

太戈编程
etiger.vip

信奥算法

pure

请同学写出题目大意
已知什么求什么

只包含2和3质因数的数叫做23杂种数。请你判断正整数 n 是不是23杂种数？

```
4 freopen("pure.in","r",stdin);
5 freopen("pure.out","w",stdout);
6 long long n;
7 cin>>n;
8 bool two=0,three=0;
9 while(n%2==0){
10     two=1;
11     n/=2;
12 }
13
14
15
16 }
17 cout<<"Yes"<<endl;
18 else cout<<"No"<<endl;
```

981

线段覆盖

输入

5

134 203

186 288

274 380

46 138

373 451

134

203

186

288

274

380

46

138

输出

357

373

451

$$(138-46)+(451-186)=357$$

请同学简述题意
突出核心要点

一维坐标轴上有 N 条线段
选择其中 $N-1$ 条
求最多能覆盖的长度

请同学分析数据范围

【数据规模与约定】

1号数据： $N=3$


2号数据： $N=5$

3号数据： $N=10$

对于所有数据： $N \leq 100, 0 \leq A_i \leq B_i \leq 1000$

细节处理

坐标点对应右侧短边

坐标 x 坐标 $x+1$

 x 号边

算法
步骤

纯暴力枚举+纯模拟

枚举删除的线段编号 $i=1,2,\dots,n$

模拟剩余线段覆盖坐标点

数据
结构

存在性数组

$ok[x]$ 表示 x 号边是否被覆盖

```
3  const int N=109;  
4  const int R=1009;  
5  int a[N],b[N];  
6   ok[];
```

```
10 int n;
11 cin>>n;
12 int ans=0;
13 for(int i=1;i<=n;++i) cin>>a[i]>>b[i];
14 for(int i=1;i<=n;++i){
15     [REDACTED]
16     for(int j=1;j<=n;++j){
17         if(j==i) [REDACTED]
18         for([REDACTED])
19             ok[x]=1;
20     }
21     int len=0;
22     for(int x=0;x<R;++x)
23         [REDACTED]
24     ans=max(ans,len);
25 }
```

