Level 1

1.你好，天梯赛

按要求输出即可

```
# include <stdio.h>

int main () {
        printf("Hello, Tian Ti Sai!");
}
```

2.天梯赛考场

n 行，每一行有 m 个人，一共有 $n \times m$ 个人

```
# include <stdio.h>

int main () {
        int a, b; scanf("%d%d", &a, &b);
        printf("%d\n", a * b);
}
```

3.《算法竞赛·进阶指南》

任务化简

输入一个字符，如果字符是 I 说明输入的是 int ，否则是 $long\ long$

按要求使用一个 if 即可

```
#include <stdio.h>

int main () {
        char s; scanf("%c", &s);
        if ( s == 'I' ) printf("2147483647");
        else
 printf("9223372036854775807");
}
```

4.《AK》

一个 AK 有两个字符，说明如果输入的数是偶数那么就可以输出完整的 "AK字符串"
但是要注意一个空字符串是不具有 AK 的，所以要特判 0

```c
#include <stdio.h>

int main () {
        int n; scanf("%d", &n);
        if ( n % 2 != 0 || n == 0 ) printf("NO");
        else {
                printf("YES\n");
                n /= 2;
                while ( n -- ) printf("AK");
        }
}
```

5.天鸟火炮

问题抽象出来

问：是否满足 $\frac{b}{a} \geq \frac{c}{100} \Rightarrow b \times 100 \geq a \times c$

```c
#include <stdio.h>

int main () {
        int cass; scanf("%d", &cass); while ( cass --
- ) {
                int a, b, c; scanf("%d%d%d", &a, &b,
&c);
                if ( b * 100 < a * c ) puts("No");
                else                   puts("Yes");
        }
}
```

6.《从你的全世界路过》

如果 y 是 x 的幂的话，它的因子里面必然只有 x
那么可以使用一个 $while$ 看看能不能将 y 化成 1

但是要特判一下如果 x 是 1 的话这么做会超时
且 1 的幂必然是 1

```
#include <stdio.h>

int main () {
        int x, y; scanf("%d%d", &x, &y);
        if ( x == 1 && y != 1 ) { puts("NO"); return
0; }

        while ( y % x == 0 ) y /= x;
        if ( y == 1 ) puts("YES");
        else          puts("NO");
}
```

7.《双城之战》

问题就是判断第几个字符串与第一行输入的字符串相等，但是要注意名字之间有空格的情况

```
# include <stdio.h>
# include <string.h>

int equil ( char *a, char *b ) {
        int lena = strlen(a), lenb = strlen(b);
        if ( lena != lenb ) return 0;
        for ( int i = 0; i < lena; i ++ ) {
                if ( a[i] != b[i] ) return 0;
        }
```

```
        return 1;
}

char s[1100];
int n;

int main () {
        gets(s);
        scanf("%d", &n);

        getchar();
        for ( int i = 1; i <= n; i ++ ) {
                char cur[1100]; gets(cur);
                if ( equil(cur, s) ) { printf("%d",
i); return 0; }
        }
}
```

8.“这个世界不需要守望先锋”

我们先算出末日铁拳的总血量 $n * 25 + 250$
看一下这个数除 20 能不能除完，除不完要再多打一次

因为 int 类型加和乘可能会爆，所以要用 $long\ long$

```
# include <stdio.h>
# include <string.h>

int main () {
        long long n; scanf("%lld", &n);
        long long all_hp = n * 25 + 250;
        printf("%lld\n", all_hp / 20 + (all_hp % 20
!= 0) );
}
```

Level 2

1.《天堂旅行团》

输出第二大的数

数据量是 10^7 那么我们用 $sort$ 函数一定超时
所以我们第一次循环记录一下最大值，第二次循环记录最大值的个数和不等于最大值的次大值的数值
如果最大值个数不为 1 那么可以输出这个最大值，否则输出次大值

