Template V1.23

Judge

September 23, 2016

目录

1	DLX 1			
	1.1	精确覆盖 DLX	. 1	
	1.2	重复覆盖 DLX	. 1	
2	Gau	ISS	2	
-	2.1	高斯 double		
	$\frac{2.1}{2.2}$	高斯消元		
	2.3	高斯消元 mod		
	2.4	高斯消元 xor		
	2.4	同为[行/L XOI · · · · · · · · · · · · · · · · · · ·	. 4	
3	NTT			
	3.1	FFT		
	3.2	NTT	. 5	
	3.3	分治 +fft	. 6	
4	math			
_	4.1	CRT		
	4.2	LL 黑科技		
	4.3	about prime		
	4.4	ext_gcd		
	4.5	$\operatorname{inv}(\operatorname{gcd})$		
	4.6	phi		
	1.0	pm		
5	spfa		8	
	5.1	Spfa 最短路	. 8	
6	划分	树	9	
	6.1	划分树比 k 小数量	. 9	
	6.2	划分树第 k 大		
7	匈牙	: ∔ i	10	
'		二分匹配	. 10	
	1.1		. 10	
8	矩阵		10	
	8.1	矩阵	. 10	
9	组合		11	
		组合数求 mod	. 11	
10	网络	流	11	
10		MinCost bhb		
	10.2	unine biib	. 11	

1 DLX

1.1 精确覆盖 DLX

```
1
    using namespace std;
    #define N 1111111
3
    #define M 8111111
    const int MaxM = 1111;
    const int MaxN = 1111;
    const int maxnode = MaxN * MaxM;
 6
    int flag;
8
    struct DLX
9
10
         int n, m, size;
         int U[maxnode], D[maxnode], R[maxnode], L[maxnode], Row[
11
         maxnode], Col[maxnode];
         int H[MaxN],S[MaxM];
12
         int ansd:
13
14
         void init(int _n,int _m)
15
16
             n = _n;
             m = _m;
17
             for(int i = 0;i <= m;i++)</pre>
18
19
                  S[i] = 0;
20
21
                  U[i] = D[i] = i;
                  L[i] = i-1;
22
                  R[i] = i+1;
23
24
25
             R[m] = 0; L[0] = m;
             size = m;
26
27
             for(int i = 1; i \le n; i++)H[i] = -1;
28
         void Link(int r,int c)
29
30
             ++S[Col[++size]=c];
31
32
             Row[size] = r;
             D[size] = D[c];
33
34
             U[D[c]] = size;
35
             U[size] = c;
36
             D[c] = size;
             if(H[r] < 0)H[r] = L[size] = R[size] = size;
37
38
39
40
                  R[size] = R[H[r]];
41
                  L[R[H[r]]] = size;
                  L[size] = H[r];
42
43
                  R[H[r]] = size;
44
             }
45
         void remove(int c)
46
47
             L[R[c]] = L[c]; R[L[c]] = R[c];
48
             for(int i = D[c];i != c;i = D[i])
                  for(int j = R[i]; j != i; j = R[j])
50
51
                      U[D[j]] = U[j];
52
                      D[U[j]] = D[j];
53
54
                        -S[Col[j]];
                  }
55
56
         void resume(int c)
57
58
59
             for(int i = U[c];i != c;i = U[i])
                  for(int j = L[i];j != i;j = L[j])
     ++S[Col[U[D[j]]=D[U[j]]=j]];
60
61
             L[R[c]] = R[L[c]] = c;
62
63
         int ans[MaxN];
64
         void Dance(int d)
66
             if (flag) return;
67
             if(R[0] == 0)
68
69
70
                  flag=1;
                  printf("%d ",d);
71
72
                  rep(i,0,d)
73
                  printf("%d%c", ans[i], i==d-1?'\n':' ');
74
                  return:
75
76
             int c = R[0];
             for(int i = R[0]; i != 0; i = R[i])
```

```
if(S[i] < S[c])
                                                                    79
                 c = i;
         remove(c);
                                                                    80
         for(int i = D[c]; i != c; i = D[i])
                                                                    81
                                                                    82
             for(int j = R[i]; j != i; j = R[j]) remove(Col[j
                                                                    83
             1);
             ans[d]=Row[i];
                                                                    84
             Dance(d+1);
                                                                    85
             for(int j = L[i];j != i;j = L[j]) resume(Col[j
                                                                    86
                                                                    87
         resume(c);
                                                                    88
                                                                    89
}dlx;
                                                                    90
int main()
                                                                    91
                                                                    92
  int n.m:
                                                                   93
  while (~scanf("%d%d",&n,&m))
                                                                    94
                                                                    95
    dlx.init(n,m);
                                                                    96
    rep(i,0,n)
                                                                    97
                                                                    98
       int k,x;
                                                                    99
       scan(&k);
                                                                    100
       rep(j,0,k)
                                                                    101
                                                                    102
         scan(&x);
                                                                    103
         dlx.Link(i+1,x);
                                                                    104
                                                                    105
                                                                    106
    flag=0;
                                                                    107
    dlx.Dance(0);
                                                                    108
    if (!flag) puts("NO");
                                                                    109
                                                                    110
    return 0;
                                                                   111
}
                                                                   112
                                                                    113
6 5
                                                                   114
1 2
                                                                   115
2 3
                                                                    116
3 4
                                                                   117
4 5
                                                                    118
3 5
                                                                    119
2 4 5 6
                                                                   120
                                                                    121
```

1.2 重复覆盖 DLX

```
const int MaxM = 15*15+10;
const int MaxN = 15*15+10;
const int maxnode = MaxN * MaxM;
const int INF = 0x3f3f3f3f;
                                                                  4
struct DLX
                                                                  5
    int n,m,size;
    int U[maxnode], D[maxnode], R[maxnode], L[maxnode], Row[
                                                                  8
    maxnode],Col[maxnode];
    int H[MaxN],S[MaxM];
                                                                  9
    int ansd;
                                                                  10
    void init(int _n,int _m)
                                                                  11
                                                                  12
        n = _n;
                                                                  13
        m = _m;
                                                                  14
        for(int i = 0;i <= m;i++)</pre>
                                                                  15
                                                                  16
             S[i] = 0;
                                                                  17
             U[i] = D[i] = i;
             L[i] = i-1;
                                                                  19
             R[i] = i+1;
                                                                  20
        R[m] = 0; L[0] = m;
                                                                  22
        size = m;
                                                                  23
        for(int i = 1; i \le n; i++)H[i] = -1;
                                                                  24
                                                                  25
    void Link(int r,int c)
                                                                  26
                                                                  27
        ++S[Col[++size]=c];
                                                                  28
        Row[size] = r;
                                                                  29
        D[size] = D[c];
                                                                  30
        U[D[c]] = size;
                                                                  31
        U[size] = c;
                                                                  32
        D[c] = size;
                                                                  33
```

```
34
             if(H[r] < 0)H[r] = L[size] = R[size] = size;
35
             else
36
             {
37
                 R[size] = R[H[r]];
38
                 L[R[H[r]]] = size;
39
                 L[size] = H[r];
                 R[H[r]] = size;
40
41
             }
42
         void remove(int c)
43
44
             for(int i = D[c];i != c;i = D[i])
45
                 L[R[i]] = L[i], R[L[i]] = R[i];
46
47
         void resume(int c)
48
49
             for(int i = U[c];i != c;i = U[i])
50
                 L[R[i]] = R[L[i]] = i;
51
52
         bool v[MaxM];
53
         int f()
54
55
         {
             int ret = 0;
56
             for(int c = R[0]; c != 0;c = R[c])v[c] = true;
57
             for(int c = R[0]; c != 0; c = R[c])
58
                 if(v[c])
59
60
61
                      ret++;
                      v[c] = false;
62
                      for(int i = D[c];i != c;i = D[i])
63
64
                          for(int j = R[i]; j != i; j = R[j])
                               v[Col[j]] = false;
65
66
             return ret:
67
68
69
         void Dance(int d)
70
71
             if(d + f() >= ansd)return;
72
             if(R[0] == 0)
73
74
                 if(d < ansd)ansd = d;
                 return:
75
76
             int c = R[0];
77
             for(int i = R[0]; i != 0; i = R[i])
78
79
                 if(S[i] < S[c])
80
                      c = i:
             for(int i = D[c];i != c;i = D[i])
81
82
                 remove(i);
83
                 for(int j = R[i]; j != i; j = R[j])remove(j);
84
                 Dance(d+1);
                 for(int j = L[i];j != i;j = L[j])resume(j);
86
87
                 resume(i);
88
             }
89
         }
    };
90
```

