PTA 函数 E

rogeryoungh

2021 年 06 月 20 日

目录

1 7-1 不能用循环是一件多么悲伤的事 2

2 7-2 乘法口诀表 2

3 7-3 素因子分解 2

4 7-4 谷歌的招聘 3

5 7-5 整数拆分 5

6 7-6 特立独行的幸福 5

7 7-7 素数对 7

8 7-8 学分绩点 8

9 7-9 函数的嵌套调用 9

10 7-10 算星期 9

11 7-11 任意进制下的可逆素数 9

12 7-12 计算 1！+2！+...+n! 10

13 7-13 二分查找（折半查找） 10

14 7-14 库函数开根号 11

15 7-15 编写函数输出一个十进制整数的十六进制形式 11

16 7-16 亲和数对 11

PTA 函数 Easy 部分，PDF。

1 7-1 不能用循环是一件多么悲伤的事

函数可以当循环用。

```c
ll n;

void f(ll a, ll b) {
    printf("%-2lld+%2lld = %-2lld ", a, b, a + b);
    if (a + b < n) {
        f(a, b + 1);
        if (b == 0) {
            f(a + 1, b);
        }
    } else {
        printf("\n");
    }
}

int main() {
    n = rr();
    f(0, 0);
    return 0;
}
```

2 7-2 乘法口诀表

```c
int main() {
    ll n = rr();
    for (ll i = 1; i <= n; i++) {
        for (ll j = 1; j <= i; j++) {
            if (j != 1)
                putchar(' ');
            printf("%lldx%lld=%2lld", j, i, i * j);
        }
        putchar('\n');
    }
    return 0;
}
```

3 7-3 素因子分解

```
int main() {
    ll n = rr();
    printf("%lld=", n);
    ll sn = sqrt(n * 1.0);
    int flag = 0;
    if (n == 1) {
        printf("1");
        return 0;
    }
    for (ll i = 2; i <= n; i++) {
        ll k = 0;
        while (n % i == 0) {
            k++, n /= i;
        }
        if (k) {
            if (flag)
                printf("*");
            flag = 1;
            if (k == 1)
                printf("%lld", i);
            else if (k > 1)
                printf("%lld^%lld", i, k);
        }
    }
    return 0;
}
```

4 7-4 谷歌的招聘

注意 10 位数字的乘法 long long 是不够的，需要更大的类型 __int128_t。

```
typedef __int128_t _i128;

_i128 qpow(_i128 a, _i128 b, _i128 p) {
    _i128 rst = 1 % p;
    for (; b > 0; b >>= 1) {
        if (b & 1)
            rst = a * rst % p;
        a = a * a % p;
    }
    return rst;
}
```

```
int miller_rabbin(ll n) {
    if (n < 3)
        return (n == 2);
    ll a = n - 1, b = 0;
    while (1 - (a & 1)) {
        a >>= 1, ++b;
    }
    ll prime[10] = {2, 7, 61};
    for (ll i = 0; i <= 2; i++) {
        ll x = prime[i];
        if (n == x)
            return 1;
        ll v = qpow(x, a, n);
        if (v == 1 || v == n - 1)
            continue;
        ll j;
        for (j = 0; j < b; j++) {
            v = v * v % n;
            if (v == n - 1)
                break;
        }
        if (j >= b)
            return 0;
    }
    return 1;
}

char ss[10086];
ll sum[10086];

int main() {
    ll l = rr(), k = rr();
    scanf("%s", ss + 1);
    ll ek = qpow(10, k, __LONG_LONG_MAX__);
    _fora (i, 1, l)
        sum[i] = sum[i - 1] * 10 + ss[i] - '0';
    int flag = 1;
    _fora (i, k, l) {
        ll t = sum[i] - sum[i - k] * ek;
        if (miller_rabbin(t)) {
            flag = 0;
            ss[i+1] = 0;
            printf("%s", &ss[i-k+1]);
            break;
        }
    }
    if (flag)
        printf("404\n");
    return 0;
}
```