复杂度分析

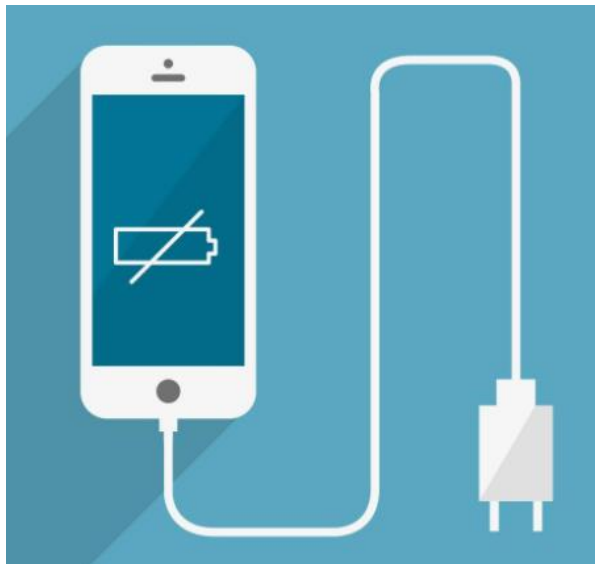
$$O(n*n*R)$$

讨论：算法优化

差分算法
对两端打标记

1909

手机读小说



请同学写出题目大意 已知什么求什么

m 格为满电，每格对应一分钟， n 分钟可用，第 i 分钟若读书可读 $d[i]$ 页，也可充电。一旦充电必须连续充电直到充满，求最多读几页

对于10%数据, $n=3$

对于10%数据, $m=1$

对于10%数据, 所有 $d[i]=1$

对于40%数据, $n \leq 500$, $m \leq 100$

对于100%数据, $n \leq 10000$, $m \leq 500$

$n=3$

共3分钟，3分钟后必须满电

最后1分钟必须用于充电

前2分钟恰好用1分钟读书

答案 = $\max(d[1], d[2])$

$m=1$

简化版问题

每次读书1分钟后必须马上充电至少1分钟

n 分钟里选择读书的时间不可以出现连续的2分钟

$f[i]$ 表示第 i 分钟用于读书时前 i 分钟最多读几页

$$f[i] = d[i] + \max(f[i-2], f[i-3])$$

$$\text{答案} = \max(f[n-1], f[n-2])$$

小心下越界

部分分

多10分

所有
 $d[i]=1$

简化版问题

求最大读几页 = 求最多读几分钟

$n=12, m=4$

i=1	i=2	i=3	i=4	i=5	i=6	i=7	i=8	i=9	i=10	i=11	i=12
读书	读书	读书	读书	充电	充电	充电	充电	读书	充电	读书	充电

i=1	i=2	i=3	i=4	i=5	i=6	i=7	i=8	i=9	i=10	i=11	i=12
读书	充电	读书	充电	读书	充电	读书	充电	读书	充电	读书	充电

答案 = $n/2$

n 是奇数时 $n/2$ 取整即可

形成满分算法

识别模型要素

序列切割问题
也叫序列分段问题

序列分割点是最关键点

发现

必须连续读书+连续充电

$$n=12, m=4$$

i=1	i=2	i=3	i=4	i=5	i=6	i=7	i=8	i=9	i=10	i=11	i=12
读书	读书	读书	读书	充电	充电	充电	充电	读书	充电	读书	充电

i=1	i=2	i=3	i=4	i=5	i=6	i=7	i=8	i=9	i=10	i=11	i=12
读书	充电	读书	充电	读书	充电	读书	充电	读书	充电	读书	充电

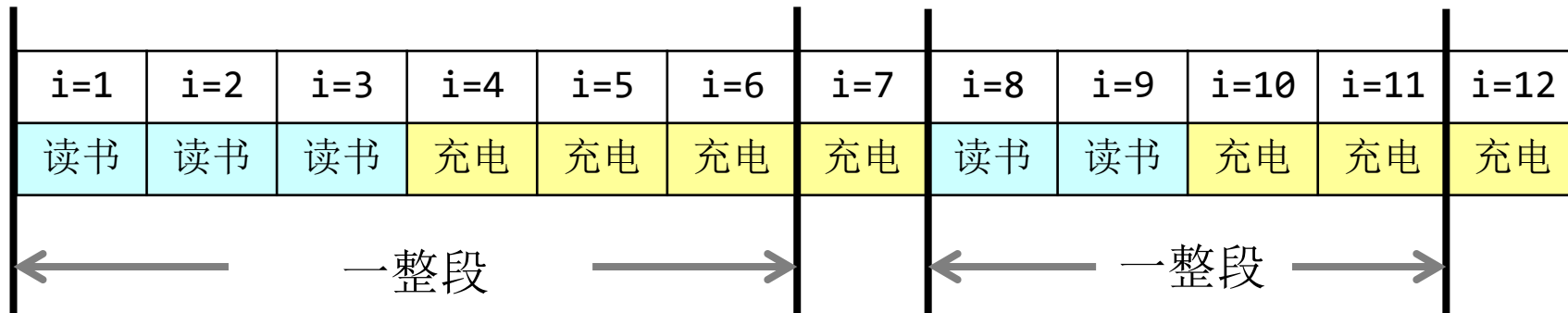
i=1	i=2	i=3	i=4	i=5	i=6	i=7	i=8	i=9	i=10	i=11	i=12
读书	读书	读书	充电	充电	充电	充电	读书	读书	充电	充电	充电