2 Gauss

HIT

2.1 高斯 double

```
///高斯消元模板
    const double eps = 1e-10;
                                ///精度
    double Aug[N][N]; ///增广矩阵
    bool free_x[N];
                           ///判断是否是不确定的变元
    double x[N];
                           ///解集
6
    int sign(double x){ return (x>eps) - (x<-eps);}</pre>
8
    int Gauss(int n, int m)//n 变元数量, m 方程数目
9
10
        memset(x,0,sizeof(x));
        memset(free_x,1,sizeof(free_x));
11
        int row.col.max r:
12
        for(row=0,col=0; row<m&&col<n; row++,col++)</pre>
13
14
           \max_r = row;
15
16
           for(int i = row+1; i < m; i++)</pre>
            {///找到当前列所有行中的最大值(做除法时减小误差)
17
               if(sign(fabs(Aug[i][col]) - fabs(Aug[max_r][
18
                col])) > 0)
19
                   \max r = i
```

```
if(max_r != row){///将该行与当前行交换
                                                           21
           for(int j = row; j < n+1; j++)
                                                           22
               swap(Aug[max_r][j],Aug[row][j]);
                                                           23
                                                           24
       if(sign(Aug[row][col])==0)
                                                           25
       {///当前列row行以下全为0(包括row行)
                                                           26
           row-:
                                                           27
           continue;
                                                           28
                                                           29
       for(int i = row+1; i < m; i++)</pre>
                                                           30
                                                           31
           if(sign(Aug[i][col])==0)
                                                           32
               continue;
                                                           33
           double ta = Aug[i][col]/Aug[row][col];
                                                           34
           for(int j = col; j < n+1; j++)</pre>
                                                           35
               Aug[i][j] -= Aug[row][j]*ta;
                                                           36
       }
                                                           37
                                                           38
    ///无解或者多个解的情况
                                                           39
   for(int i = row; i < m; i++)</pre>
                                                           40
                                                           41
       if(sign(Aug[i][col]))
                                                           42
           return -1;///col=n存在0...0,a的情况,无解
                                                           43
                                                           44
   if(row < n)
                                                           45
                                                           46
       for(int i = row-1; i >=0; i---)
                                                           47
                                                           48
           int free_num = 0;
                               ///自由变元的个数
                                                           49
           int free_index;
                              ///自由变元的序号
                                                           50
           for(int j = 0; j < n; j++)
                                                           51
           {
                                                           52
               if(sign(Aug[i][j])!=0 && free_x[j])
                                                           53
                   free_num++, free_index=j;
                                                           54
           if(free_num > 1) continue; ///该行中的不确定
                                                           56
            的变元的个数超过1个,无法求解,它们仍然为不确定
           ///只有一个不确定的变元free_index, 可以求解出该
                                                           57
            变元,且该变元是确定的
           double tmp = Aug[i][n];
                                                           58
           for(int j = 0; j < n; j++)
                                                           59
                                                           60
           {
               if(sign(Aug[i][j])!=0 && j!=free_index)
                                                           61
                   tmp -= Aug[i][j]*x[j];
                                                           62
                                                           63
           x[free_index] = tmp/Aug[i][free_index];
                                                           64
           free_x[free_index] = false;
                                                           65
                                                           66
       return n-row;///存在0...0,0的情况,有多个解,自由变
                                                           67
        元个数为n-row个
                                                           68
    ///无解或者多个解的情况
                                                           69
   for(int i = n-1; i >= 0; i--)
                                                           70
                                                           71
       double tmp = Aug[i][n];
                                                           72
       for(int j = i+1; j < n; j++)</pre>
                                                           73
           if(sign(Aug[i][j])!=0)
                                                           74
               tmp -= Aug[i][j]*x[j];
                                                           75
       x[i] = tmp/Aug[i][i];
                                                           76
                                                           77
    return 0;///有且仅有一个解,严格的上三角矩阵(n==m)
                                                           78
}
                                                           79
```

2.2 高斯消元

```
const int MAXN=50;
namespace gaosi
    int a[MAXN][MAXN];//增广矩阵
    int x[MAXN];//解集
                                                            5
    bool free_x[MAXN];//标记是否是不确定的变元
                                                            8
    void Debug(void)
                                                            9
                                                           10
        int i, j;
                                                           11
        for (i = 0; i < equ; i++)
                                                            12
                                                           13
           for (j = 0; j < var + 1; j++)
                                                            14
                                                            15
               cout << a[i][j] << " ";
                                                           16
```

```
17
               cout << endl;
18
19
20
           cout << endl;
21
22
23
     void init(){memset(a, 0, sizeof(a));};
24
       inline int gcd(int a,int b)
25
26
           int t:
27
           while(b!=0)
28
           {
               t=b:
29
               b=a%b;
30
               a=t;
31
32
           return a;
33
34
       inline int lcm(int a,int b)
35
36
           return a/gcd(a,b)*b;//先除后乘防溢出
37
38
39
       // 高斯消元法解方程组(Gauss—Jordan elimination).(—2表
40
        示有浮点数解, 但无整数解,
       //-1表示无解,0表示唯一解,大于0表示无穷解,并返回自由
41
        变元的个数)
       //有equ个方程,var个变元。增广矩阵行数为equ,分别为0到
42
       equ-1, 列数为var+1, 分别为0到var.
       int Gauss(int equ,int var)
43
44
45
           int i, j, k;
           int max_r;// 当前这列绝对值最大的行.
46
           int col;//当前处理的列
47
48
           int ta, tb;
49
           int LCM;
           int temp;
50
           int free_x_num;
51
           int free_index;
52
53
54
           for(int i=0;i<=var;i++)</pre>
55
56
               x[i]=0;
57
               free_x[i]=true;
           }
58
59
           //转换为阶梯阵.
60
           col=0; // 当前处理的列
61
           for(k = 0; k < equ \&\& col < var; k++, col++)
62
           {// 枚举当前处理的行.
63
       // 找到该co1列元素绝对值最大的那行与第k行交换.(为了在
64
        除法时减小误差)
65
               max r=k:
66
               for(i=k+1;i<equ;i++)</pre>
67
                   if(abs(a[i][col])>abs(a[max_r][col]))
68
                   max_r=i;
69
               if(max_r!=k)
70
               {// 与第k行交换.
                   for(j=k;j<var+1;j++) swap(a[k][j],a[max_r</pre>
72
73
               if(a[k][col]==0)
74
               {// 说明该col列第k行以下全是0了,则处理当前行
75
                的下一列.
76
                   continue;
78
79
               for(i=k+1;i<equ;i++)</pre>
               {// 枚举要删去的行.
80
                   if(a[i][col]!=0)
81
82
                   {
                       LCM = lcm(abs(a[i][col]), abs(a[k][col
83
                       ]));
                       ta = LCM/abs(a[i][col]);
84
                       tb = LCM/abs(a[k][col]);
85
86
                       if(a[i][col]*a[k][col]<0)tb=-tb;//异号
                       的情况是相加
                       for(j=col;j<var+1;j++)</pre>
87
88
                       {
                           a[i][j] = a[i][j]*ta-a[k][j]*tb;
89
90
```

```
}
    }
                                              92
 }
                                              93
                                              94
// Debug();
                                              95
                                              96
 // 1. 无解的情况: 化简的增广阵中存在(0, 0, ..., a)
                                              97
  这样的行(a != 0).
 for (i = k; i < equ; i++)</pre>
                                              98
 { // 对于无穷解来说,如果要判断哪些是自由变元,那
                                              99
  么初等行变换中的交换就会影响,则要记录交换.
     if (a[i][col] != 0) return -1;
                                              100
                                              101
 // 2. 无穷解的情况: 在var * (var + 1)的增广阵中出
                                              102
  现(0, 0, ..., 0)这样的行,即说明没有形成严格的上三
  角阵
 // 且出现的行数即为自由变元的个数.
                                              103
 if (k < var)
                                              104
                                              105
     // 首先,自由变元有var - k个,即不确定的变元至
                                              106
     少有var - k个.
     for (i = k - 1; i \ge 0; i - )
                                              107
                                              108
        // 第i行一定不会是(0, 0, ..., 0)的情况,因
                                              109
        为这样的行是在第k行到第equ行.
        // 同样, 第i行一定不会是(0, 0, ..., a), a
                                              110
        != 0的情况,这样的无解的.
        free_x_num = 0; // 用于判断该行中的不确定
                                              111
        的变元的个数,如果超过1个,则无法求解,它
        们仍然为不确定的变元.
        for (j = 0; j < var; j++)
                                              112
                                              113
           if (a[i][j] != 0 && free_x[j])
                                              114
           free_x_num++, free_index = j;
                                              115
        if (free_x_num > 1) continue; // 无法求解
                                              116
        出确定的变元.
        // 说明就只有一个不确定的变元free_index,
                                              117
        那么可以求解出该变元,且该变元是确定的.
        temp = a[i][var];
                                              118
        for (j = 0; j < var; j++)
                                              119
                                              120
           if (a[i][j] != 0 && j != free_index)
                                              121
           temp -= a[i][j] * x[j];
                                              122
        x[free\_index] = temp / a[i][free\_index];
                                              123
        // 求出该变元.
        free_x[free_index] = 0; // 该变元是确定的.
                                              124
                                              125
     return var - k; // 自由变元有var - k个.
                                              126
                                              127
 // 3. 唯一解的情况: 在var * (var + 1)的增广阵中形
                                              128
 成严格的上三角阵.
 // 计算出Xn-1, Xn-2 ... X0.
                                              129
 for (i = var - 1; i >= 0; i-
                                              130
                                              131
     temp = a[i][var];
                                              132
     for (j = i + 1; j < var; j++)
                                              133
                                              134
        if (a[i][j] != 0) temp -= a[i][j] * x[j];
                                              135
                                              136
     137
     浮点数解,但无整数解.
     x[i] = temp / a[i][i];
                                              138
                                              139
 return 0;
                                              140
                                              141
                                              142
```