5 7-5 整数拆分

暴力搜索复杂度过高，需要使用动态规划。

```
ll dp[105][105];

int main() {
    ll n, k;
    while (scanf("%lld,%lld", &n, &k) != EOF) {
        for (ll i = 1; i <= n; i++)
            dp[0][i] = dp[1][i] = 1;
        ll ans = 1;
        for (ll i = 2; i <= k; i++) {
            dp[i][1] = dp[i][0] = 0;
            for (ll j = 2; j <= n; j++) {
                if (j > i)
                    dp[i][j] = dp[i - 1][j - 1] + dp[i][j - i];
                else
                    dp[i][j] = dp[i - 1][j - 1];
            }
            ans += dp[i][n];
        }
        printf("%lld\n", ans);
    }
    return 0;
}
```

6 7-6 特立独行的幸福

将一次迭代看作一个数字到另一个数字的有向边，于是这变成了一个图论问题，DFS 即可。

```
const ll MN = 10086;

```

```
int dlx[MN], vis[MN], fat[MN], dp[MN];

void dfs(int n) {
    if (n == 1 || dlx[n] == 1)
        return;
    int fa = fat[n];
    dlx[n] = vis[fa] = 1;
    dfs(fa);
    if (dlx[fa] != 1) {
        dlx[n] = -1;
        dp[n] = dp[fa] + 1;
    }
}

int notp[10000001];
void init(int n) {
    for (ll i = 2; i <= n; i++) {
        int s = 0;
        for (int x = i; x; x /= 10) {
            s += (x % 10) * (x % 10);
        }
        fat[i] = s;
    }
    for (ll i = 2; i <= n; i++) {
        if (!notp[i]) {
            int tn = n / i;
            for (int j = i; j <= tn; j++)
                notp[i * j] = 1;
        }
    }
}

int main() {
    int a = rr(), b = rr();
    init(MN);
    for (ll i = a; i <= b; i++) {
        dfs(i);
    }
    int flag = 1;
    for (ll i = a; i <= b; i++) {
        if (dlx[i] < 0 && !vis[i]) {
            flag = 0;
            ll t = dp[i];
            if (!notp[i])
                t *= 2;
            printf("%lld %lld\n", i, t);
        }
    }
    if (flag)
        printf("SAD");
    return 0;
}
```

7 7-7 素数对

打表！打表！

```
int oeisA006512[] = {
    5,    7,     13,   19,   31,   43,   61,   73,   103,  109,  139,  151,
    181,  193,   199,  229,  241,  271,  283,  313,  349,  421,  433,  463,
    523,  571,   601,  619,  643,  661,  811,  823,  829,  859,  883,  1021,
    1033, 1051,  1063, 1093, 1153, 1231, 1279, 1291, 1303, 1321, 1429, 1453,
    1483, 1489,  1609, 1621, 1669, 1699, 1723, 1789, 1873, 1879, 1933, 1951,
    1999, 2029,  2083, 2089, 2113, 2131, 2143, 2239, 2269, 2311, 2341, 2383,
    2551, 2593,  2659, 2689, 2713, 2731, 2791, 2803, 2971, 3001, 3121, 3169,
    3253, 3259,  3301, 3331, 3361, 3373, 3391, 3463, 3469, 3529, 3541, 3559,
    3583, 3673,  3769, 3823, 3853, 3919, 3931, 4003, 4021, 4051, 4093, 4129,
    4159, 4219,  4231, 4243, 4261, 4273, 4339, 4423, 4483, 4519, 4549, 4639,
    4651, 4723,  4789, 4801, 4933, 4969, 5011, 5023, 5101, 5233, 5281, 5419,
    5443, 5479,  5503, 5521, 5641, 5653, 5659, 5743, 5851, 5869, 5881, 6091,
    6133, 6199,  6271, 6301, 6361, 6451, 6553, 6571, 6661, 6691, 6703, 6763,
    6781, 6793,  6829, 6871, 6949, 6961, 7129, 7213, 7309, 7333, 7351, 7459,
    7489, 7549,  7561, 7591, 7759, 7879, 7951, 8011, 8089, 8221, 8233, 8293,
    8389, 8431,  8539, 8599, 8629, 8821, 8839, 8863, 8971, 9001, 9013, 9043,
    9241, 9283,  9343, 9421, 9433, 9439, 9463, 9631, 9679, 9721, 9769, 9859,
    9931, 10009,
};

int main() {
    ll n = rr();
    if (n < 5)
        printf("empty\n");
    for (ll i = 0; i <= 10086; i++) {
        int t = oeisA006512[i];
        if (t <= n) {
            printf("%lld %lld\n", t - 2, t);
        } else {
            break;
        }
    }
    return 0;
}
```