发现

必须连续读书+连续充电

x个蓝色+x个黄色

读书x分钟+充电x分钟 作为一段分割出的整体



序列切割	100分																									
发现	必须连续读书+连续充电																									
状态	$h[t]$ 第 t 分钟用于充电后恰为满电时最多读了几页																									
转移决策分2层	第 $t-1$ 分钟后是否已经满电	是非型决策																								
	前一次连续读书共几分钟	选择型决策																								
方程	$h[t] = \max(\quad h[t - 1],$ $\max_{1 \leq x \leq t/2} \{h[t - 2x] + d[t - 2x + 1] + d[t - 2x] + \cdots + d[t - x]\})$																									
<div><table><tr><td>i=1</td><td>i=2</td><td>i=3</td><td>i=4</td><td>i=5</td><td>i=6</td><td>i=7</td><td>i=8</td><td>i=9</td><td>i=10</td><td>i=11</td><td>i=12</td></tr><tr><td>读书</td><td>读书</td><td>读书</td><td>充电</td><td>充电</td><td>充电</td><td>充电</td><td>读书</td><td>读书</td><td>充电</td><td>充电</td><td>充电</td></tr></table><div><div>←</div><div>一整段</div><div>→</div><div>←</div><div>一整段</div><div>→</div></div></div>			i=1	i=2	i=3	i=4	i=5	i=6	i=7	i=8	i=9	i=10	i=11	i=12	读书	读书	读书	充电	充电	充电	充电	读书	读书	充电	充电	充电
i=1	i=2	i=3	i=4	i=5	i=6	i=7	i=8	i=9	i=10	i=11	i=12															
读书	读书	读书	充电	充电	充电	充电	读书	读书	充电	充电	充电															

序列切割	100分	
发现	必须连续读书+连续充电	
状态	$h[t]$ 第 t 分钟用于充电后恰为满电时最多读了几页	
转移决策分2层	第 $t-1$ 分钟后是否已经满电	是非型决策
	前一次连续读书共几分钟	选择型决策
方程	$h[t] = \max(\quad h[t - 1], \quad \max_{1 \leq x \leq t/2} \{h[t - 2x] + s[t - x] - s[t - 2x]\})$	

i=1	i=2	i=3	i=4	i=5	i=6	i=7	i=8	i=9	i=10	i=11	i=12
读书	读书	读书	充电	充电	充电	充电	读书	读书	充电	充电	充电

一整段

一整段

```
1  /*
2  识别序列切割，连续读+连续充电
3  h[t] 第t分钟用于充电后恰为满电时最多读了几页
4  输入n=5,m=2
5      t=0,1,2,3,4,5
6  d[t]=0,5,3,4,2,10
7  h[t]=0 0 5 
8
9  h[t]=max(h[t-1], max{h[t-2*x]+ d[t-2*x+1]+...+d[t-x] } ) ,
10  其中x=1,2,...,t/2
11  */
```

```
19 cin>>n>>m;
20 for(int t=1;t<=n;++t)cin>>d[t];
21 for(int t=1;t<=n;++t)s[t]=s[t-1]+d[t];
22 for(int t=1;t<=n;++t){
23     h[t]=h[t-1];
24     for(int x=1;x*2<=t;++x)
25         h[t]=max(h[t],h[t-2*x]+s[t-x]-s[t-2*x]);
26 }
27 cout<<h[n]<<endl;
```

```
38 bool allEq(){
39     for(int t=1;t<n;++t)
40         if(d[t]!=d[t+1])return 0;
41     return 1;
42 }
43 int main(){
44     freopen("reader.in","r",stdin);
45     freopen("reader.out","w",stdout);
46     cin>>n>>m;
47     for(int t=1;t<=n;++t)cin>>d[t];
48     if(allEq()) ←
49         cout<<d[1]*n/2<<endl;
50     else if(m==1) ←
51         solveM1();
52     else ←
53         solve();
54     return 0;
55 }
```