如果 10^7 的大小直接开到 $main$ 函数里会发生爆栈，显示栈错误

```c
# include <stdio.h>

int a[10000007];

int max ( int a, int b ) { return a > b ? a : b; }

int main () {
        int n; scanf("%d", &n);

        int cnt_max = 0, val_max = 0, val_secmax =
0;
        for ( int i = 0; i < n; i ++ ) {
                scanf("%d", &a[i]);
                val_max = max ( val_max, a[i] );
        }
        for ( int i = 0; i < n; i ++ ) {
                if ( a[i] == val_max ) cnt_max ++;
                if ( a[i] < val_max )  val_secmax =
max ( a[i], val_secmax );
        }
        if ( cnt_max > 1 ) printf("%d", val_max);
        else printf("%d", val_secmax);
}
```

2.《肖申克的救赎》

一个血量，从数组首遍历到尾，如果中途血量不够减了就跳出循环

```
# include <stdio.h>

const int N = 110;
int a[N];

int main () {
        int n, h; scanf("%d", &n);
        for ( int i = 0; i < n; i ++ ) scanf("%d",
&a[i]);
        scanf("%d", &h);

        int res = 0;
        for ( int i = 0; i < n; i ++ ) {
                if ( h - a[i] < 0 ) break;
                h -= a[i], res += a[i];
        }
        printf("%d", res);
}
```

3.请再一次做我的棋子

因为正方形可以是斜着的，所以我们用距离判断就很方便

一共有 6 条边，四个正方形边两个正方形对角线

我们存一下边然后升序排个序，如果前四个相等，后两个相等，且没有 0 边长，就说明是个正方形，输出 `好耶!`
否则直接输出 `emo!`

> 实数判断相等要注意消除精度（ZZULIOJ上之前有过

```
# include <stdio.h>
# include <math.h>

int x[4], y[4];
double dis[20];
int id_dis = 0;

double get_Dis ( int x1, int y1, int x2, int y2 ) {
// (x1, y1)与(x2, y2)的距离
        long long dirx = x1 - x2; dirx *= dirx;
        long long diry = y1 - y2; diry *= diry;
        return sqrt(dirx + diry);
}
int equal ( double x, double y ) { // 相等返回1，否则返回0
        if ( fabs(x - y) < 1e-9 ) return 1;
        return 0;
}
```

```
int main () {
        for ( int i = 0; i < 4; i ++ ) scanf("%d",
&x[i]);
        for ( int i = 0; i < 4; i ++ ) scanf("%d",
&y[i]);
        for ( int i = 0; i < 4; i ++ ) {
                for ( int j = i + 1; j < 4; j ++ ) {
                        dis[id_dis ++] = get_Dis
(x[i], y[i], x[j], y[j]);
                }
        }
        for ( int i = 0; i < id_dis; i ++ ) {
                for ( int j = i + 1; j < id_dis; j
++ ) {
                        if ( dis[i] > dis[j] ) {
                                double tmp = dis[i];
                                dis[i] = dis[j];
                                dis[j] = tmp;
                        }
                }
        }
        if ( !equal(dis[0], 0) && equal(dis[0],
dis[1]) && equal(dis[1], dis[2]) && equal(dis[2],
dis[3]) && equal(dis[4], dis[5]) ) {
                if ( equal(dis[0] * sqrt(2), dis[4])
) puts("好耶!");
                else puts("emo! ");
```

```
        } else puts("emo! ");
}
```

4.《夏摩山谷》

$36\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{a=1}^{i}\sum_{b=1}^{j}a\times b \qquad O(n^4)\to 3$分