2.3 高斯消元 mod

}

```
13
           b=a%b;
14
           a=t;
15
16
       return a;
17
   inline int lcm(int a,int b)
18
19
       return a/gcd(a,b)*b;//先除后乘防溢出
20
21
22
   // 高斯消元法解方程组(Gauss-Jordan elimination).(-2表示有
23
    浮点数解,但无整数解,
   //-1表示无解,0表示唯一解,大于0表示无穷解,并返回自由变元
24
    的个数)
   //有equ个方程,var个变元。增广矩阵行数为equ,分别为0到equ
25
    -1, 列数为var+1, 分别为0到var.
   int Gauss(int equ,int var)
26
27
28
       int i, j, k;
       int max_r;// 当前这列绝对值最大的行.
29
       int col;//当前处理的列
30
31
       int ta,tb;
       int LCM;
32
33
       int temp;
       int free_x_num;
34
       int free_index;
35
36
37
       for(int i=0;i<=var;i++)</pre>
38
39
           x[i]=0;
40
           free_x[i]=true;
41
       col=0;
42
       for(k = 0; k < equ \&\& col < var; k++, col++)
43
44
45
           max r=k;
           for(i=k+1:i<equ:i++)</pre>
46
47
             if(abs(a[i][col])>abs(a[max_r][col])) max_r=i;
48
           if(max_r!=k)
             for(j=k;j<var+1;j++) swap(a[k][j],a[max_r][j]);</pre>
49
50
           if(a[k][col]==0)
51
52
53
               continue:
54
55
           for(i=k+1;i<equ;i++)</pre>
56
57
               if(a[i][col]!=0)
58
59
               {
                   LCM = lcm(abs(a[i][col]), abs(a[k][col]));
60
                   ta = LCM/abs(a[i][col]);
61
                   tb = LCM/abs(a[k][col]);
62
63
                   if(a[i][col]*a[k][col]<0)tb=-tb;//异号的情
                   况是相加
                   for(j=col;j<var+1;j++)</pre>
64
65
                       a[i][j] = ((a[i][j]*ta-a[k][j]*tb)%MOD
66
                       +MOD)%MOD;
67
                   }
               }
68
69
           }
70
71
72
73
       for (i = k; i < equ; i++)</pre>
74
75
           if (a[i][col] != 0) return -1;//无解
76
77
       if (k < var)
78
           // 首先, 自由变元有var - k个, 即不确定的变元至少有
79
           var - k \uparrow.
           for (i = k - 1; i \ge 0; i - )
80
81
               // 第i行一定不会是(0, 0, ..., 0)的情况,因为这
82
               样的行是在第k行到第egu行.
               // 同样, 第1行一定不会是(0, 0, ..., a), a != 0
83
                的情况,这样的无解的.
               free_x_num = 0; // 用于判断该行中的不确定的变
84
                元的个数,如果超过1个,则无法求解,它们仍然为
                不确定的变元.
85
               for (j = 0; j < var; j++)
```

```
{
                                                         86
            if (a[i][j] != 0 && free_x[j]) free_x_num
                                                         87
            ++, free_index = j;
                                                         88
        if (free_x_num > 1) continue; // 无法求解出确
                                                         89
        定的变元.
        // 说明就只有一个不确定的变元free_index,那么
                                                         90
        可以求解出该变元,且该变元是确定的.
        temp = a[i][var];
                                                         91
        for (j = 0; j < var; j++)
                                                         92
                                                         93
            if (a[i][j] != 0 && j != free_index) temp
-= a[i][j] * x[j]%MOD;
                                                         94
            temp=(temp%MOD+MOD)%MOD;
                                                         95
                                                         96
        x[free\_index] = (temp / a[i][free\_index])%MOD;
                                                         97
         // 求出该变元.
        free_x[free_index] = 0; // 该变元是确定的.
                                                         98
                                                         99
    return var - k; // 自由变元有var - k个.
                                                         100
                                                         101
for (i = var - 1; i \ge 0; i - )
                                                         102
                                                         103
    temp = a[i][var];
                                                         104
    for (j = i + 1; j < var; j++)
                                                         105
                                                         106
        if (a[i][j] != 0) temp —= a[i][j] * x[j];
                                                         107
        temp=(temp%MOD+MOD)%MOD;
                                                         108
                                                         109
    while (temp % a[i][i] != 0) temp+=MOD;
                                                         110
   x[i] = (temp / a[i][i])%MOD;
                                                         111
                                                         112
return 0;
                                                         113
                                                         114
```

2.4 高斯消元 xor

```
int equ, var;
                                                                 1
int a[110][110];
int x[110];
                                                                 3
int free_x[110];
int free_num;
                                                                 6
//返回值为–1表示无解,为0是唯一解,否则返回自由变元个数
int Gauss()
                                                                 8
                                                                 9
    int max_r, col, k;
                                                                 10
    free_num = 0;
                                                                 11
    for(k = 0, col = 0; k < equ && col < var; k++, col++)
                                                                 12
                                                                 13
        \max_r = k;
                                                                 14
        for(int i = k+1 ; i < equ; i++)</pre>
             if(abs(a[i][col]) > abs(a[max_r][col]))
                                                                 16
                 \max_r = i;
                                                                 17
        if(a[max_r][col] == 0)
                                                                 18
                                                                 19
        {
                                                                 20
             free_x[free_num++] = col; //自由变元
                                                                 21
             continue:
                                                                 22
                                                                 23
        if(max_r != k)
                                                                 24
                                                                 25
             for(int j = col; j < var+1; j++)</pre>
                                                                 26
                 swap(a[k][j],a[max_r][j]);
                                                                 27
                                                                 28
        for(int i = k+1; i < equ;i++)</pre>
                                                                 29
             if(a[i][col] != 0)
                                                                 30
                 for(int j = col; j < var+1; j++)</pre>
                                                                 31
                     a[i][j] ^= a[k][j];
                                                                 32
                                                                 33
    for(int i = k;i < equ;i++)
        if(a[i][col] != 0)
                                                                 35
             return -1:
                                                                 36
    if(k < var)
                                                                 37
    {//多解求最小1个数
                                                                 38
             int ans = INF;
                                                                 39
             int t=var-k;
                                                                 40
             int tot = (1<<t);
                                                                 41
             for(int i = 0;i < tot;i++)</pre>
                                                                 42
                                                                 43
                 int cnt = 0;
                                                                 44
                 for(int j = 0;j < t;j++)</pre>
                                                                 45
                                                                 46
```

```
47
                             if(i&(1<<j))
48
49
                                  x[free_x[j]] = 1;
50
                                 cnt++;
51
                             else x[free_x[j]] = 0;
52
53
                        for(int j = var-t-1; j \ge 0; j--)
54
55
                             int idx;
56
57
                             for(idx = j;idx < var;idx++)</pre>
58
                                  if(a[j][idx])
59
                                      break:
                             x[idx] = a[j][var];
60
                             for(int 1 = idx+1;1 < var;1++)</pre>
61
62
                                 if(a[j][1])
                                      x[idx] \stackrel{\sim}{} x[1];
63
64
                             cnt += x[idx];
65
66
                        ans = min(ans, cnt);
67
68
                   printf("%d\n", ans);
69
70
              return var-k;
71
         for(int i = var-1; i \ge 0; i \longrightarrow)
72
73
74
              x[i] = a[i][var];
              for(int j = i+1; j < var;j++)</pre>
75
76
                   x[i] ^= (a[i][j] \&\& x[j]);
77
78
         return 0:
    }
```