特殊数据
分类讨论

哪怕正解写错
都有部分得分

2218

missing

请同学写出题目大意
已知什么求什么

共 n 个物品, $v[i]$ 表示 i 号物品体积,箱子容积为 m ,求消失单个物品的背包方案计数。

请同学讨论部分分策略

对于30%数据, $n, m \leq 100$

对于100%数据 $n, m \leq 2000$, $1 \leq v[i] \leq m$

基本功

背包计数问题模板

$f[i][j]$ 表示考虑前 i 件物品
恰好凑成体积 j 的方案数取模

```
17 f[0][0]=  
18 for(int i=1;i<=n;++i)  
19     for(int j=0;j<=m;++j)  
20         if(j<v[i])f[i][j]=  
21         else f[i][j]=
```

对于k号物品消失的情况
重新求一边背包计数问题

$f[i][j]$ 表示考虑前i件物品
但不能用k号物品
恰好凑成体积j的方案数取模

```
17 f[0][0]=1;
18 for(int k=1;k<=n;++k){
19     for(int i=1;i<=n;++i){
20         if(i==k){
21             
22             continue;
23         }
24         for(int j=0;j<=m;++j)
25             if(j<v[i])f[i][j]=f[i-1][j];
26             else f[i][j]=(f[i-1][j]+f[i-1][j-v[i]])%MOD;
27     }
28     for(int j=1;j<m;++j)cout<<f[n][j]<<" ";
29     cout<<f[n][m]<<endl;
30 }
```

正解思路的形成

手算表格启发思路

$f[n][j]$ 表示考虑所有 n 件物品
恰好凑成体积 j 的方案数取模

$g[j]$ 表示缺了 k 号物品后用其他物品
恰好凑成体积 j 的方案数取模

$$v[k]=2$$

	$j=0$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$f[n][j]=$	1	10	8	22	100	52
$g[j]=$						

$f[n][j]$ 表示考虑所有 n 件物品
恰好凑成体积 j 的方案数取模

$g[j]$ 表示缺了 k 号物品后用其他物品
恰好凑成体积 j 的方案数取模

$$v[k]=2$$

	$j=0$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$f[n][j]=$	1	10	8	22	100	52
$g[j]=$	1					

因为 体积 $j=0$ 小于 k 号物品体积 $v[k]=2$
所以 $f[n][0]$ 方案里不可能包含用了 k 号物品的方案
 $g[0]=f[n][0]$

$f[n][j]$ 表示考虑所有 n 件物品
恰好凑成体积 j 的方案数取模

$g[j]$ 表示缺了 k 号物品后用其他物品
恰好凑成体积 j 的方案数取模

$$v[k]=2$$

	$j=0$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$f[n][j]=$	1	10	8	22	100	52
$g[j]=$	1	10				

因为 体积 $j=1$ 小于 k 号物品体积 $v[k]=2$
所以 $f[n][1]$ 方案里不可能包含用了 k 号物品的方案
 $g[1]=f[n][1]$

$f[n][j]$ 表示考虑所有 n 件物品
恰好凑成体积 j 的方案数取模

$g[j]$ 表示缺了 k 号物品后用其他物品
恰好凑成体积 j 的方案数取模

$$v[k]=2$$

	$j=0$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$f[n][j]=$	1	10	8	22	100	52
$g[j]=$	1	10	7			

因为 体积 $j=2$ 等于 k 号物品体积 $v[k]=2$
所以 $f[n][2]$ 方案里可能包含用了 k 号物品的方案

$$\begin{aligned} g[2] &= f[n][2] - \text{用了} k \text{号物品的方案} \\ &= f[n][2] - g[2 - v[k]] \\ &= f[n][2] - g[0] \end{aligned}$$

$f[n][j]$ 表示考虑所有 n 件物品
恰好凑成体积 j 的方案数取模

$g[j]$ 表示缺了 k 号物品后用其他物品
恰好凑成体积 j 的方案数取模

$$v[k]=2$$

	$j=0$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$f[n][j]=$	1	10	8	22	100	52
$g[j]=$	1	10	7	12		

