PAT 真题解析

参考代码

PAT20110828

A. World Cup Betting (20)

1. C 语言版本

```
#include <stdio.h>
int main()
{
    double p[3], t;
    int i, j, k;
    for (i = 0; i < 3; i++) {
        scanf("%lf", &p[i]);
        k = 0;
        for (j = 1; j < 3; j++) {
            scanf("%lf", &t);
            if ( t > p[i] ) {
                p[i] = t;
                k = j;
            }
        }
        switch (k) {
          case 0: printf("W "); break;
          case 0: printf("T "); break;
          case 0: printf("L "); break;
        }
    }
    printf("%.2lf\n", (p[0] * p[1] * p[2] * 0.65 - 1.0) * 2.0);
return 0;
}
```

2. Lua 语言版本

```
str = ''
tag = 'WTL'
result = 1.0
for line in io.lines() do
    index = 0
    maxv = 0
```

```
        for v in string.gmatch(line, '[^%s]+') do
            index = index + 1
            v = tonumber(v)
            if v > maxv then
                maxv = v
                maxi = index
            end
        end
        str = str .. string.sub(tag, maxi, maxi) .. ' '
        result = result * maxv
    end
    print(string.format(str .. '%.2f', result * 1.3 - 2))
```

B. The Best Rank (25)

1. C 语言版本

```
    #include <stdio.h>
    #include <math.h>
    #include <stdlib.h>

    struct country {
        int A, C, M, E;
        int rank, type,tmprank, index, id;
    } record[2000], tmpr;

    int mapindex[1000000];

    //定义结构体，分别记录 3 个分值以及平均分、排序以及索引。

    int comparA(const void *a, const void *b) {
        return ((const struct country*)a)->A < ((const struct country*)b)->A ?
1 : 0;
    }

    int comparC(const void *a, const void *b) {
        return ((const struct country*)a)->C < ((const struct country*)b)->C ?
1 : 0;
    }

    int comparM(const void *a, const void *b) {
        return ((const struct country*)a)->M < ((const struct country*)b)->M ?
1 : 0;
```

```
    }

    int comparE(const void *a, const void *b) {
        return ((const struct country*)a)->E < ((const struct country*)b)->E ?
1 : 0;
    }

    int comparInd(const void *a, const void *b) {
        return   ((const   struct   country*)a)->index   >   ((const   struct
country*)b)->index ? 1 : 0;
    }

    void mysort(int n, int (*cmp)())
    {
        int i, j;

        for ( i=1; i<n; i++ ) {
            tmpr.A = record[i].A;
            tmpr.C = record[i].C;
            tmpr.M = record[i].M;
            tmpr.E = record[i].E;
            tmpr.rank = record[i].rank;
            tmpr.type = record[i].type;
            tmpr.index = record[i].index;
            tmpr.id = record[i].id;
            for ( j=i; (j>0) && cmp(&record[j-1], &tmpr); j-- ) {
                record[j].A = record[j-1].A;
                record[j].C = record[j-1].C;
                record[j].M = record[j-1].M;
                record[j].E = record[j-1].E;
                record[j].rank = record[j-1].rank;
                record[j].type = record[j-1].type;
                record[j].index = record[j-1].index;
                record[j].id = record[j-1].id;
            }
            record[j].A = tmpr.A;
            record[j].C = tmpr.C;
            record[j].M = tmpr.M;
            record[j].E = tmpr.E;
            record[j].rank = tmpr.rank;
            record[j].type = tmpr.type;
            record[j].index = tmpr.index;
            record[j].id = tmpr.id;
        }
```

```
    }

    int main()
    {
        int n, m, p, i;

        for (i=0; i<1000000; i++)
            mapindex[i] = -1;

        scanf("%d %d", &n, &m);
        for (i=0; i<n; i++) {
            scanf("%d %d %d %d", &record[i].id, &record[i].C, &record[i].M,
&record[i].E);
            record[i].A = (floor)((float)(record[i].C + record[i].M +
record[i].E)/3.0+0.5);
            record[i].index = i;
            mapindex[record[i].id] = i;
        }

        mysort(n, comparA);

        for (i=0; i<n; i++) {
            if ( i && (record[i].A==record[i-1].A) ) p = record[i-1].rank;
            else p = i;
            record[i].rank = p;
            record[i].type = 1;
        }

        mysort(n, comparC);

        for (i=0; i<n; i++) {
            if ( i && (record[i].C==record[i-1].C) ) p = record[i-1].tmprank;
            else p = i;
            record[i].tmprank = p;
            if ( p < record[i].rank ) {
                record[i].rank = p;
                record[i].type = 2;
            }
        }

        mysort(n, comparM);

        for (i=0; i<n; i++) {
            if ( i && (record[i].M==record[i-1].M) ) p = record[i-1].tmprank;
```

```
        else p = i;
        record[i].tmprank = p;
        if ( p < record[i].rank ) {
            record[i].rank = p;
            record[i].type = 3;
        }
    }

    mysort(n, comparE);

    for (i=0; i<n; i++) {
        if ( i && (record[i].E==record[i-1].E) ) p = record[i-1].tmprank;
        else p = i;
        record[i].tmprank = p;
        if ( p < record[i].rank ) {
            record[i].rank = p;
            record[i].type = 4;
        }
    }

    mysort(n, comparInd);

    for (i=0; i<m; i++) {
        scanf("%d", &p);
        if ((p = mapindex[p]) != -1) {
            printf("%d ", record[p].rank+1);
            switch(record[p].type) {
            case 1: printf("A\n"); break;
            case 2: printf("C\n"); break;
            case 3: printf("M\n"); break;
            case 4: printf("E\n"); break;
            }
        }
        else printf("N/A\n");
    }

    return 0;
}
```

