串珠游戏。
> 一段时间后，小花和小树分别完成了一串漂亮的项链。她们分别将自己的项链用一个字符串表示。例如"helloworld"和"imfinethankyouandyou".
> 小花提出了用"最长公共子序列"来衡量她们之间的默契值。并且，小花对"子序列"做出了解释：由原字符串在不改变字符的相对顺序的情况下删除若干个字符后组成的新字符串。
> **输入格式**
> 第一行包含一个数字N和一个字符串S。N表示字符串S的长度。
> 第二行包含一个数字M和包含一个字符串T。M表示字符串T的长度。
> $(0 \leq N, M \leq 10^4)$
> S和T字符串中仅包含大写和小写英文字符$A \sim Z, a \sim z$
> **输出格式**
> 一个整数，表示两个字符串的最长公共子序列。

思路

- 二维动态规划
- 时间复杂度$O(n^2)$
- 求两个字符串最长的公共子序列。
- 令 `dp[i][j]` 为 `text1[0...i]` 和 `text2[0...j]` 的最长公共子序列的长度。转移公式如下：
  - `dp[i][j] = dp[i-1][j-1] + 1 (if text1[i] == text2[j])`
  - `dp[i][j] = max(dp[i][j-1], dp[i-1][j]) (if text1[i] != text2[j])`

核心代码

```
for(int i=1;i<=n;i++) {
    for(int j=1;j<=m;j++) {
        char s_temp = s[i-1];
        char t_temp = t[j-1];
        if(s_temp == t_temp) {
            dp[i][j] = dp[i-1][j-1] + 1;
        } else {
            dp[i][j] = max(dp[i-1][j], dp[i][j-1]);
        }
    }
}
cout<<dp[n][m]<<endl;
```

扩展链接

- https://lcs-demo.sourceforge.net/
- https://oi-wiki.org/dp/basic/?query=%E6%9C%80%E9%95%BF%E5%85%AC%E5%85%B1%E5%AD%90%E5%BA%8F%E5%88%97

E2-2 小花打工

题面

小花在课余时间做一些兼职的工作。由于老板比较抠门，不允许她连续工作两天及以上天数。并且，每天的工资都会随着市场行情而变化。假设小花已经通过自己的预知能力知道了接下来N天内的兼职工资，她需要在不连续工作的情况下，让自己在接下来N天所获得的工资最大化，请你算出她最多能拿到的工资。

输入格式

第一行包含一个整数N

第二行包含N个正整数，第i个整数表示为a_i，a_i表示第i天小花能够拿到的工资。

$(0 \leq N \leq 10^6, 0 \leq a_i \leq 10^4)$

输出格式

输出一个正整数，表示小花最多能拿到的工资金额。

问题描述

- 动态规划
- 在一个序列里，在不能连续选择的情况下，选择一些元素，使得这些的和最大。

思路

- 线性动态规划
- 时间复杂度$O(n)$
- 令$dp[i][0/1]$为第i天时最大的工资，当第二维为1时，表示第i天选择了工作，当第二维为0时，表示第i天不工作。转移公式如下：
  - $dp[i][0] = max(dp[i-1][0], dp[i-1][1])$
  - $dp[i][1] = dp[i-1][0] + w[i]$

核心代码

```
for(int i=0;i<n;i++) {
    if(i == 0) {
        dp[i][0] = 0;
        dp[i][1] = nums[i];
    } else {
        dp[i][0] = max(dp[i-1][1], dp[i-1][0]);
        dp[i][1] = dp[i-1][0] + nums[i];
    }
}
cout<<max(dp[n-1][0], dp[n-1][1]);
```

扩展题目

E2-3 小花的圣诞树

题面

> 圣诞节快到了，小花正在装饰自己的圣诞树。她将这棵圣诞树抽象为一个树状结构，她可以在这个树上的每个节点放置装饰品。
> 在这棵圣诞树上，一共有N个节点。在第i个节点上放置装饰品，可以得到a_i点漂亮值。但是，小花不希望父节点和子节点都挂上装饰品。（父节点与子节点相邻）。
> 请问，小花应该怎么挂装饰品，才能使得整棵圣诞树的漂亮值总值最大?