8 7-8 学分绩点

```
double jidian(ll n) {
    if (n >= 90)
        return 4.0;
    else if (n >= 85)
        return 3.7;
    else if (n >= 82)
        return 3.3;
    else if (n >= 78)
        return 3.0;
    else if (n >= 75)
        return 2.7;
    else if (n >= 72)
        return 2.3;
    else if (n >= 68)
        return 2.0;
    else if (n >= 64)
        return 1.5;
    else if (n >= 60)
        return 1.0;
    return 0;
}

ll fen[10086];

int main() {
    ll n = rr();
    double s1 = 0;
    for (ll i = 1; i <= n; i++)
        s1 += fen[i] = rr();
    double s2 = 0;
    for (ll i = 1; i <= n; i++) {
        ll t = rr();
        s2 += fen[i] * jidian(t);
    }
    printf("%.2lf\n", s2 / s1);
    return 0;
}
```

9 7-9 函数的嵌套调用

略。

10 7-10 算星期

```
ll week(ll y, ll m, ll d) {
    if (m < 3) {
        m += 12;
        y--;
    }
    ll ans = d + 2 * m + 3 * (m + 1) / 5 + y + y / 4 - y / 100 + y / 400 + 1;
    return ans % 7;
}

char ww[][10] = {
    "日", "一", "二", "三", "四", "五", "六",
};

int main() {
    ll y = rr(), m = rr(), d = rr();
    printf("星期%s", ww[week(y, m, d)]);
    return 0;
}
```

11 7-11 任意进制下的可逆素数

```
int main() {
    while (1) {
        ll n = rr();
        if (n < 0)
            break;
        ll d = rr();
        ll t = 0, tn = n;
        while (tn) {
            t = t * d + tn % d;
            tn /= d;
        }
        if (miller_rabbin(t)) {
            printf("Yes\n");
        } else {
            printf("No\n");
        }
    }
    return 0;
}
```

12 7-12 计算 1！ +2！ +...+n!

```
int main() {
    ll n = rr();
    ll ans = n;
    for (ll i = n - 1; i >= 1; i--)
        ans = (ans + 1) * i;
    printf("sum=%lld", ans);
    return 0;
}
```

13 7-13 二分查找（折半查找）

```
ll nn[100086];

ll lower_bound(ll l, ll r, ll val) {
    while (l < r) {
        ll mid = (l + r) >> 1;
        if (nn[mid] > val)
            r = mid;
        else
            l = mid + 1;
    }
    return l;
}

int main() {
    ll ttt = rr();
    while (ttt--) {
        ll n = rr(), key = rr();
        ll p = -1;
        for (ll i = 1; i <= n; i++) {
            ll t = rr();
            if (p == -1 && t == key)
                p = i;
        }
        printf("%lld\n", p);
    }
    return 0;
}
```

14 7-14 库函数开根号

略。

15 7-15 编写函数输出一个十进制整数的十六进制形式

```
void pr(int n, int x) {
    char c = n % x;
    c += c >= 10 ? 'A' - 10 : '0';
    if (n >= x)
        pr(n / x, x);
    putchar(c);
}

int main() {
    ll a = rr(), b = rr();
    for (ll i = a; i <= b; i++) {
        pr(i, 16);
        if (i != b)
            putchar(' ');
    }
    return 0;
}
```

16 7-16 亲和数对

```
// OIES A259180 Amicable pairs.
ll nn[] = {
    220,   284,   1184,  1210,  2620,  2924,  5020,  5564,  6232,  6368,
    10744, 10856, 12285, 14595, 17296, 18416, 63020, 76084, 66928, 66992,
};

ll lower_bound(ll l, ll r, ll val) {
    while (l < r) {
        ll mid = (l + r) >> 1;
        if (nn[mid] > val)
            r = mid;
        else
            l = mid + 1;
    }
    return l;
}

int main() {
    ll a = rr(), b = rr();
    ll sa = lower_bound(0, 19, a), sb = lower_bound(0, 19, b);
    sa = sa / 2, sb = sb / 2;
    if (nn[sa * 2] < a)
        sa++;
    if (nn[sb * 2 + 1] > b)
        sb--;
    for (ll i = sa; i <= sb; i++) {
        printf("%lld,%lld\n", nn[i * 2], nn[i * 2 + 1]);
    }
    return 0;
}
```