2. C++语言版本

```
#include <iostream>
#include <map>
#include <algorithm>
```

```
    using namespace std;

    const int N = 2000;

    struct Student {
        int id, best;
        char tag;
        int a[4], rank[4];

        void read() {
            cin >> id;
            for (int i = 1; i <= 3; ++i)
                cin >> a[i];
            a[0] = (a[1] + a[2] + a[3]) / 3.0;
        }

        void sort() {
            int k = 0;
            char str[10] = "ACME";
            for (int i = 1; i < 4; ++i)
                if (rank[i] < rank[k])
                    k = i;
            best = rank[k];
            tag = str[k];
        }

    };

    void sort(vector < Student > &a, int k) {
        sort(a.begin(), a.end(), [k](const Student& a, const Student& b){ return
a.a[k] > b.a[k]; });
        for (int i = 0; i < a.size(); ++i) {
            if (i == 0 || a[i].a[k] != a[i - 1].a[k])
                a[i].rank[k] = i + 1;
            else
                a[i].rank[k] = a[i - 1].rank[k];
        }
    }

    int main() {
        int n, m;
        cin >> n >> m;
        vector < Student > a(n);
        map < int, int > mp;
```

```
    for (int i = 0; i < n; ++i)
        a[i].read();
    for (int i = 0; i < 4; ++i)
        sort(a, i);
    for (int i = 0; i < n; ++i) {
        mp[ a[i].id ] = i;
        a[i].sort();
    }
    while (m--) {
        int id;
        scanf("%d", &id);
        if (mp.find(id) == mp.end())
            cout << "N/A" << endl;
        else {
            id = mp[id];
            cout << a[id].best << " " << a[id].tag << endl;
        }
    }
    return 0;
}
```