输入格式
第一行包含一个整数N，表示圣诞树的节点数。
第二行包含N个整数，表示每个节点挂上装饰品后能够获得的装饰品a_i
接下来包含$N-1$行，每行包含两个整数$$x_j，y_j$$，其中y_i是x_i的父节点。
$(1 \leq N \leq 10^3, -128 \leq a_i \leq 127, 1 \leq x_j, y_j \leq N)$
输出格式
输出一个正整数表示整棵圣诞树能够获得的最大漂亮值。

问题描述

- 在一个树状结构中选择元素，其中父节点和子节点不能同时被选择，求所有选择的元素和的最大值。

思路

- 树形dp
- 需要找出根节点。
- 令 `dp[i][0/1]` 表示以节点 `i` 为根结点的子树的最大元素和。其中第二维为0表示不选择节点 `i` ，第二维为1表示选择节点 `i` . 转移公式如下：
  - `dp[i][0] += max(dp[j][0], dp[j][1])` （j为i的子结点）
  - `dp[i][1] += dp[j][0]` （j为i的子结点）

核心代码

```
void dfs(int u) {
    dp[u][0] = 0;
    dp[u][1] = r[u];
    for(int v:e[u]) {
        dfs(v);
        dp[u][0] += max(dp[v][0], dp[v][1]);
        dp[u][1] += dp[v][0]; // u节点选择了，则v节点必然不能选择
    }
    return;
}
```

扩展题目

- HDU 2196 Computer
- POJ 1463 Strategic game
- POI2014 FAR-FarmCraft
- 「SDOI2017」苹果树
- https://www.luogu.com.cn/problem/P2014

E2-4 小花的糖果堆

题面

> 万圣节就要到了，小花采购了很多很多糖果。N个糖果堆排成一排，第i堆含有c_i颗糖果。
> 小花突发奇想，如果她将两堆相邻的糖果堆合并，并且将每次合并后的新糖果堆的糖果数加入总得分，最终，将这一排N个糖果堆合并为一个糖果堆，那么她选择怎样的合并策略，可以让她的总得分最小呢?
> 输入格式
> 第一行包含一个正整数N
> 第二行包含N个整数，第i个整数表示第i堆的糖果数c_i
> $(2 \leq N \leq 200, 1 \leq c_i \leq 1000)$
> 输出一个整数，表示小花将糖果堆合并后能够得到的最小总分。

问题描述

- 相邻的两个节点合并，对答案的贡献为两个节点相加后的值。希望得到最小的答案。

思路

- 区间dp
- 枚举区间 `[i, j]` 内，两个子区间合并的的代价。
- 枚举的顺序：从小到大枚举区间的长度，使这个区间被分为 `[i,k]` 和 `[k+1,j]` 两个区间，取一遍最小值加上合并的即为当前区间所求。
- 令 `dp[i][j]` 表示区间 `[i,j]` 的最小价值。转移公式如下：
- `dp[i][j]=min(dp[i][j],dp[i][k]+dp[k+1][j]+sum[j]-sum[i-1])`
- 至于合并的代价用前缀和表示，容易得到区间 `[i, j]` 之间的元素和。

核心代码

```
 for(int len = 2; len<=n; len++) {
    for(int l = 1; l + len - 1 <= n; l++) {
        int r = l + len - 1;
        for(int k = l; k < r; k++) {
            dp[l][r] = min(dp[l][r], dp[l][k] + dp[k+1][r] +
sum[r] - sum[l-1]);
        }
    }
}
cout<<dp[1][n]<<endl;
```