NTT

3.1 FFT

```
1
    #include <bits/stdc++.h>
    #define maxn 400200
 2
    #define mod 313
    #define PI acos(-1.0)//acosl(-1.0)
    using namespace std;
    typedef double LD;//long double
 6
    typedef long long LL;
 8
    //typedef complex<LD> cpx;
9
    struct cpx
10
      LD x,
11
12
      (}()xap
      cpx(LD x, LD y):x(x),y(y){}
14
15
    cpx operator +(cpx a, cpx b)
16
17
      return cpx(a.x+b.x, a.y+b.y);
18
19
    cpx operator –(cpx a, cpx b)
20
21
      return cpx(a.x-b.x, a.y-b.y);
22
23
    cpx operator *(cpx a, cpx b)
24
      return cpx(a.x*b.x-a.y*b.y, a.x*b.y+a.y*b.x);
25
26
    }
27
    int rev(int x, int n)
28
29
30
         int tmp=0;
31
         for (int i=n>>1;i;i>>=1,x>>=1)
32
             tmp=tmp<<1|x&1;
         return tmp;
33
34
35
    void fft(cpx *a, int n, int flag)
36
37
      for (int i=0, j=i;i<n;i++, j=rev(i, n))</pre>
38
39
        if (i<j) swap(a[i], a[j]);</pre>
40
      for (int k=1;k<n;k<<=1)</pre>
41
42
         cpx wn(cos(PI/k), flag*sin(PI/k));
43
         //cpx wn(cosl(PI/i), flag*sinl(PI/i));
         cpx w(1, 0);
44
```

```
for (int i=0;i<k;i++,w=w*wn)</pre>
      for (int j=i;j<n;j+=(k<<1))</pre>
                                                                    46
                                                                    47
         cpx x=a[j], y=w*a[j|k];
                                                                    48
         a[j]=x+y;
                                                                    49
         a[j|k]=x-y;
                                                                    50
                                                                    51
                                                                    52
  if (flag==-1)
                                                                    53
    for (int i=0;i<n;i++)</pre>
                                                                    54
      a[i].x/=n, a[i].y/=n;
                                                                    55
                                                                    56
                                                                    57
cpx A[maxn], B[maxn];
                                                                    58
int a[maxn], b[maxn], c[maxn];
                                                                    59
int n:
                                                                    60
void roll(int *a, int *b, int *c, int n, int m)
                                                                    62
  int num=1:
                                                                    63
  while (num<n+m) num<<=1;//move out if slow
                                                                    64
  for (int i=0;i<num;i++) A[i]=(i<n)?cpx(a[i],0):cpx(0,0);</pre>
                                                                    65
  for (int i=0;i<num;i++) B[i]=(i<m)?cpx(b[i],0):cpx(0,0);</pre>
                                                                    66
  fft(A, num, 1);
                                                                    67
  fft(B, num, 1);
                                                                    68
  for (int i=0;i<num;i++) A[i]=A[i]*B[i];</pre>
                                                                    69
                                                                    70
  fft(A, num, -1);
  for (int i=0;i<num;i++) c[i]=(LL)(A[i].x+0.5)%mod;</pre>
                                                                    71
                                                                    72
```

2

6

10

11

13

16

18

19

21 22

23

25

26

27

28

29

32

34

35

37

38

41

42

43

44

45

47

48

49

NTT3.2

```
LL quick_mod(LL a, LL b, LL m)
    LL ans = 1;
                                                                 3
    a %= m:
    while(b)
         if(b \& 1) ans = ans * a % m;
         b >>= 1;
        a = a * a % m;
                                                                 9
    return ans;
                                                                 12
namespace ntt{
    const int N = (1 << 19) + 20;
                                                                 14
    const int P = 998244353;
                                                                 15
    const int G = 3;
    const int NUM = 25;
                                                                 17
       wn[NUM];
    LL a[N], b[N];
                                                                 20
    LL mul(LL x,LL y)
    {
         return (x*y-(LL)(x/(long double)P*y+1e-3)*P+P)%P;
    }
    void GetWn()
         for(int i = 0; i < NUM; i++)</pre>
             int t = 1 << i;
                                                                 30
             wn[i] = quick_mod(G, (P - 1) / t, P);
                                                                 31
         }
    }
                                                                 33
    void Rader(LL a[], int len)
                                                                 36
         int j = len >> 1;
        for(int i = 1; i < len - 1; i++)
                                                                 39
             if(i < j) swap(a[i], a[j]);</pre>
             int k = len >> 1;
             while(j >= k)
                   -= k:
                 k >>= 1;
                                                                 46
             if(j < k) j += k;
        }
    }
                                                                 50
                                                                 51
    void NTT(LL a[], int len, int on)
                                                                 52
```

目录

```
53
         {
             Rader(a, len);
 54
 55
             int id = 0;
             for(int h = 2; h <= len; h <<= 1)</pre>
 56
 57
                 id++:
 58
                 for(int j = 0; j < len; j += h)</pre>
 59
 60
 61
                     for(int k = j; k < j + h / 2; k++)
 62
 63
 64
                         LL u = a[k] \% P;
                         LL t = w * a[k + h / 2] % P;
 65
                         a[k] = (u + t) \% P;
 66
                         a[k + h / 2] = (u - t + P) \% P;
 67
                         w = w * wn[id] % P;
 68
 69
                     }
 70
                 }
 71
             if(on == -1)
 72
 73
 74
                 for(int i = 1; i < len / 2; i++)</pre>
                     swap(a[i], a[len - i]);
 75
 76
                 LL inv = quick_mod(len, P - 2, P);
                 for(int i = 0; i < len; i++)</pre>
 77
                     a[i] = a[i] * inv % P;
 78
 79
             }
 80
 81
         void Conv(LL a[], LL b[], int n)
 82
 83
 84
             NTT(a, n, 1);
 85
             NTT(b, n, 1);
             for(int i = 0; i < n; i++)
 86
                 a[i] = a[i] * b[i] % P;
 87
             NTT(a, n, -1);
 88
 89
 90
         void workNTT(LL a[], LL b[],int L1,int L2,int &len)
 91
 92
             GetWn();
 93
             len = 1;
             while(len < 2 * L1 || len <2 * L2) len <<= 1;
 94
 95
             rep(i,L1,len) a[i] = 0;
 96
             rep(i,L2,len) b[i] = 0;
 97
             Conv(a, b, len);
 98
 99
         100
         void deal()
101
             int a[100]={0};
102
             int n=P-1;
103
104
             int x=sqrt(n)+1;
105
             a[0]=1;
106
             rep(i, 2, x)
107
108
                 while (n%i==0) a[a[0]]=i, n/=i;
109
                 if (a[a[0]]) a[0]++;
110
             if (n!=1) a[a[0]++]=n;
111
             n=P;
112
113
             int t=1;
114
             while (t++)
115
                 int flag=0;
116
                 rep(i,1,a[0])
117
118
                 if (quick_mod(t,(n-1)/a[i],n)==1) flag=1;
119
                 if (!flag) break;
120
121
             cout<<t<endl;
122
123
         124
     }
```