C. Battle Over Cities (25)

C++语言版本

```
#include<stdio.h>
#include<string.h>
#include<vector>
#include<iostream>
using namespace std;
#define N 1001
vector<int>G[N];

int vis[N], Time, delPoint;//这里利用时间戳优化，避免每次 vis 数组的初始化
void dfs(int u){
    vis[u] = Time;
    for(int i = 0; i < G[u].size(); i++)
    {
        int v = G[u][i];
        if(vis[v] != Time && v != delPoint)
            dfs(v);
    }
}
```

```
int main(){
    int n, m, query, u, v;
    memset(vis, 0, sizeof(vis));
    Time = 0;
    while(~scanf("%d %d %d", &n, &m, &query)){
        for(int i = 1; i <= n; i++)G[i].clear();
        while(m--)
        {
            scanf("%d %d", &u, &v);
            G[u].push_back(v);
            G[v].push_back(u);
        }
        while(query--)
        {
            scanf("%d",&u);
            Time++;
            delPoint = u;
            int cnt = 0;
            for(int i = 1; i <= n ; i++)if(vis[i] != Time && i != delPoint)
                dfs(i), cnt++;
            printf("%d\n", cnt-1);
        }
    }
    return 0;
}
```

BFS 做法：（直接替换 dfs 函数即可）

```
void bfs(int u){
    queue<int>q;
    q.push(u);
    while(!q.empty()){
        u = q.front(); q.pop();
        for(int i = 0; i < G[u].size(); i++)
        {
            int v = G[u][i];
            if(vis[v] != Time && v != delPoint)
            {
                vis[v] = Time;
                q.push(v);
            }
        }
    }
}
```

D. Waiting in Line (30)

C 语言版本

```
#include<stdio.h>

#define MAXN 20
#define MAXM 10
#define MAXK 2000

struct queue {
    int t, index;
} Q[MAXN][MAXM];

int cnt[MAXN], front[MAXN], rear[MAXN], T[MAXK], time, N, M, aval[MAXN];

void InitializeQ()
{
    int i;

    for (i=0; i<N; i++) {
        cnt[i] = 0;
        front[i] = 0;
        rear[i] = -1;
        aval[i] = 0;
    }
}

int push(int i, int t, int index)
{
    if ( cnt[i] == M )
        return 0;

    if ( ++rear[i] == M ) rear[i] = 0;
    Q[i][rear[i]].t = t;
    Q[i][rear[i]].index = index;
    cnt[i]++;
    return 1;
}

int pop( int i )
{
    if ( !cnt[i] )
```

```
        return 0;

    if ( ++front[i] == M ) front[i] = 0;
    cnt[i]--;
    return 1;
}

int Update( int dt )
{
    int i, ind, mint, min;

    ind = 0;
    time += dt;

    for ( i=0; i<N; i++) {
        if ( cnt[i] ) {
            Q[i][front[i]].t -= dt;
            if ( !Q[i][front[i]].t ) {
                T[Q[i][front[i]].index] = time;
                pop( i );
                if ( time < 540 )
                    aval[i] = 1;
                else
                    while ( cnt[i] ) {
                        T[Q[i][front[i]].index] = -1;
                        pop( i );
                    }
            }
        }
    }

    for (i=0; i<N; i++)
        if ( cnt[i] )   break;
    if ( i==N ) return -1;

    min = i;
    mint = Q[min][front[min]].t;

    for ( ++i; i<N; i++) {
        if ( cnt[i] ) {
            if ( Q[i][front[i]].t < mint ) {
                min = i;
                mint = Q[i][front[i]].t;
            }
```

```
        }
    }
    return min;
}

void Process( int K )
{
    int next, i, j, ind, min, mint;

    next = N*M;

    if ( K <= next ) {
        for (i=N; i<K; i++)
            T[i] += T[i-N];
        return;
    }

    InitializeQ();
    for (i=0; i<N; i++) {
        for (j=0; j<M; j++) {
            ind = i+j*N;
            push(i, T[ind], ind);
        }
    }
    min = 0; mint = Q[0][front[0]].t; ind = 0;
    for (i=1; i<N; i++)
        if ( Q[i][front[i]].t < mint ) {
            min = i;
            mint = Q[i][front[i]].t;
        }
    while ( (next != K) && (time < 540) ) {
        min = Update( mint );
        mint = Q[min][front[min]].t;
        for ( i=0; i<N; i++ ) {
            if ( next == K ) break;
            else if ( aval[i] ) {
                push(i, T[next], next);
                aval[i] = 0;
                next++;
            }
        }
    }

    while ( next != K ) T[next++] = -1;
```

```
    min = Update( mint );
    while ( min != -1 ) {
        mint = Q[min][front[min]].t;
        min = Update( mint );
    }
}

void Output( int nQ )
{
    int i, ind, hh, mm;

    for ( i=0; i<nQ; i++) {
        scanf("%d", &ind);
        ind--;
        if ( T[ind] == -1 )
            printf("Sorry\n");
        else {
            mm = T[ind]%60;
            hh = (T[ind]-mm)/60 + 8;
            if ( hh<10 ) printf("0");
            printf("%d:", hh);
            if ( mm<10 ) printf("0");
            printf("%d\n", mm);
        }
    }
}

int main()
{
    int i, K, nQ;

    scanf("%d %d %d %d", &N, &M, &K, &nQ);
    for (i=0; i<K; i++)
        scanf("%d", &T[i]);
    Process( K );
    Output( nQ );

    return 0;
}
```