看后面

$1\times(1+2+\cdots+j)$

$2\times(1+2+\cdots+j)$

$\vdots$

$i\times(1+2+\cdots+j)$

$=(1+2+\cdots+i)(1+2+\cdots+j)=\frac{(1+i)i}{2}\frac{(1+j)j}{2}$

那么原式$=9\sum_{i=1}^{n}\sum_{j=1}^{n}(i+i^2)(j+j^2) \qquad O(n^2)\to 15$分

同上一步理，乘法分配律，变成 $9\sum_{i=1}^{n}(i+i^2)\sum_{j=1}^{n}(j+j^2)$

对于 $\sum_{i=1}^{n}(i+i^2)$ 前一部分使用 等差数列求和 后一部分使用等差数列求平方和

$$
\begin{aligned}
&= \frac{(1+n)n}{2} + \frac{n(n+1)(2n+1)}{6} \\
&= \frac{n(n+1)3 + n(n+1)(2n+1)}{6} \\
&= \frac{n(n+1)(2n+4)}{6} \\
&= \frac{n(n+1)(n+2)}{3}
\end{aligned}
$$

j 那一部分一样，那么原式就变成

$$
\begin{aligned}
&9\frac{n(n+1)(n+2)}{3}\frac{n(n+1)(n+2)}{3} \\
=&n^2(n+1)^2(n+2)^2
\end{aligned}
$$

$O(1) \to 25$分

```
# include <stdio.h>

const int mod = 1e9 + 7;

int main () {
        int cass; scanf("%d", &cass); while ( cass -
- ) {
                long long n; scanf("%lld", &n);
                printf("%lld\n", n * n % mod * (n +
1) % mod * (n + 1) % mod * (n + 2) % mod * (n + 2) %
mod);
        }
}
```

Level 3

1.进制转化

按题目要求，要有一个好的模拟策略

如果最后一个字符是字母，就截取前面的进行转化，否则截取后面的
字符串函数如果不会用的话可以手动进行截取和判断长度

注意会有 $0B$ 这种情况，可以特判，也可以通过从第三个字符开始截取来消掉这种情况

```
# include <stdio.h>
# include <string.h>
# define ll long long

ll change28 ( ll base, char *s ) { // base进制转化为10进制
        ll res = 0, len = strlen(s);
        for ( int i = 0; i < len; i ++ ) res = res * base + (s[i] - '0');
        return res;
}
ll change16 ( char *s ) { // 16进制转化为10进制
        ll res = 0, len = strlen(s);
        for ( int i = 0; i < len; i ++ ) {
                switch ( s[i] ) {
                        case 'A': res = res * 16 + 10; break;
                        case 'B': res = res * 16 + 11; break;
                        case 'C': res = res * 16 + 12; break;
                        case 'D': res = res * 16 + 13; break;
```

```
                        case 'E': res = res * 16 +
14; break;
                        case 'F': res = res * 16 +
15; break;
                        case 'a': res = res * 16 +
10; break;
                        case 'b': res = res * 16 +
11; break;
                        case 'c': res = res * 16 +
12; break;
                        case 'd': res = res * 16 +
13; break;
                        case 'e': res = res * 16 +
14; break;
                        case 'f': res = res * 16 +
15; break;
                        default : res = res * 16 +
(s[i] - '0'); break;
                }
        }
        return res;
}

void Solve() {
        char s[100]; getchar(); scanf("%s", s);
        if ( s[0] == '0' ) {
                char ss[100];
```

```
                for ( int i = 2; i < strlen(s); i ++ ) ss[i - 2] = s[i]; // 从第三个字符开始截取
                ss[strlen(s) - 2] = '\0';// 设置结束符
                switch ( s[1] ) { // 第二个字符
                        case 'b':
                                printf("%lld\n", change28(2, ss));
                        break;
                        case 'o':
                                printf("%lld\n", change28(8, ss));
                        break;
                        case 'x':
                                printf("%lld\n", change16(ss));
                        break;
                        default: puts("0");
                }
        } else {
                char ss[100];
                for ( int i = 0; i < strlen(s) - 1; i ++ ) ss[i] = s[i]; // 不截取最后一个
                ss[strlen(s) - 1] = '\0';// 设置结束符
                switch ( s[strlen(s) - 1] ) { // 最后一个字符
```

```
                        case 'B':
                                printf("%lld\n",
change28(2, ss));
                        break;
                        case 'O':
                                printf("%lld\n",
change28(8, ss));
                        break;
                        case 'H':
                                printf("%lld\n",
change16(ss));
                        break;
                }
        }
}

int main () {
        int cass; scanf("%d", &cass); while ( cass -
- ) {
                Solve();
        }
}
```