扩展题目

- http://poj.org/problem?id=1179
- https://www.luogu.com.cn/problem/P1880

E3

Problems List

#	Title	Total	AC Rate
3-1	小花的最小生成树	497	32.19%
3-2	小花小树的美丽邂逅	389	41.13%
3-3	小花的家族	837	18.52%
3-4	小花的图	462	33.98%

E3-1 小花的最小生成树

题面

> 小花有一个N个节点M条边的带权无向图G，她希望输出这张图的最小生成树的各边之和。
> **输入**
> 第一行包含两个正整数N和M。
> 接下来M行，每行包含三个整数x，y，z，表示x和y之间有一条边相连，权重为z。
> 其中，$1 \leq N \leq 5000, 1 \leq M \leq 10^5, 1 \leq z_i \leq 10^4$.
> **输出**
> 输出图G的最小生成树的各边权重和。如果图不连通，则输出-1.

题目大意

- 求出给定图的最小生成树的各边之和

解题思路

- 使用Prim算法或者Kruskal算法求解

核心代码

```
int find(int x) {
    return f[x] == x ? x : f[x] = find(f[x]);
}
bool unionset(int a, int b) {
    a = find(a);
    b = find(b);
    if(a != b) {
        f[b] = a;
        return true;
    }
    return false;
}
// 图
struct node {
    int u, v;
    int val;
} e[maxn];

bool cmp(node a,node b) {
    return a.val < b.val;
}
sort(e+1, e+1+m, cmp);
for(int i=1;i<=m;i++) {
    if(unionset(e[i].u, e[i].v)) {
        total ++;
        sum += e[i].val;
    }
    if(total == n-1) {
        cout<<sum<<endl;
        return 0;
    }
}
```

扩展题目

- https://www.acwing.com/problem/content/description/348/
- http://poj.org/problem?id=2728
- https://www.luogu.com.cn/training/209#problems

E3-2 小花小树的美丽邂逅

题面

> 一年一度的美丽邂逅又在中科大如火如荼地开展了，小花和她的朋友们报名了这场活动。与此同时，小树和他们朋友们也报名了。
> 已知小花和她的朋友们共N人，小树和他的朋友们共M人，小花们和小树们之间有E条连接，假设小花中的一员为x，小树中的一员为y，那么连接(x, y)则表示x和y互有好感。
> 请问，美丽邂逅结束后，小花和她的朋友们和小树和他的朋友们，最多有几对可以成为情侣?
> **输入**
> 第一行输入三个整数N, M, E
> 接下来E行，每行输入两个整数x, y，表示x和y互有好感。
> $(1 \leq N, M, \leq 500, 1 \leq 10^4, 1 \leq x \leq N, 1 \leq y \leq M)$
> **输出**
> 输出一个整数，表示最大匹配数。

题目大意

- 给定一个二分图 G，即分左右两部分，各部分之间的点没有边连接，要求选出一些边，使得这些边没有公共顶点，且边的数量最大。
- 概括：求二分图的最大匹配

解题思路

- 应用匈牙利算法
- 时间复杂度：
- 参考链接：
    - https://oi-wiki.org/graph/graph-matching/bigraph-match/

核心代码

```
bool dfs(int u) {
    for (int v : e[u]){
        if(!vis[v]){
            vis[v] = 1;
            if(match[v] ==0||dfs(match[v])) {
                match[v] = u;
                return true;
            }
        }
    }
    return false;
}
for (int i = 1;i<=n;i++) {
    memset(vis, 0, sizeof(vis));
    if(dfs(i))
        ans++;
}
```

扩展题目

- https://www.luogu.com.cn/problem/P1525

E3-3 小花的家族

题面

> 小花来自于一个古老的大家族，共有N名亲戚。但是有一天，她们家族唯一的一本家谱被虫子啃得支离破碎。
> 小花通过拼凑这本家谱，得到了M条亲属关系的表示。
> 亲属关系表示为x，y，说明y是x的后代。
> 假设没有任何后代的人，是整个家族辈分最低的人。不是任何人的后代的人，是整个家族辈分最高的人。
> 小花提出了一个亲属链的关系，这条链上的起点是辈分最高的人，终点是辈分最低的人，并且，该链表始终满足规律：前一个节点的辈分一定大于后一个节点，并且该辈分关系存在于小花得到的M条关系之内。
> （例子：假设小花得到了关系（1，2），（2，3），则表示2是1的后代，3是2的后代，则它们组成亲属链"1-2-3"，而不能组成亲属链"1-3"）
> 请你给出这本家谱中所有的亲属链的条数，结果对10^9+7取模。

