分析:

考虑分块，每个位置维护一个跳出该块需要的步数cnt[],和跳出该块后到达下一块的哪个位置to[]。

关于修改操作：直接修改所在块的左端点到修改的位置。 然后需要逆序修改。因为后面的位置的值会影响到前面位置的值。

关于询问操作：直接暴力计算即可。

思考：

第一道分块题，初步了解一下分块的用法（把点与点的联系可以转换成块与块的联系…

Code：

#include<iostream>

#include<cstdio>

#include<cstring>

#include<string>

#include<algorithm>

#include<cmath>

#include<queue>

#include<map>

#include<vector>

#include<set>

#include<bitset>

#include<stack>

#include<cctype>

#define sc(x) scanf("%d",&x);

#define PR(x) cout<<#x<<": "<<x<<endl;

using namespace std;

typedef long long ll;

typedef pair<int,int> p;

const double eps = 1e-4;

const int mod = 1e9+7;

const ll INF = 0x3f3f3f3f3f3f3f3f;

const int inf = 0x3f3f3f3f;

const int maxn = 5e5+7;

const int maxm = 1e6+7;

const double pi = acos(-1.0);

int n,q;

//跳出该块需要的部数 跳向下一个块的哪个点

int a[maxn],cnt[maxn],to[maxn];

//点属于第几个块/块长/块数 块左端 块右端

int belong[maxn],block,num,l[maxn],r[maxn];

//下标从1开始

void Build(){

block=(int)sqrt(n+0.5);

num=n/block;

if(n%block)

num++;

for(int i=1; i<=num; i++){

l[i]=(i-1)\*block+1;

r[i]=i\*block;

}

r[num]=n;//最右端端点更新

for(int i=1; i<=n; i++)

belong[i]=((i-1)/block)+1;

for(int i=num; i>0; i--){

for(int j=r[i]; j>=l[i]; j--){

if(j+a[j]>r[i]){

cnt[j]=1; to[j]=min(n+1,j+a[j]);

}

else{

cnt[j]=cnt[j+a[j]]+1;

to[j]=min(n+1,to[j+a[j]]);

}

}

}

}

void modify(int pos,int val){

a[pos]=val;

for(int i=pos; i>=l[belong[pos]]; i--){

if(i+a[i]>r[belong[pos]]){

cnt[i]=1;

to[i]=min(n+1,i+a[i]);

}

else{

cnt[i]=cnt[i+a[i]]+1;

to[i]=min(n+1,to[i+a[i]]);

}

}

}

int query(int pos){

int ans=0;

for(int i=pos; i<=n; i=to[i])

ans+=cnt[i];

return ans;

}

int main(){

while(scanf("%d",&n)!=EOF){

for(int i=1; i<=n; i++)

scanf("%d",&a[i]);

Build();

scanf("%d",&q);

int op,pos,v;

for(int i=0; i<q; i++){

scanf("%d",&op);

if(op==1){

scanf("%d",&pos); pos++;

printf("%d\n",query(pos));

}

else{

scanf("%d%d",&pos,&v); pos++;

modify(pos,v);

}

}

}

return 0;

}