2.小Z爱读书

也是个模拟题，需要好的策略
我们可以把这个长方体打印以正面中间的棱分成两部分进行打印
注意回车和空格的分布情况

要特判：边长为 1 的情况下，这一个维度没有面积
如果有一个 1，那么是打印长方形，两个 1 打印线，三个 1 打印点

```
#include <stdio.h>

int min ( int a, int b ) { return a < b? a : b; }

char t;
int a, b, c;
int h, w, l, sum;

void work() {
        if (sum == 0 || sum == h) {
                printf("%c", t);
        } else {
                for (int i = 0; i < min(min(b - 2, c - 2), min(sum, h - sum - 1)); i++)
                        printf(" ");
                printf("%c", t);
```

```
        }
        sum++;
}
void sol() {
        int x = b - 1, sk = 0;
        while (x) {
                for (int i = 0; i < x; i++) printf("
");
                if (x == b - 1) for (int i = 0; i <
a; i++) printf("%c", t);
                else {
                        printf("%c", t);
                        for (int i = 0; i < a - 2;
i++) printf(" ");

                        if (a > 1) printf("%c", t);
                        if (c > 1 && b > 1) work();
                }
                printf("\n");
                x--;
        }
        for (int i = 0; i < a; i++) printf("%c", t);
        if (c > 1 && b > 1) work();
        printf("\n");

        if (c == 1) return;
        x = c - 2;
```

```
        while (x--) {
                printf("%c", t);
                for (int i = 0; i < a - 2; i++)
printf(" ");

                if (a > 1) printf("%c", t);
                if (c > 1 && b > 1) work();
                printf("\n");
        }

        for (int i = 0; i < a; i++) printf("%c", t);
        printf("\n");
}
int main()
{
        while (~scanf("%c", &t))
        {
                scanf("%d%d%d", &a, &b, &c);
                getchar();
                h = c + b - 3, w = c - 2, l = b - 2,
sum = 0;
                sol();
        }
}
```

3.震惊长安第一拳

这个题采用二分查找正确答案

可以发现，对于任意套餐，都需要一份自瞄一份透视
所以我们二分一下套餐数量，检查合不合法

令 A, B, C 分别为如果一份套餐只有“自瞄和透视”，剩余自瞄个数、透视个数、加伤害个数
第一个套餐要加一个“加伤害”，第二个要加一个“自瞄“，第三个加一个”透视“

因为不能有连续的套餐，所以任意的套餐数量为 $\left\lceil \frac{x}{2} \right\rceil$

我们二分出来的 x（套餐数量），检查方式就是尽可能充分利用我们的剩余脚本个数多拿
看看在这一次情况下我们可以拿的套餐数量是否多于 x ，如果多了就继续往上二分，否则往下二分

(下界是0，上界是$min($自瞄,透视$)$)

```
#include <stdio.h>
```

```
int a, b, c;

int min ( int a, int b ) {
        return a < b ? a : b;
}

int check ( int x ) {
        int A = a - x, B = b - x, C = c;
        int res = 0;
        int mx = (x + 1) / 2;

        res += min ( mx, A );
        res += min ( mx, B );
        res += min ( mx, C );
        if ( res >= x ) return 1;
        return 0;
}


int main() {
        int cass;
        for ( scanf("%d", &cass); cass; cass -- ) {
                scanf("%d %d %d", &a, &b, &c);
                int l = 0, r = min (a, b);
                while (l <= r) {
                        int mid = (l + r) >> 1;
                        if (check(mid)) l = mid + 1;
```

```
                        else r = mid - 1;

                }
                printf("%d\n", r);
        }
        return 0;
}
```