> **输入**
> 第一行包含两个整数N，M
> 接下来M行，每行包含两个整数x，y，表示y是x的后代。
> $(1 \leq N \leq 5000, 1 \leq M \leq 10^5, 1 \leq x, y \leq N)$
> **输出**
> 输出一个整数，表示这本家谱中所有的亲属链的条数

题目大意

- 在一个有向无环图内，找到所有从入度为0的点，到出度为0的点的路径数之和。

解题思路

- 思路1：在拓扑排序的过程中更新路径计数。
  - 对于相邻的点，更新它们的路径计数，当到达一个出度为0的点时，将到达该节点的路径数贡献给总答案。
- 思路2：通过dfs记忆化搜索。

核心代码

```
// 拓扑排序
while(!q.empty()) {
    int u = q.front();
    q.pop();
    for(int v:e[u]) {
        dp[v] = (dp[u] + dp[v]) % mod;
        ind[v]--;
        if(!ind[v]) {
            if(!outd[v]) { // 到达一个出度为0的点
                ans = (ans + dp[v]) % mod;
                continue;
            }
            q.push(v);
        }
    }
}
```

扩展题目

- https://codeforces.com/problemset/problem/1385/E
- https://www.luogu.com.cn/problem/P1038

E3-4 小花的图

题面

> 小花正在学习有趣的图算法，她有一张顶点数为N，边数为M的无向无权图。
> 首先，她学习了最短路算法，然后，她想知道，从顶点1开始，到其他每个点的最短路有几条。
> **输入**
> 第一行包含两个正整数N和M。
> 接下来M行，每行包含两个整数x和y，表示x和y之间有一条边相连。
> 其中，$1 \leq N \leq 10^5, 1 \leq M \leq 10^5, 1 \leq x, y \leq N$.
> **输出**
> 输出$N-1$行整数，表示顶点1到顶点i的最短路的条数。$2 \leq i \leq N$.
> 如果顶点1无法到达顶点i，则输出0.
> 由于答案规模较大，请输出对ans取模$ans \ mod \ 10^9+7$.

题目大意

- 单源最短路问题，求从源点到每个其他的点的最短路的条数。
- 边权为1

解题思路

- 因为所有的边权都为1，所以一个点的最短路就相当于是它在BFS搜索树中的深度。
- 一个点最短路一定经过了一个层数比它少1的结点，否则该路径则不是最短路。
- 所以假设节点 `u` 和节点 `v` 相邻，且 `dis[v] == dis[u] + 1`，此时在最短路上，即可更新路径数。

核心代码

```cpp
queue<int> q;
q.push(1);
dis[1] = 0;
vis[1] = 1;
cnt[1] = 1;
while(!q.empty()) { // bfs搜索
    int u = q.front();
    q.pop();
    for(int v:e[u]) {
        if(!vis[v]) {
            vis[v] = 1;
            dis[v] = dis[u] + 1;
            q.push(v);
        }
        if(dis[v] == dis[u]+1) { // 更新路径数
            cnt[v] = (cnt[v] + cnt[u]) % mod;
        }
    }
}
```

E4

Problems List

#	Title	Total	AC Rate
4-1	小花的素数	1215	12.43%
4-2	小花与斐波那契数列	542	28.04%
4-3	小花的回文串	405	38.02%
4-4	小花的子串	480	31.46%

E4-1 小花的素数

题面

> 小花正在研究素数，素数是一个非常有趣的东西。
> 她想要借助计算机的力量，帮助她计算区间$[L, R]$之间所有的素数个数。
> **输入**
> 输入为两个整数L, R表示查询的区间。