因为 体积 $j=3$ 大于 k 号物品体积 $v[k]=2$
所以 $f[n][3]$ 方案里可能包含用了 k 号物品的方案

$$\begin{aligned} g[3] &= f[n][3] - \text{用了} k \text{号物品的方案} \\ &= f[n][3] - g[3 - v[k]] \\ &= f[n][3] - g[1] \end{aligned}$$

$f[n][j]$ 表示考虑所有 n 件物品
恰好凑成体积 j 的方案数取模

$g[j]$ 表示缺了 k 号物品后用其他物品
恰好凑成体积 j 的方案数取模

$$v[k]=2$$

	$j=0$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$f[n][j]=$	1	10	8	22	100	52
$g[j]=$	1	10	7	12	93	

因为 体积 $j=4$ 大于 k 号物品体积 $v[k]=2$
所以 $f[n][4]$ 方案里可能包含用了 k 号物品的方案

$$\begin{aligned} g[4] &= f[n][4] - \text{用了} k \text{号物品的方案} \\ &= f[n][4] - g[4 - v[k]] \\ &= f[n][4] - g[2] \end{aligned}$$

$f[n][j]$ 表示考虑所有 n 件物品
恰好凑成体积 j 的方案数取模

$g[j]$ 表示缺了 k 号物品后用其他物品
恰好凑成体积 j 的方案数取模

$$v[k]=2$$

	$j=0$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$f[n][j]=$	1	10	8	22	100	52
$g[j]=$	1	10	7	12	93	40

因为 体积 $j=5$ 大于 k 号物品体积 $v[k]=2$
所以 $f[n][5]$ 方案里可能包含用了 k 号物品的方案

$$\begin{aligned} g[5] &= f[n][5] - \text{用了} k \text{号物品的方案} \\ &= f[n][5] - g[5 - v[k]] \\ &= f[n][5] - g[3] \end{aligned}$$

$f[n][j]$ 表示考虑所有 n 件物品
恰好凑成体积 j 的方案数取模

$g[j]$ 表示缺了 k 号物品后用其他物品
恰好凑成体积 j 的方案数取模

总结递推式

当 $j < v[k]$

$f[n][j]$ 方案里不可能包含用了 k 号物品的方案

当 $j \geq v[k]$

$f[n][j]$ 方案里可能包含用了 k 号物品的方案

$$g[j] = f[n][j] - g[j - v[k]]$$

用了 k 号物品的方案数

$f[n][j]$ 表示考虑所有 n 件物品
恰好凑成体积 j 的方案数取模

```
16 for(int i=1;i<=n;++i)cin>>v[i];  
17 f[0][0]=1;  
18 for(int i=1;i<=n;++i)  
19     for(int j=0;j<=m;++j)  
20         if(j<v[i])f[i][j]=f[i-1][j];  
21         else f[i][j]=(f[i-1][j]+f[i-1][j-v[i]])%MOD;
```

预计算

```
16 for(int i=1;i<=n;++i)cin>>v[i];
17 f[0][0]=1;
18 for(int i=1;i<=n;++i)
19     for(int j=0;j<=m;++j)
20         if(j<v[i])f[i][j]=f[i-1][j];
21         else f[i][j]=(f[i-1][j]+f[i-1][j-v[i]])%MOD;
22 for(int k=1;k<=n;++k){
23     for(int j=0;j<v[k];++j)
24     for(int j=v[k];j<=m;++j)
25
26     for(int j=1;j<m;++j)cout<<g[j]<<" ";
27     cout<<g[m]<<endl;
28 }
```

代码优化

一维数组
覆盖填写

```
17 f[0]=1;
18 for(int i=1;i<=n;++i)
19     for(int j=m;j>=v[i];--j)
20         (f[j]+=f[j-v[i]])%=MOD;
21 for(int k=1;k<=n;++k){
22     for(int j=0;j<v[k];++j)g[j]=f[j];
23     for(int j=v[k];j<=m;++j)
24         
25     for(int j=1;j<m;++j)cout<<g[j]<<" ";
26     cout<<g[m]<<endl;
27 }
```


1812