信奥算法

星期几

请同学写出题目大意
已知什么求什么

给出年、月、日，求出该日期是周几。

请同学阅读[数据规模和约定]
有什么收获?

【数据规模与约定】
对于100%的数据，给出的日期一定是在1971到2100年之间的有效日期。

世纪年只有1900(x) 2000 2100。

就算没有样例1,查看右下角时间不难得到1971.1.1

朴素枚举

计算日期差？
1971~2100最多就130年 不妨直接枚举 年[1971, year) 月[1, month) 且题目范围只有2100年是个例外

经验：做题要时刻紧盯题目的数据范围

完善程序练习

```
 7    string w[7] = {                                                    };
 8    int d[14]   = {
 9
10    int year, month, day;
11    cin >> year >> month >> day;
12
13    for (int i = 1971; i < year; ++i) {
14        day += 365;
15
16    }
17
18    for (int i = 1; i < month; ++i) {
19        day += d[i];
20    }
21
22    if (year % 4 == 0                          ) day ++;
23
24    cout << w[          ] << endl;
```

设计测试点

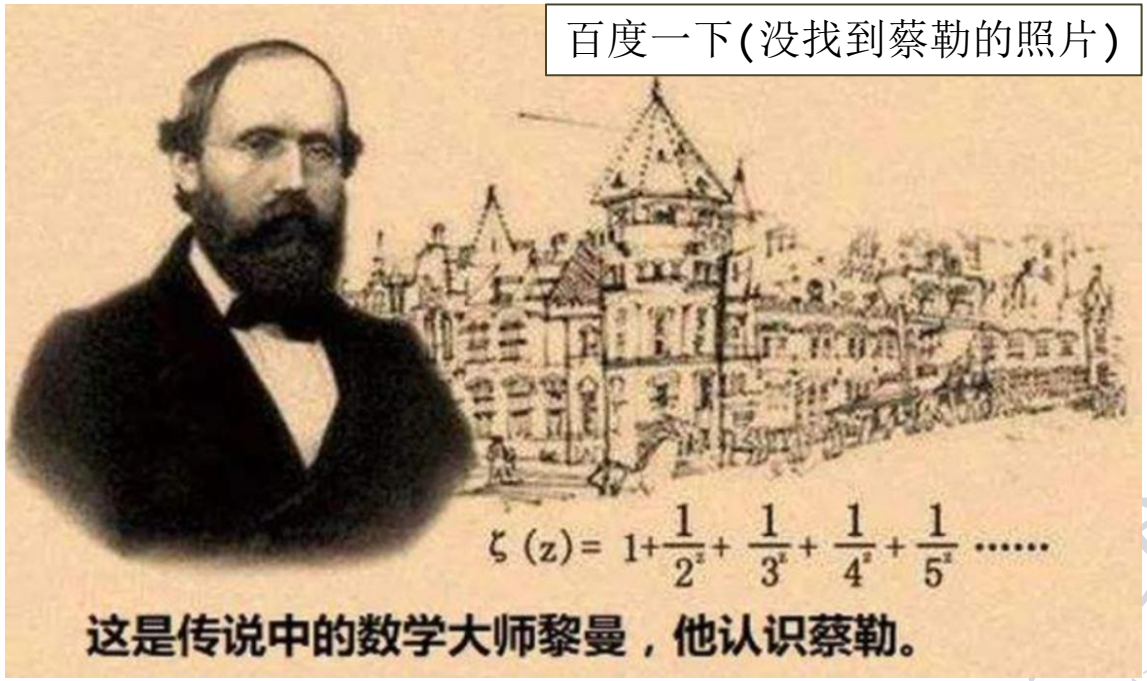
1972.2.1
1972.2.29
1972.3.1
2000.2.1
2000.2.29
2000.7.1

1971.2.1
1971.2.28
1971.3.1
2100.2.1
2100.2.28
2100.12.31

蔡勒公式

嘲笑这题太简单？

蔡勒以一行简洁明快的公式告诉我们
编程的尽头是数学！



太戈编程

2795

高频词汇

请同学写出题目大意
已知什么求什么

给出若干个单词，求出现频率最高的前K个
典型TopK问题

请同学思考该如何处理输入？

方案一:单词数量未知，考虑while-cin结构
但K的值是在第二行给出，考虑字符串转数字

方案二:把第一行读进一个字符串S
然后遍历S，提取单词

map做频率统计

TopK问题	求出现频率最高的K个字符串

常用方法：

map用于统计次数很方便

map映射

每个	字符串	对应一个	出现次数

```
map<string,int> cnt;
```

```
string s;
while(cin>>s) cnt[s]++;
```

cnt[s]对应字符串s出现的次数

while搭配cin处理 不能提前预知 输入个数的情况	DevC++里运行 输入结束时 按Ctrl Z再换行

处理输入

(while-cin)+字符串转数字

```
11      int k = 0;
12      string s;
13      while (cin >> s) {
14          if ('0' <= s[0] && s[0] <= '9') {
15              for (int i = 0; i < s.size(); ++i) {
16                  k = k * 10 +
17              }
18              break;
19          }
20
21      }
```