> $(1 \le L \le R \le 2^{31}-1), R-L \le 10^6$
> **输出**
> 输出一个整数，表示给定区间中素数的个数。

题目大意

- 求区间 `[L, R]` 之间的素数

解题思路

- 首先关注题目的数据范围： `L` 和 `R` 的范围很大，但是 `R - L` 的范围小于 `1e6` 。所以不能使用朴素的素数筛法。
- 任何合数 `x` 都具有一个小于等于 `sqrt(x)` 的质因子。
- 先通过素数筛，筛选小于等于 `sqrt(R)` 的质因子，然后再通过素数筛筛出的素数，标记出所有在区间 `[L, R]` 之内的合数。

核心代码

```
void Gprime() { // 筛选小于sqrt(R)的素数
    for(int i=2;i<=50000;++i) {
        if(!vis[i])prime[++cnt]=i;
        for(int j=1;i*prime[j]<=50000;j++) {
            vis[i*prime[j]]=1;//标记合数
            if(i%prime[j]==0) break;
        }
    }
}

for(int i=1;i<=cnt;++i) {
    long long p = prime[i];
    long long start = (l+p-1)/p*p>2*p ? (l+p-1)/p*p : 2*p; //
倍增
    for(long long j=start;j<=r;j+=p) {
        vis[j-l+1]=1; // 标记合数
    }
}
```

扩展题目

- https://vjudge.net/problem/POJ-2689

E4-2 小花与斐波那契数列

题面

> 小花正在研究斐波那契数列。
> 斐波那契数列满足：$F_1 = 1, F_n = F_{n-1} + F_{n-2} (n \geq 2)$
> 她发现，当n变得很大时，F_n也会变得相当大。
> 她想知道，当n变得很大时，$F_n \ mod \ 10^9 + 7$的值。
> 输入
> 输入一个正整数n.
> $(35 < n < 2^{31} - 1)$
> **输出**
> 输出一个正整数，表示$F_n \ mod \ 10^9 + 7$的值。

题目大意

- 求出斐波那契堆数列的第i个数字。

解题思路

- 矩阵快速幂。
- 时间复杂度$O(nlogn)$
- 通过 `f[i] = f[i-1] + f[i-2]` 构造矩阵递推公式。

$$[f_{n+1} \quad f_n] = [f_n \quad f_{n-1}] \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = [f_n + f_{n-1} \quad f_n] \tag{1}$$

核心代码

```
struct Matrix {
    long long a[3][3];
    Matrix() { memset(a, 0, sizeof a); } // 构造函数，矩阵初始化全零
    Matrix operator*(const Matrix &b) const {
        Matrix res;
```

```
        for (int i = 1; i <= 2; ++i) {
            for (int j = 1; j <= 2; ++j) {
                for (int k = 1; k <= 2; ++k) {
                    res.a[i][j] = ((res.a[i][j] + a[i][k] * b.a[k][j] % mod) % mod + mod) % mod;
                }
            }
        }
        return res;
    }
} base;

Matrix qpow(Matrix x, long long y) { // 求 a 的 b 次方
    Matrix res = x;
    y -= 1;
    for (; y; y >>= 1, x = x * x) if (y & 1) res = res * x;
    return res;
}

base.a[1][1] = 1;
base.a[1][2] = 1;
base.a[2][1] = 1;
base.a[2][2] = 0;

Matrix ans = qpow(base, n);
cout<<ans.a[1][2]<<endl;
```

扩展题目

- http://poj.org/problem?id=3233
- http://poj.org/problem?id=3735

E4-3 小花的回文串

题面

> 小花的回文串
> 小花有一个长度为N的字符串，由英文字符a z组成。她想知道其中长度最大的回文串的长度是多少?
> 对于一个回文串，该字符串顺序或逆序得到的结果都是一致的。例如$abbcbba$、$baab$是回文串。
> **输入**

