Maximum subsequence sum such that no three are consecutive

> 给定一个整数序列，找出和最大的子集序列——子集中不能出现原序列中三个位置连续的元素

dp[i]表示 arr[0~i]的非连续位置子集最大和，对于dp[i]有：

1. dp[i] 不包括 arr[i]，则dp[i] = dp[i - 1]
2. dp[i] 不包括 arr[i- 1]，则dp[i] = dp[i - 2] + arr[i]
3. dp[i] 不包括 arr[i - 2]，则dp[i] = dp[i - 3] + arr[i - 1] + arr[i]

选择其中最大的作为当前位置i的dp[i]值

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

using namespace std;

int maxSumSeq(vector<int> &arr, int &N) {
    vector<int> dp(N);
    dp[0] = arr[0], dp[1] = arr[0] + arr[1];
    dp[2] = max(dp[1], max(arr[0] + arr[2], arr[1] + arr[2]));
    for (int i = 3; i < N; ++i) {
        dp[i] = max(max(dp[i - 1], dp[i - 2] + arr[i]), dp[i - 3] + arr[i - 1] + arr[i]);
    }
    return dp[N - 1];
}

int main() {
    int T;
    scanf("%d", &T);
    while (T--) {
        int N, num;
        vector<int> arr;
        scanf("%d", &N);
        for (int i = 0; i < N; ++i) {
            scanf("%d", &num);
            arr.push_back(num);
        }
        printf("%d\n", maxSumSeq(arr, N));
    }
}
```