分治 +fft 3.3

```
#include <bits/stdc++.h>
   #define INF 0x3f3f3f3f3f
3
   #define EPS 1e-8
   #define PI acos(-1.0)
   #define LL long long
```

```
#define ld double
#define ULL unsigned long long
#define rep(i,a,b) for(int i=a;i<b;i++)</pre>
                                                                   8
#define PII pair<int,int>
#define PLL pair<LL,LL>
                                                                   10
#define MP make_pair
                                                                   11
#define sf(x) scanf("%d",&x)
                                                                   12
#define sqr(x) ((x)*(x))
                                                                   13
template <class T>
                                                                   14
inline void rd(T &x) { char c = getchar(); x = 0;while(!
                                                                   15
isdigit(c)) c = getchar();
while(isdigit(c)) { x = x * 10 + c - '0'; c = getchar();
                                                                   16
#define IN freopen("in.txt", "r", stdin);
                                                                   17
#define OUT freopen("out.txt", "w", stdout);
                                                                   18
using namespace std;
                                                                   19
const int N=1e5+10;
                                                                   20
const int M=313;
                                                                   21
struct cpx
                                                                   22
                                                                   23
     ld r,i;
                                                                   24
     cpx(){}
                                                                   25
     cpx(ld r ,ld i):r(r),i(i) {}
                                                                   26
     cpx operator + (const cpx& t) const
                                                                   27
                                                                   28
         return cpx(r+t.r,i+t.i) ;
                                                                   29
                                                                   30
     cpx operator - (const cpx& t) const
                                                                   31
                                                                   32
                                                                   33
         return cpx(r-t.r,i-t.i);
                                                                   34
     cpx operator * (const cpx& t) const
                                                                   35
     {
                                                                   36
         return cpx(r*t.r-i*t.i,r*t.i+i*t.r);
                                                                   37
                                                                   38
                                                                   39
};
const double pi=acos(-1.0);
                                                                   40
int rev(int x, int n)
                                                                   41
                                                                   42
     int tmp=0:
                                                                   43
     for (int i=1;i<n;i<<=1)</pre>
                                                                   44
                                                                   45
         tmp<<=1;
                                                                   46
         if (x & i) tmp |=1;
                                                                   47
                                                                   48
     return tmp;
                                                                   49
                                                                   50
                                                                   51
void fft(cpx *a, int n, int flag)// 1 -1
                                                                   52
                                                                   53
     for (int i=0;i<n;i++)</pre>
                                                                   54
         if (i<rev(i, n))</pre>
                                                                   55
             swap(a[rev(i, n)], a[i]);
                                                                   56
     for (int i=1;i<n;i<<=1)</pre>
                                                                   57
                                                                   58
         cpx wn(cos(pi/i), flag*sin(pi/i));
                                                                   59
         for (int j=0;j<n;j+=(i<<1))</pre>
                                                                   60
                                                                   61
         {
             cpx w(1, 0);
                                                                   62
             for (int k=0;k<i;k++, w=w*wn)</pre>
                                                                   63
                                                                   64
                  cpx x=a[j+k], y=w*a[j+k+i];
                                                                   65
                  a[j+k]=x+y;
                                                                   66
                  a[j+k+i]=x-y;
                                                                   67
                                                                   68
         }
                                                                   69
                                                                   70
     if (flag==-1) for (int i=0;i<n;i++) a[i].r/=n;</pre>
                                                                   71
                                                                   72
cpx x[1<<20],y[1<<20];
                                                                   73
int a[N];
                                                                   74
LL f[N];
                                                                   75
///oi/d//cdq(int 1,int r)
                                                                   76
                                                                   77
   //cout<<l<" "<<r<endl;
                                                                   78
  if (l==r)
                                                                   79
                                                                   80
     (f[1]+=a[1])\%=M;
                                                                   81
     return;
                                                                   82
                                                                   83
   int mid=l+r>>1;
                                                                   84
  cdq(1,mid);
                                                                   85
  int L=1;
                                                                   86
```

```
87
        while (L<=r-l+1) L<<=1;
 88
 89
        rep(i, 0, L)
 90
          if (l+i<=mid) x[i]=cpx(f[i+l],0);</pre>
 91
 92
          else x[i]=cpx(0,0);
 93
          if (i+1+l<=r) y[i]=cpx(a[i+1]%M,0);</pre>
 94
          else y[i]=cpx(0,0);
 95
 96
 97
        fft(x,L,1);
 98
        fft(y,L,1);
        rep(i, 0, L)
 99
100
          x[i]=x[i]*y[i];
        fft(x,L,-1);
101
102
        rep(i,mid+1,r+1)
103
104
          f[i]+=(LL)(x[i-l-1].r+0.5);
105
          f[i]%=M;
106
        cdq(mid+1,r);
107
108
      int main()
109
110
111
        int n:
112
113
        while (\sim sf(n)\&\&n)
114
          memset(f, 0, sizeof(LL)*(n+2));
115
116
          rep(i, 0, n)
          rd(a[i+1]);
117
118
          cdq(1,n);
          cout<<f[n]<<endl;
119
120
121
          return 0:
122
     }
123
124
     2
125
     8 6
126
```

4 math

HIT

4.1 CRT

```
vector<LL> linear_mod_equation(LL a, LL b, LL n)
    //线性方程求解
3
    //ax = b \pmod{n}
 4
5
        LL x, y, d;
        vector<LL> sol;
 6
 7
        sol.clear();
8
        EXT\_GCD(a, n, d, x, y);
 9
        if( b\%d ) d = 0;
10
        else
11
             sol.push\_back(x * (b/d) % n);
12
             for (int i = 1; i < d; i++)
13
                 sol.push_back((sol[i-1] + n/d + n) % n);
14
15
16
        return sol;
17
    LL mega_mod(int n)
18
19
    //解 n 个一元线性同于方程组
    //x \equiv r \pmod{a}
20
    //求x
21
22
    {
        LL a1, a2, r1, r2, d, c, x, y, x0,s;
23
24
        bool flag = true;
25
        scanf("%lld%lld", &a1, &r1);
26
        for(int i = 1; i < n; i++)</pre>
27
             scanf("%lld%lld", &a2, &r2);
28
             if(!flag) continue;
29
30
             c = r2 - r1;
            EXT\_GCD(a1, a2, d, x, y);
31
32
            if(c%d!=0)
33
                 flag = false;
34
35
                 continue;
36
            x0 = x*c/d;
37
```

```
s = a2/d;
         x0 = (x0\%s+s)\%s;
                                                                      39
         r1=r1+x0*a1;
                                                                      40
         a1=a1*a2/d;
                                                                      41
                                                                      42
    if(flag) return r1;
                                                                      43
    else return -1LL;
                                                                      44
                                                                      45
                                                                      46
LL CRT(LL *a, LL *m, int n)
                                                                      47
//中国剩余定理
                                                                      48
//x \equiv a[i] \pmod{m[i]}
                                                                      49
//m[i] is coprime
                                                                      50
                                                                      51
    LL MM = 1, Mi, x0, y0, d, ret = 0; for(int i = 0; i < n; i++)
                                                                      52
                                                                      53
         MM *= m[i];
    for(int i = 0; i < n; i++)</pre>
                                                                      55
                                                                      56
    {
         Mi = MM/m[i];
                                                                      57
         EXT_GCD(Mi, m[i], d, x0, y0);
                                                                      58
         ret = (ret+Mi*x0*a[i]) % MM;
                                                                      59
                                                                      60
    if(ret < 0)
                                                                      61
         ret += MM;
                                                                      62
    return ret:
                                                                      63
                                                                      64
```

4.2 LL 黑科技

目录

```
LL mult( LL A, LL B, LL Mo )
{
    LL temp = ( ( LL ) ( ( db ) A*B/Mo+1e-6 ) * Mo );
    return A*B - temp;
}
```