> 第一行输入一个整数N，表示输入一个长度为N的字符串。
> 第二行输入一个长度为N的字符串S，仅包含小写字母$a \sim z$.
> $(1 \leq N \leq 10^6)$
> **输出**
> 请输入一个整数，表示该字符串的最大回文串的长度。

题目大意

- 求出字符串中最长的回文串的长度。

解题思路

- 应用马拉车算法
- 马拉车算法可以在线性时间时间内求出给定字符串的最大回文串长度。
- 参考链接：
  - https://www.scaler.com/topics/data-structures/manachers-algorithm/
  - https://oi-wiki.org/string/manacher/

核心代码

```
int manacher() {
    int ans = 0;
    int c = -1, mx = -1;
    for(int i = 1; i <= n; ++i)
    {
        if (i > mx) r[i] = 1;
        else r[i] = min(mx - i + 1, r[2 * c - i]);
        while (s[i - r[i]] == s[i + r[i]]) ++r[i];
        if (i + r[i] - 1 > mx)
        {
            mx = i + r[i] - 1;
            c = i;
        }
        ans = max(ans, r[i] - 1);
    }
    return ans;
}

void change() {
    n = strlen(c) * 2 + 3;
    s[0] = '@';
```

```
22      s[1] = '$';
23      s[n] = '&';
24      for(int i = 2; i < n; ++i)
25      {
26          if (i & 1) s[i] = c[(i >> 1) - 1];
27          else s[i] = '#';
28      }
29  }
```

扩展题目

- https://vjudge.net/contest/273553

E4-4 小花的子串

题面

> 小花的子串
> 小花从小树那里得到了两个字符串S和T，S的长度为N，T的长度为M．她发现，T是S的子串，并且在S中存在多个为T的子串。
> 她想知道，字符串S中一共有多少个这样的子串T。以及它们在S中的位置。
> 其中，如果$[T_0, T_1, \ldots, T_{M-1}] = [S_i, S_{i+1}, \ldots, S_{i+M-1}]$，则$T$在$S$中的位置表示为$i$．
> 并且，字符串的下标从0开始．
> **输入**
> 第一行包含两个正整数 N，M，表示字符串的长度
> 第二行包含一个字符串S．
> 第三行包含一个字符串T．$(1 \leq M \leq N \leq 10^6)$，$S$和$T$中包含大写字母$A \sim Z$和小写字母$a \sim z$．
> **输出**
> 第一行输出一个整数T，表示S中存在多少子串T．
> 第二行按照升序输出T个整数，表示T在S中的位置，用空格隔开。

题目大意

- 求出子串T在字符串S中存在的所有位置。

解题思路

- 应用KMP算法
- 时间复杂度$O(n)$
- 参考链接：https://oi-wiki.org/string/kmp/

核心代码

```
vector<int> ans;
void get_next() { // next数组是从 S[0到i-1]前子串 的前缀后缀最大值
    int t1 = 0, t2;
    next1[0] = t2 = -1;
    while(t1 < len2)
        if(t2 == -1 || s2[t1] == s2[t2]) // 匹配
            next1[++t1] = ++t2;
        else t2 = next1[t2];// 失配
}
void KMP() { // KMP
    int t1 = 0,t2 = 0;
    while(t1 < len1) {
        if(t2 == -1 || s1[t1] == s2[t2]) // 匹配成功，继续
            t1++,t2++;
        else t2 = next1[t2]; // 失配
        if(t2 == len2) { // t2==len2时，匹配成功；t1-len2+1即为第一个字母的位置
            ans.push_back(t1 - len2); // 匹配成功，记录子串的未知
            t2 = next1[t2];
        }
    } // 匹配成功后，t2置为next[t2]
}
```

扩展题目

- https://acm.hdu.edu.cn/showproblem.php?pid=2087
- https://acm.hdu.edu.cn/showproblem.php?pid=3746
- https://acm.hdu.edu.cn/showproblem.php?pid=1358