make them equal

chenjiuri_cf_dp_makethemequal_1600_bfs_dponbag.

- 给定一个数组$a_1....a_n$;全为1的数组。
- 在数组中，任意选定数组中的一个元素。然后选定一个数字x，令$x = a_i/x(\text{向下取整}) + a_i$;
- 当进行规定次数操作次数之后。如果$a_i == b_i$,那么将收获c_i金币。

题解

- 只要计算得到每一个b_i的最小花费，这样就可以转化为简单的背包问题。
- 计算最小花费的方法；
  - n^2建图，然后利用图论经典的单源最短路算法。求出1到其它位置的的最短路。
- 直接暴力枚举即可。

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const int maxn = 1e3 + 10;
const int mmaxn = 1e6 + 10;

int cost[maxn];
int b[maxn];
int c[maxn];
int f[mmaxn];

void solve()
{
    int n, k;
    cin >> n >> k;
    f[0] = 0;

    for (int i = 1; i <= n; i++)
        cin >> b[i];
    for (int i = 1; i <= n; i++)
        cin >> c[i];
    for (int i = 0; i <= min(12 * n, k); i++)
        f[i] = 0;
    for (int i = 1; i <= n; i++)
        for (int j = min(k, 12 * n); j >= cost[b[i]]; j--)
            f[j] = max(f[j], f[j - cost[b[i]]] + c[i]);
    printf("%d\n", f[min(k, 12 * n)]);
}
int main()
{
    ios::sync_with_stdio(false);
```

```
    cin.tie(nullptr), cout.tie(nullptr);
    int t;
    cin >> t;
    for (int i = 0; i < maxn; i++)
        cost[i] = -1;
    queue<int> que;
    cost[1] = 0;
    que.emplace(1);
    while (!que.empty())
    {
        int tt = que.front();
        que.pop();
        for (int i = 1, flag = (tt + 2) / 2; i <= flag; i++) //这里利用一个整除分块进行优化。
            if (tt + tt / i <= 1000 && cost[tt + tt / i] == -1)
            {
                cost[tt + tt / i] = cost[tt] + 1;
                que.emplace(tt + tt / i);
            }
    }
    while (t--)
        solve();
}
```

生长思考

- 由于是一个整除分块的问题可以考虑，用整除分块相关的算法来优化。
- 之前的补题网站

优化如下：

```
 queue<int> que;
    cost[1] = 0;
    que.emplace(1);
    while (!que.empty())
    {
        int tt = que.front();
        que.pop();
        // for (int i = 1, flag = (tt + 2) / 2; i <= flag; i++) //这里利用一个整除分块进行优化。
        //     if (tt + tt / i <= 1000 && cost[tt + tt / i] == -1)
        //     {
        //         cost[tt + tt / i] = cost[tt] + 1;
        //         que.emplace(tt + tt / i);
        //     }
        for (int r = 1, l = 1; l <= tt; l = r + 1)
        {
            r = tt / (tt / l);
            if (tt + tt / l <= 1000 && cost[tt + tt / l] == -1)
```

```
            {
                cost[tt + tt / l] = cost[tt] + 1;
                que.emplace(tt + tt / l);
            }
        }
    }


    cost[1] = 0;  //预处理这一个小问题上都是差不多的。
    for (int i = 1; i <= 1000; i++)
        for (int j = i; j >= 1 && (i + i / j <= 1000); j--)
            cost[i + i / j] = min(cost[i + i / j], cost[i] + 1);
```