Maximum number of segments of lengths a, b and c

> 给定长度为N的线段，用a，b，c三种方式对其分割，使得分段的数量最多

dp[i]表示长度为i的线段被切割所能获得最多的分段数量，初始化dp[0] = 0, dp[1~N] = -1

初始化当前长度i为0，持续遍历到N，对于每一个dp[i]有：

1. 若dp[i] = -1，则说明当前长度无法分割，直接 i++
2. 如果dp[i] ！ = -1，则说明当前长度可分割，且分割数量为dp[i]，那么
    1. 若i + a ≤ N，则长度为i + a的线段的分割方式有两种，分为长度为i和长度为a，此时的分割总数量为dp[i] + 1，不使用长度a进行分割，则此时的分割总数量为dp[i + a]，取两者之中较大的值，即dp[i + a] = max(dp[i + a], dp[i] + 1)
    2. 若i + b ≤ N，则长度为i + b的线段的分割方式有两种，分为长度为i和长度为b，此时的分割总数量为dp[i] + 1，不使用长度b进行分割，则此时的分割总数量为dp[i + b]，取两者之中较大的值，即dp[i + b] = max(dp[i + b], dp[i] + 1)
    3. 若i + c ≤ N，则长度为i + c的线段的分割方式有两种，分为长度为i和长度为c， 此时的分割总数量为dp[i] + 1，不使用长度c进行分割，则此时的分割总数量为dp[i + c]，取两者之中较大的值，即dp[i + c] = max(dp[i + c], dp[i] + 1)
3. 最终返回dp[N]

```
#include<bits/stdc++.h>

using namespace std;

typedef vector<int> vi;

int maxSegments(int &N, int &a, int &b, int &c) {
    vi dp(N + 1, -1);
    dp[0] = 0;
    for (int i = 0; i <= N; ++i) {
        if (dp[i] != -1) {
            if (i + a <= N)
                dp[i + a] = max(dp[i] + 1, dp[i + a]);
            if (i + b <= N)
                dp[i + b] = max(dp[i] + 1, dp[i + b]);
            if (i + c <= N)
                dp[i + c] = max(dp[i] + 1, dp[i + c]);
        }
    }
    return dp[N];
}

int main() {
    int T;
    scanf("%d", &T);
    while (T--) {
        int N, a, b, c;
        scanf("%d %d %d %d", &N, &a, &b, &c);
        printf("%d\n", maxSegments(N, a, b, c));
    }
    return 0;
}
```