2861.最大合金数

题目大意

你有 k 台机器，制作合金总共需要用到 n 种材料，第 i 台机器需要用到的第 j 种金属个数为 $composition[i][j]$。初始时，第 j 种金属已有 $stock[j]$ 的库存，每新购入一块第 j 种金属，需要花 $cost[j]$ 块钱。现在你有 $budget$ 元，请挑选一台机器（只能用一台），最大化能制造的合金数量。

题目分析

如果选用第 i 台机器，想制造 $amount$数量的合金，那么固定会用到 $composition[i][j] * amount$ 块第 j 种金属。如果这个数目大于 $stock[j]$，那么就需要额外购入 $composition[i][j] * amount - stock[j]$ 块金属。否则不需要采购，记需要采购第 j 种金属的个数为 buy_j，总共需要花费的金额为 $money$，有如下定量关系：

$$buy[j] = \begin{cases} 0 & stock[j] \geq need[j] \\ need[j] - stock[j] & stock[j] < need[j] \end{cases}$$

$$money_i = \sum_{j=1}^{n} buy[j] * cost[j]$$

观察到，$money_i$ 关于 $amount$ 单调增，因此我们可以对 $amount$ 进行二分，由题目数据，至多制造 $1e8$ 数目的合金。每次对于给定的 $amount$，我们可以在 $O(n)$ 时间内判断是否能控制在 $budget$ 以内，总体时间复杂度为 $O(kn)$。以下为完整代码：

```
int ret = 0;
inline bool ok(long long amount, int n, int machine, int budget,
        vector<vector<int>>& composition, vector<int>& stock, vector<int>& cost) {
    if (amount == 0) return true;
    long long money = 0;
    for (int i = 0;i < n;i++) {
        if (amount * composition[machine][i] > stock[i])
            money += (amount * (long long)composition[machine][i] - (long long)stock[i])
        if (money > budget) return false;
    }
    return true;
}
```

```
int maxNumberOfAlloys(int n, int k, int budget,
        vector<vector<int>>& composition, vector<int>& stock, vector<int>& cost) {
    for (int machine = 0;machine < k;machine++) {
        int l = 0, r = (int)2e8 + 1;
        while (l < r) {
            int mid = l + r + 1 >> 1;
            if (ok(mid, n, machine, budget, composition, stock, cost)) l = mid;
            else r = mid - 1;
        }
        ret = max(ret, l);
    }
    return ret;
}
```