遍历map

迭代器iterator

key-value(first-second)
一对键值(pair)

```
23      vector<string> v;
24
25      map<string, int>::iterator it;
26      for(it=cnt.begin();it!=cnt.end();++it) {
27          v.push_back(                    );
28      }
29
30      sort(                          cmp);
31
32      for (int i = 0; i < k; ++i) {
33          cout << v[i] << endl;
34      }
```

for(auto & x : cnt)

排序

sort自定义排序规则

```
3  map<string, int> cnt;
4  bool cmp(string x, string y) {
5      return [          ] || cnt[x] == cnt[y] && [      ];
6  }
```

把map里的内容重新捆绑成结构体
然后对结构体数组排序也是可以的

太戈编程

2795

跳来跳去

破题

请同学简述题意
突出核心要点

记忆化
搜索

破题

记忆化搜索

bfs 队列 位置+步数+来时的方向

```
int q[1000001][4];
//q[i][0]存横坐标，q[i][1]存纵坐标，q[i][2]存上一个点的最少步数，q[i][3]存上一次的方向
```

记忆化 访问数组

```
bool v[101][101][8];
```

方向 方向数组

```
int ll[8]={1,1,-1,-1,1,-1,0,0};
int rr[8]={0,1,-1,0,-1,1,-1,1};
```

最优化剪枝

```
if(d[m][n]>0) return;//有值就结束
```

代码

100分代码

```
void bfs()
{
    int st=1,en=1;
    q[1][0]=1;
    q[1][1]=1;
    q[1][3]=-1;
    while(st<=en)
    {
        for(int i=0;i<8;i++)//枚举各个方向
          if(i!=q[st][3])//不是上一次的方向
          {
              int x=q[st][0]+ll[i]*a[q[st][0]][q[st][1]];
              int y=q[st][1]+rr[i]*a[q[st][0]][q[st][1]];
              if(x>=1&&x<=m&&y>=1&&y<=n&&!v[x][y][i])
              {
                  v[x][y][i]=1;//这个点从这个方向来过了
                  d[x][y]=q[st][2]+1;
                  en++;
                  q[en][0]=x;
                  q[en][1]=y;
                  q[en][2]=d[x][y];
                  q[en][3]=i;
              }
              if(d[m][n]>0) return;//有值就结束
          }
        st++;
    }
    return;
}
```

太戈编程

2203

魔法

请同学简述题意
突出核心要点

已知n个节点，m条权值为ti的有向边，有k次将权值一次性反转机会，求从1到n的最短路径是多少

1 ≤ n ≤ 100;
1 ≤ m ≤ 2500;
0 ≤ K ≤ 10^6;
1 ≤ ui,vi ≤ n;
1 ≤ ti ≤ 10^9。
数据保证图中无自环，无重边，至少存在一条从 1 号城市到达 n 号城市的路径。

K=0

30分

图论要素识别

有向图

正权图

单源单汇

n<=100, m<=2500

算法选择

Dijkstra朴素版

复杂度$O(V^2)$

Floyd-Warshall算法

复杂度$O(V^3)$

K=0

Dijkstra朴素版

复杂度O(V^2)

30分

```
fill(dst,dst+n+9,INF);
dst[1]=0;
for(int k=1;k<=n-1;k++){
    int u=n+1;
    for(int j=1;j<=n;j++)
        if(!ok[j]&&dst[j]<dst[u])u=j;
    if(dst[u]>=INF)break;
    ok[u]=1;
    for(int i=0;i<to[u].size();i++)
        dst[to[u][i]]=min(dst[to[u][i]],dst[u]+w[u][i]);
}
```