4.3 about prime

```
const int maxn = 3010020;
                                                                 1
bool isprime[maxn];
                                                                 2
LL prime[maxn];
int doprime(LL n)
                                                                 4
//prime[] 储存质数。1—based index;
                                                                 6
    int nprime = 0;
                                                                 7
    memset(isprime, true, sizeof(isprime));
                                                                 8
    isprime[1] = false;
                                                                 9
    for(LL i = 2; i <= n; i++)
                                                                 10
                                                                 11
        if(isprime[i])
                                                                 12
        {
             prime[++nprime] = i;
                                                                 14
             for(LL j = i*i; j <= N; j+=i)</pre>
                                                                 15
                 isprime[j] = false;
                                                                 16
                                                                 17
        }
                                                                 18
    return nprime;
                                                                 19
                                                                 20
LL slow_mul(LL a, LL b, LL p)
                                                                 21
                                                                 22
    // cout << a << " " << b << endl;
                                                                 23
    LL ret = 0;
                                                                 24
    while(b) {
                                                                 25
        if(b \& 1) ret = (ret + a) \% p;
                                                                 26
        a = (a + a) \% p;
                                                                 27
        h >>= 1:
                                                                 28
                                                                 29
    return ret % p;
                                                                 30
                                                                 31
LL pow_mod(LL a, LL b, LL p)
                                                                 33
//快速幂
                                                                 34
{
                                                                 35
    LL ret = 1;
                                                                 36
    while(b) {
                                                                 37
        if(b & 1) ret = (ret*a)%p;
                                                                 38
        a = (a*a)%p;
                                                                 39
        b >>= 1;
                                                                 40
                                                                 41
    return ret%p;
                                                                 42
                                                                 43
```

44

13

18

```
HIT
 45
     //判断Mp = 2^p-1 是否为梅森素数
 46
 47
     bool lucas_lehmer(int p)
 48
 49
          if(p == 2) return true;
 50
          LL m = (1LL << p)-1LL, tmp = 4LL;
          for(int i = 0; i < p-2; i++)
 51
 52
              tmp = (slow_mul(tmp, tmp, m) - 2 + m) % m;
 53
 54
 55
          if(tmp == 0LL) return true;
 56
          return false;
     }
 57
 58
     LL witness(LL a, LL b, LL c)
 59
 60
          if(b==0)return 1;
 61
 62
          LL x, y, t=0;
 63
         while((b\&1)==0)
 64
             b>>=1, t++;
 65
          y=x=pow_mod(a,b,c);
 66
          //二次探测
         while(t--)
 67
 68
              y=slow_mul(x,x,c);
 69
              if(y==1 && x!=1 && x!=c-1)
 70
 71
                  return false;
 72
 73
 74
          return y==1;
 75
     bool miller_rabin(LL n)
 76
     //..质数为true, 非质数为false..
 77
 78
 79
          if(n==2)return true;
          if(n<2 || (n&1)==0)return false;</pre>
 80
 81
          for(int i=0;i<3;i++)</pre>
 82
              if(witness(rand()%(n-2)+2,n-1,n)!=1)
 83
                  return false;
          return true:
 84
 85
     LL ANS = INF;
 86
 87
     LL pollard_rho(LL n, LL c)
     //...随机返回一个 n 的约数...
 88
 89
     {
 90
          if(n%2==0)return 2;
 91
          LL i=1, k=2, x=rand()%n, y=x, d;
         while(1){
 92
 93
              x=(slow_mul(x,x,n)+c)%n;
 94
 95
              d=gcd(y-x,n);
              if(d==n)return n;
 97
              if(d!=n && d>1)return d;
 98
              if(i==k) y=x, k<<=1;
 99
         }
100
101
     void calc(LL n, LL c=240)
     //寻找最小的约数...
102
103
          if(n==1)return;
104
          if(miller_rabin(n)){
105
106
              ANS=min(ANS, n);
107
              return;
108
109
          LL k=n;
110
         while(k==n)k=pollard_rho(n,c--);
111
         calc(k,c),calc(n/k,c);
```

4.4 ext_{gcd}

112 }

```
LL ext_gcd(LL a, LL b, LL& x, LL& y)
 2
    // a >= 0, b > 0
         LL x1=0LL, y1=1LL, x0=1LL, y0=0LL;
 4
 5
         LL r = (a\%b + b) \% b;
        LL q = (a-r) / b;
 6
        x = 0LL, y = 1LL;
7
 8
        while(r)
9
10
             x=x0-q*x1; y=y0-q*y1;
11
             x0=x1; y0=y1;
12
             x1=x;y1=y;
```

```
a=b;b=r;
    r=a\%b:
                                                                 14
    q=(a-r)/b;
                                                                 15
                                                                 16
return b:
                                                                 17
```

$4.5 \quad inv(gcd)$

```
void EXT_GCD(LL a, LL b, LL &d, LL &x, LL &y)
//a , b 任意
                                                            2
{
    if(!b) \{d = a, x = 1, y = 0;\}
    else {EXT_GCD(b, a % b, d, y, x), y = x * (a / b);}
                                                            5
                                                            6
                                                            7
//递归求逆元
                                                            8
                                                            9
//p, x 互质
|\mathsf{LL}| inv(LL a, LL c)
                                                            10
// 用扩展欧几里得求逆元
                                                            11
// 要求 a, c 互质
                                                            12
// 如果没有逆元返回 -1
                                                            13
{
                                                            14
  LL d, x, y;
                                                            15
  EXT\_GCD(a, c, d, x, y);
                                                            16
  return d == 1 ? (x + c) % c : -1;
                                                            17
                                                            18
```

4.6 phi

```
//欧拉函数
|LL calphi(LL n)
                                                                 3
    LL res = n;
    for(LL i = 2; i*i <= n; i++)</pre>
  if(n%i==0)
    {
        res -= res/i:
                                                                 8
        while(n%i==0) n/=i;
                                                                 10
    if(n > 1)
                                                                 11
        res -= res/n;
                                                                 12
    return res;
                                                                 13
                                                                 14
                                                                 15
//欧拉函数预处理
                                                                 16
int phi[maxn];
                                                                 17
void getpthi(int n)
                                                                 18
                                                                 19
    memset(phi, 0, sizeof(phi));
                                                                 20
    phi[1] = 1;
                                                                 21
    for(int i = 2; i <= n; i++)if(!phi[i])</pre>
                                                                 22
                                                                 23
    {
         for(int j = i; j <= n; j+=i)
                                                                 24
                                                                 25
             if(!phi[j])
                                                                 26
                 phi[j] = j;
                                                                 27
             phi[j] = phi[j]/i*(i-1);
                                                                 29
         }
    }
                                                                 30
                                                                 31
```

spfa 5

Spfa 最短路 5.1

```
#include <iostream>
#include <cmath>
#include <algorithm>
#include <cstdio>
#include <queue>
#include <cstring>
#define N 1006
using namespace std;
                                                               8
struct node{
                                                               10
    int x, y, w, next;
                                                               11
}e[100006];
                                                               12
int a[N], d[N], v[N], head[N],
                                                               13
    ans, tot, n, m, x, y, w;
                                                               14
queue <int> Q;
                                                               15
```

23

24

25

39

41

44

47

49

60

66

69

目录

```
inline void addEdge(int x,int y,int w)
17
18
         tot++; e[tot].x = x; e[tot].y = y; e[tot].w = w;
19
         e[tot].next = head[x]; head[x] = tot;
20
21
    void spfa(int S)
22
    {
23
         int x, k;
         memset(d, 0x3f, sizeof(d));
24
         memset(v, 0, sizeof(v));
25
26
         Q.push(S); v[S] = 1;
         d[S] = 0;
27
         while(!Q.empty())
28
29
             x = Q.front(); Q.pop();
v[x] = 0; k = head[x];
30
31
             while(k != -1)
32
33
34
                  if(d[x]+e[k].w < d[e[k].y])
35
                  {
                      d[e[k].y] = d[x] + e[k].w;
36
37
                      if(!v[e[k].y])
38
39
                          Q.push(e[k].y);
40
                          v[e[k].y] = 1;
                      3
41
42
43
                  k = e[k].next;
             }
44
45
46
    inline void init()
47
48
49
        memset(head, -1, sizeof(head));
50
51
    int main()
52
53
         freopen("test.in", "r", stdin);
         while(scanf("%d%d", &n, &m) != EOF)
54
55
56
             init();
             for(int i = 0; i < m; i++){
57
                  scanf("%d%d%d", &x, &y, &w);
58
                  addEdge(x, y, w);
59
                 addEdge(y, x, w);
60
61
62
             int st, ed;
             scanf("%d%d", &st, &ed);
63
64
             spfa(st);
65
             int ans = d[ed]:
             if(ans >= 0x3f3f3f3f) ans = -1;
66
             printf("%d\n", ans);
67
68
        }
69
    }
```

```
build(mid+1, R, p+1);
int qry(int 1,int r,int k,int L=1,int R=n,int dep=0)
                                                                    26
    if (l>r) return 0;
                                                                   27
    if (l==r) return f[dep][1]<=k;</pre>
                                                                   28
    int cnt=lef[dep][r]-lef[dep][l-1];
                                                                   29
    int mid=L+R>>1;
                                                                   30
    if (sorted[mid]<=k)</pre>
                                                                    31
                                                                   32
         int nr=r+lef[dep][R]—lef[dep][r];
                                                                   33
         int nl=nr-(r-l-cnt);
                                                                    34
         return cnt+qry(nl,nr,k,mid+1,R,dep+1);
                                                                   35
    }else{
                                                                    36
         \quad \textbf{int} \  \, \texttt{nl=L+lef[dep][l-1]-lef[dep][L-1];}
                                                                    37
         int nr=cnt+nl-1:
                                                                   38
         return qry(nl,nr,k,L,mid,dep+1);
                                                                    40
    }
int main()
                                                                    42
                                                                    43
    int T, m, Case=0;
    scanf("%d",&T);
                                                                    45
    while (T—)
                                                                    46
         memset(lef,0,sizeof(lef));
                                                                    48
         scanf("%d%d",&n,&m);
         rep(i, 1, n+1)
                                                                    50
                                                                   51
             scanf("%d", &f[0][i]);
                                                                   52
             sorted[i]=f[0][i];
                                                                   53
                                                                   54
         sort(sorted+1, sorted+1+n);
                                                                    55
        build();
printf("Case %d:\n",++Case);
                                                                   56
                                                                   57
         rep(i, 0, m)
                                                                    58
         {
                                                                   59
             int x,y,z;
             scanf("%d%d%d",&x,&y,&z);
                                                                    61
             X++, Y++
                                                                    62
             printf("%d\n",qry(x,y,z));
                                                                    63
         }
                                                                   64
    }
                                                                    65
}
                                                                   67
1
                                                                    68
10 20
1423567890
                                                                   70
1 3 2
                                                                   71
                                                                    72
1
10 1
                                                                   73
1523647300
                                                                   74
                                                                   75
392
                                                                   76
```