K=0

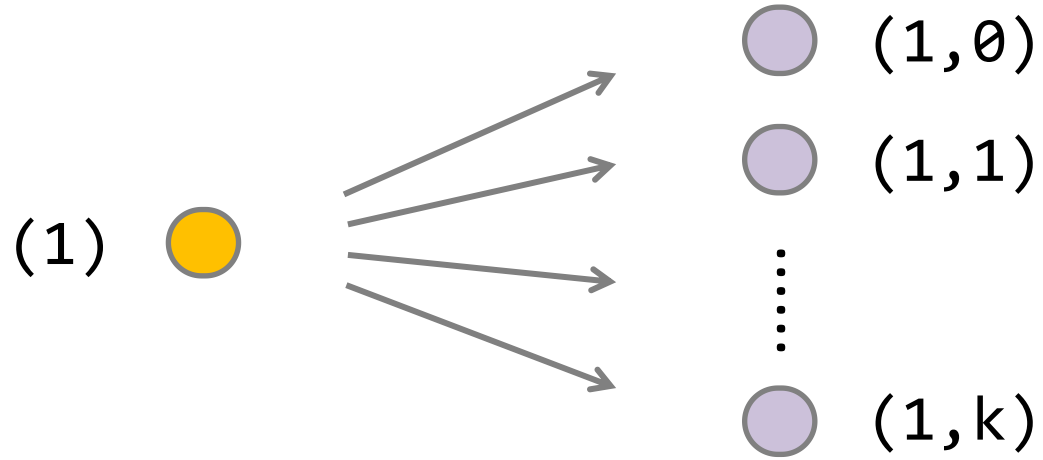
30分

Floyd-Warshall

```
11      cin>>n>>m>>k;
12      for(int i=1; i<N; i++)
13          for(int j=1; j<N; j++)
14              f[i][j]=INF;
15      for(int i=1; i<=m; i++) {
16          int t,u,v;
17          cin>>u>>v>>t;
18          f[u][v]=t;
19      }
20      for(int k2=1; k2<=n; k2++)
21          for(int i=1; i<=n; i++)
22              for(int j=1; j<=n; j++)
23          f[i][j]=min(f[i][j],f[i][k2]+f[k2][j]);
24      cout<<f[1][n]<<endl;
25      return 0;
26  }
```

拆点

原图中1个节点对应
新图中多个节点



拆点 | 分层图 | 状态具体化

算法1

80分

分层图

从1号节点
到n号节点
最多反转k次
最短路径

从(1,0)状态
到(n,?)
其中之一的状态
最短路径

f[i][j]:从1号走到i号节点，使用了k次反转的最短路径长度

选k到i边的原始路径

$t(k,i)$:从k到i原始路径长度

$f[i][j]: \min(f[i][j], f[k][j-1] - t(k,i))$

K到i边的没有原始路径

$f[i][j]: \min(f[i][j], f[k][j] + \mathrm{d(k,i)}$

$d(k,i)$:从k到i路径长度

算法1

80分

分层图

```
for(int i=1; i<=n; i++) f[i][0]=d[1][i];
for(int j=1; j<=k; j++) {
    for(int i=1; i<=n; i++)
        for(int k2=1; k2<=n; k++)
            f[i][j]=min(f[i][j],(a[k2][i]>0 ? f[k2][j-1]-a[k2][i] : INF))
    for(int i=1; i<=n; i++)
        for(int k2=1; k2<=n; k++)
            f[i][j]=min(f[i][j],f[k2][j]+d[k2][i]);
}
```

算法2

100分

矩阵快速幂

k太大了，转移次数必须小于线性。考虑把它优化至矩阵乘法

F[k][i][j]：从i到j至多用了k次魔法的最短路径

假设从s到t至多用了k-1次魔法，从t到v至多用了1次魔法

F[k][s][v]=min(F[k][s][v],F[k-1][s][t]+F[1][t][v])

F[k]=F[k-1]MULF[1]　(MUL为重定义后的运算符)

F[k]=k个F[1] MUL后的结果

算法2

100分

矩阵快速幂

STEP1:求解F[1]矩阵的情况，即最多反转1条边的情况

```
for(ll cc=1;cc<=n;cc++){
    for(ll i=1;i<=n;i++){
        for(ll j=1;j<=n;j++){
            dis[i][j]=min(dis[i][j],dis[i][cc]+dis[cc][j]);//floyd
        }
    }
}
memcpy(f,dis,sizeof(dis));
for(ll i=1;i<=m;i++){
    ll u=e[i].from,v=e[i].to,w=e[i].val;
    for(ll j=1;j<=n;j++)
        for(ll cc=1;cc<=n;cc++)
            f[j][cc]=min(f[j][cc],dis[j][u]-w+dis[v][cc]);//F[1]
```

算法2

100分

矩阵快速幂

STEP2:用F[1]矩阵递推求得F[k]矩阵

快速幂：顾名思义，快速幂就是快速算底数的n次幂。其时间复杂度为 $O(\log_2 N)$， 与朴素的O(N)相比效率有了极大的提高

```
void Mul(ll d[N][N],ll a[N][N],ll b[N][N]){
    ll t[N][N];
    memset(t,0x3f,sizeof(t));
    for(ll i=1;i<=n;i++)
        for(ll j=1;j<=n;j++)
            for(ll k=1;k<=n;k++)
                t[i][j]=min(t[i][j],a[i][k]+b[k][j]);//重定义后的矩
    memcpy(d,t,sizeof(t));

    for(;k;k>>=1){//矩阵快速幂
        if(k&1)Mul(dis,dis,f);
        Mul(f,f,f);
    }
```

结合律

太戈编程

2195