划分树

HIT

划分树比 k 小数量 6.1

```
#define N 111111
 1
2
    #define M 22
    int sorted[N];
    int lef[22][N],f[22][N];
 5
    int n;
 6
    void build(int L=1,int R=n,int p=0)
7
 8
         if (L==R) return;
9
         int mid=L+R>>1;
10
         int same=mid—L+1
11
         int X=sorted[mid];
         rep(i, L, R+1)
12
13
             if (f[p][i]<X) same—;</pre>
         int lp=L,rp=mid+1;
14
15
         rep(i, L, R+1)
16
         {
             if (f[p][i]<X) f[p+1][lp++]=f[p][i];</pre>
17
18
             else if (f[p][i] == X\&\&same) f[p+1][lp++] = f[p][i],
             else f[p+1][rp++]=f[p][i];
19
20
             lef[p][i]=lef[p][L-1]+lp-L;
21
         build(L, mid, p+1);
22
```

划分树第 k 大 6.2

```
template <class T>
inline void rd(T &x) { char c = getchar(); x = 0;while(!)
                                                                 2
isdigit(c)) c = getchar();
while(isdigit(c)) { x = x * 10 + c - '0'; c = getchar();
                                                                 3
using namespace std;
#define N 111111
                                                                 5
#define M 22
                                                                 6
int f[M][N];
int lef[M][N];
                                                                 8
int sorted[N];
int n;
                                                                 10
void build(int l=0,int r=n-1,int dep=0)
                                                                 11
                                                                 12
  if (l==r) return;
                                                                 13
  int mid=(l+r)>>1;
                                                                 14
  int s_same=0;
                                                                 15
  int X=sorted[mid];
                                                                 16
  int pl=1,pr=mid+1;
                                                                 17
  rep(i, l, r+1)
                                                                 18
    if (f[dep][i]<X) s_same++;</pre>
                                                                 19
  s_sme=mid-l+1-s_sme;
                                                                 20
  rep(i, l, r+1)
                                                                 21
                                                                 22
    if (f[dep][i]<X)
                                                                 23
                                                                 24
```

```
25
           f[dep+1][pl++]=f[dep][i];
           lef[dep][i]=(i-l?lef[dep][i-1]:0)+1;
26
27
         }else if (f[dep][i]==X&&s_same)
28
           f[dep+1][pl++]=f[dep][i];
29
30
31
           lef[dep][i]=(i-1?lef[dep][i-1]:0)+1;
32
         }else
33
           f[dep+1][pr++]=f[dep][i];
34
             lef[dep][i]=(i-l?lef[dep][i-1]:0);
35
36
37
38
      build(1,mid,dep+1);
39
      build(mid+1, r, dep+1);
40
    int qry(int 1,int r,int k,int L=0,int R=n-1,int dep=0)
41
42
      int mid=(L+R)>>1;
43
44
      if (l==r) return f[dep][1];
45
46
      int cnt=lef[dep][r]-(1-L?lef[dep][1-1]:0);
      if (k>cnt)
47
48
49
         int nl,nr;
         nl=mid+1+l-L-(l-L?lef[dep][l-1]:0);
50
51
         nr=mid+1+r_L_lef[dep][r];
52
         return qry(nl,nr,k-cnt,mid+1,R,dep+1);
53
      }else
54
         int nl,nr;
55
         nl=L+(l-L?lef[dep][l-1]:0);
56
57
         nr=L+lef[dep][r]-1;
         return qry(nl,nr,k,L,mid,dep+1);
58
59
      }
60
    }
61
    int main()
62
63
           rd(n);rd(m);
64
65
           memset(f,0,sizeof(f));
           rep(i,0,n)
66
67
             rd(sorted[i]);
68
             f[0][i]=sorted[i];
69
70
           sort(sorted, sorted+n);
71
           build();
72
73
           rep(i, 0, m)
74
           {
             int s,t,k;
75
             rd(s);rd(t);rd(k);
77
             printf("%d\n",qry(s,t,k));
78
79
         return 0;
80
    }
81
82
    10 3
83
    1527543877
84
85
    2 5 3
86
    4 4 1
87
    173
88
```

7 匈牙利

HIT

7.1 二分匹配

```
using namespace std;
 2
    struct nod
 3
      int y,id;
    }a[N];
 5
    int n;
 6
    bool cmp(nod a, nod b)
8
 9
      if (a.y!=b.y) return a.y>b.y;
      return a.id<b.id;
10
11
12
    int b[N][N];
   int linker[N], ma[N];
```

```
bool use[N];
bool dfs(int u)
                                                                     15
                                                                     16
  rep(i, 1, n+1)
                                                                     17
  if (b[u][i]&&!use[i])
                                                                     18
                                                                     19
    use[i]=true;
                                                                     20
    if (linker[i]==-1||dfs(linker[i]))
                                                                     21
                                                                     22
       linker[i]=u;
                                                                     23
       ma[u]=i;
                                                                     24
       return true;
                                                                     25
                                                                     26
                                                                     27
  return false;
                                                                     28
                                                                     29
void hungary()
                                                                     30
                                                                     31
  int u;
                                                                     32
  memset(linker,-1,sizeof(linker));
                                                                     33
  rep(i,0,n)
                                                                     34
                                                                     35
    memset(use, 0, sizeof(use));
                                                                     36
    if (!dfs(a[i].id)) ma[a[i].id]=0;
                                                                     37
                                                                     38
  rep(i,1,n+1)
printf("%d%c",ma[i],i==n?'\n':' ');
                                                                     39
                                                                     40
                                                                     41
int main()
                                                                     42
                                                                     43
  scan(&n);
                                                                     44
  rep(i,0,n)
                                                                     45
                                                                     46
    scan(&a[i].y);
                                                                     47
    a[i].id=i+1;
                                                                     48
  sort(a,a+n,cmp);
                                                                     50
  rep(i, 1, n+1)
                                                                     51
                                                                     52
    int m:
                                                                     5.3
    scan(&m);
                                                                     54
    rep(j,0,m)
                                                                     55
                                                                     56
       int x;
                                                                     57
       scan(&x);
                                                                     58
       b[i][x]=1;
                                                                     59
                                                                     60
    }
                                                                     61
  hungary();
                                                                     62
                                                                     63
  return 0:
```

8 矩阵

8.1 矩阵

```
#define N 2
#define M 1000000007
LL fis[N];
struct mart
                                                                  5
    LL a[N][N];
    mart(int x){
         memset(a, 0, sizeof(a));
                                                                   8
         if (x==1)
         rep(i,0,N) a[i][i]=1;
                                                                  10
         if (x==2)
                                                                   12
         {
             a[0][0]=lala;a[0][1]=0;
                                                                  13
             a[1][0]=1;a[1][1]=1;
                                                                  15
                                                                   16
    mart operator *(mart &b)
                                                                   17
                                                                   18
                                                                   19
         mart c(0);
         rep(i, 0, N)
                                                                  20
             rep(j,0,N)
                                                                   21
             if (a[i][j])
                                                                   22
                 rep(k, 0, N)
                                                                   23
                 c.a[i][k]=(c.a[i][k]+a[i][j]*b.a[j][k])%M;
                                                                   24
         return c;
                                                                   25
    }
                                                                   26
```

64

```
27
         void show()
28
             puts("
29
                                       _");
30
             rep(i, 0, N)
31
32
                  rep(j, 0, N)
                  printf("%d ",a[i][j]);
33
                  puts("");
34
35
             puts("_
                                       _");
36
37
38
    LL pow(LL k)
39
40
    {
         if (k<0) return 1;
41
42
         mart ret(1), a(2);
         //a.show();
43
44
         while (k)
45
46
             if (k&1) ret=ret*a;
47
             a=a*a;k>>=1;
48
         // ret.show();
49
50
         LL ans=0;
51
         rep(i, 0, N)
         ans=(ans+fis[i]*ret.a[i][0]%M)%M;
52
53
         return ans;
    }
```

9 组合

9.1 组合数求 mod

```
LL exp_mod(LL a, LL b, LL p) {
1
         LL res = 1;
2
         while(b != 0) {
3
             if(b\&1) res = (res * a) % p;
5
             a = (a*a) \% p;
             b >>= 1;
6
8
         return res:
    }
9
10
    LL Comb(LL a, LL b, LL p) {
11
12
         if(a < b)
                      return 0;
13
         if(a == b) return 1;
         if(b > a - b) b = a - b;
14
15
         LL ans = 1, ca = 1, cb = 1;
16
         for(LL i = 0; i < b; ++i) {
    ca = (ca * (a - i))%p;
17
18
             cb = (cb * (b - i))%p;
19
20
21
         ans = (ca*exp\_mod(cb, p-2, p)) \% p;
         return ans;
22
    }
23
24
25
    LL Lucas(int n, int m, int p) {
26
          LL ans = 1;
27
28
          while(n&&m&&ans) {
             ans = (ans*Comb(n%p, m%p, p)) % p;
29
             n /= p;
30
31
             m /= p;
32
          return ans;
33
    }
34
```

10 网络流

10.1 MinCost_bhb

```
struct edge{
  int x, ne, c, f, w;
};

struct MinCostFlow{
  edge e[M];
  int S, T, pos, quantity, cost;
  int head[N], dis[N], pre[N], at[N];
  queue <int> q;
```

```
bool used[N];
                                                                 10
                                                                 11
  void adde(int u, int v, int c, int w){
                                                                 12
    //printf("Add edge : %d \rightarrow %d c : %d w : %d\n", u, v,
                                                                 13
    c, w);
    e[++pos] = (edge)\{v, head[u], c, 0, w\};
                                                                 14
    head[u] = pos;
                                                                 15
    e[++pos] = (edge)\{u, head[v], c, c, -w\};
                                                                 16
                                                                 17
    head[v] = pos;
                                                                 18
                                                                 19
  bool spfa(){
                                                                 20
    memset(dis, 0x3f, sizeof(dis));
                                                                 21
    memset(used, 0, sizeof(used));
                                                                 22
    used[S] = true;
                                                                 23
    while (!q.empty())
                                                                 24
      q.pop();
                                                                 25
    q.push(S);
                                                                 26
    dis[S] = 0;
                                                                 27
    while (!q.empty()){
                                                                 28
                                                                 29
      int x = q.front();
      //printf("Now : %d\n", x);
                                                                 30
      for (int i = head[x]; i; i = e[i].ne){
                                                                 31
        int y = e[i].x;
                                                                 32
        if (e[i].c > e[i].f && dis[x] + e[i].w < dis[y]){</pre>
                                                                 33
          dis[y] = dis[x] + e[i].w;
                                                                 34
          at[y] = i;
                                                                 35
          pre[y] = x;
                                                                 36
          if (!used[y]){
                                                                 37
            used[y] = true;
                                                                 38
             q.push(y);
                                                                 39
                                                                 40
        }
                                                                 41
                                                                 42
      used[x] = false;
                                                                 43
      q.pop();
                                                                 45
    //printf("Spfa : %d\n", dis[T]);
                                                                 46
    return dis[T] != INF;
                                                                 47
  }
                                                                 48
                                                                 49
  void update(){
                                                                 50
    int cut = INF;
                                                                 51
    for (int i = T; i != S; i = pre[i]){
                                                                 52
      cut = min(cut, e[at[i]].c - e[at[i]].f);
                                                                 53
                                                                 54
    //printf("Cut %d's path : %d \rightarrow ",
                                                                 55
    for (int i = T; i != S; i = pre[i]){
                                                                 56
      e[at[i]].f += cut;
                                                                 57
      e[at[i] ^1].f = cut;
                                                                 58
      //printf(" - > %d", pre[i]);
                                                                 59
    quantity += cut;
                                                                 61
    cost += cut * dis[T];
                                                                 62
    //puts("-
                                                                 63
                                                                 64
                                                                 65
  void init(int s, int t){
                                                                 66
    S = s;
                                                                 67
    T = t;
                                                                 68
    pos = 1;
                                                                 69
    quantity = cost = 0;
                                                                 70
    memset(head, 0, sizeof(head));
                                                                 71
                                                                 72
                                                                 73
  PII work(){
                                                                 74
    //puts("Starting");
                                                                 75
    while (spfa())
                                                                 76
      update();
                                                                 77
    //printf("%d %d\n", quantity, cost);
                                                                 78
    return make_pair(quantity, cost);
                                                                 79
                                                                 80
}flow;
                                                                 81
```

10.2 dinic_bhb

```
8
      edge e[M];
      int head[N], h[N], dis[N], q[N], stack[N];
9
10
      bool used[N];
11
      int pos, stop, top, S, T;
      LL flow;
12
13
14
      void init(int s, int t)
15
16
         pos = 1;
         flow = top = 0;
17
18
         S = s;
         T = t;
19
         memset(head, 0, sizeof(head));
20
21
22
      void adde(int u, int v, int c)
23
24
         e[++pos] = (edge)\{v, head[u], c, 0\};
25
26
         head[u] = pos;
27
         e[++pos] = (edge)\{u, head[v], c, c\};
28
         head[v] = pos;
29
30
      bool number()
31
32
         memset(dis, 0, sizeof(dis));
33
         memset(used, 0, sizeof(used));
34
35
         int p1, p2, x;
         used[q[p1 = p2 = 1] = S] = true;
36
37
         while (p1 <= p2)
38
           x = q[p1++];
39
           for (int i = head[x]; i; i = e[i].ne)
40
             if (e[i].c > e[i].f && !used[e[i].x])
41
42
                used[q[++p2] = e[i].x] = true;
43
                dis[e[i].x] = dis[x] + 1;
44
45
46
         if (!used[T])
47
48
           return false;
         memcpy(h, head, sizeof(head));
49
50
         return true;
51
52
53
      bool dinic(int x)
54
         if (x == T)
55
56
           int cut = INF;
57
           for (int i = 1; i <= top; ++i)</pre>
58
           cut = min(cut, e[stack[i]].c - e[stack[i]].f);
for (int i = 1; i <= top; ++i)</pre>
60
61
             e[stack[i]].f += cut;
62
             e[stack[i] ^ 1].f -= cut;
63
64
             if (e[stack[i]].c == e[stack[i]].f)
                stop = i;
65
66
67
           flow += cut;
           return true;
68
69
         for (int &i = h[x]; i; i = e[i].ne)
  if (e[i].c > e[i].f && dis[x] == dis[e[i].x] - 1)
70
71
72
73
             stack[++top] = i;
             if (dinic(e[i].x) && stop != top)
74
76
                —top;
77
                return true;
78
79
               -top;
80
         return false;
81
82
83
      LL maxflow()
84
85
         while (number())
86
           dinic(S);
87
88
         return flow;
89
90
    }net;
```