PAT20120218

A. Have Fun with Numbers (20)

1. C 语言版本

```
#include<stdio.h>
#include<string.h>

#define MAXD 20

int N[MAXD+1];

int doubleN( int l )
{
    int i, carry = 0;

    N[0] = N[0] << 1;
    if (N[0] > 9) {
        carry = 1;
        N[0] -= 10;
    }
    for (i=1; i<l; i++) {
        N[i] = carry + (N[i]<<1);
        if (N[i] > 9) {
            carry = 1;
            N[i] -= 10;
        }
        else carry =0;
    }
    if (carry) {
        N[l] = 1;
        l++;
    }

    return l;
}

int main()
{
    char s[MAXD+1];
```

```
    int i, l, ll, cnt[10];

    for (i=0; i<10; i++) cnt[i] = 0;

    scanf("%s", s);
    l = strlen(s);
    for (i=l-1; i>=0; i--) {
        N[i] = s[l-i-1]-'0';
        cnt[N[i]] ++;
    }
    ll = doubleN(l);
    if (ll > l) printf("No\n");
    else {
        for (i=0; i<l; i++)
            cnt[N[i]] --;
        for (i=0; i<10; i++)
            if (cnt[i]) break;
        if (i<10) printf("No\n");
        else printf("Yes\n");
    }
    for (i=ll-1; i>=0; i--)
        printf("%d", N[i]);
    printf("\n");

    return 0;
}
```

2. Python 语言版本

```
def check(length, num, mp): # 判断 num 中的数字个数与 mp 表是否一一对应
    if length != len(num):
        return False
    for i in xrange(length):
        mp[num[i]] = mp.get(num[i], 0) - 1
        if mp[num[i]] < 0:
            return False
    return True

num = raw_input()
length = len(num)
mp = {}
for i in xrange(length):
    mp[num[i]] = mp.get(num[i], 0) + 1
num = str(int(num) * 2)
```

```
print 'Yes' if check(length, num, mp) else 'No'
print num
```

B. Palindromic Number (25)

1. C 语言版本

```
#include<stdio.h>
#include<string.h>
#define MAXD 100

char S1[MAXD], S[MAXD];

int IsSym( int L )
{
    int i, j;
    int flag = 1;

    i = 0; j = L-1;
    while ( i<j ) {
        if (S1[i] != S1[j]) {
            flag= 0;
            break;
        }
        else {
            i++; j--;
        }
    }
    return flag;
}

void InverseS1( int L )
{
    int i;
    for (i=0; i<L; i++)
        S[i]= S1[L-i-1];
    S[L] = '\0';
    return;
}

void Add( int L )
{
    int i, carry=0, n;
```

```
    for ( i=0; i<L; i++ ) {
        n = S1[i]-'0'+S[i]-'0'+carry;
        if (n>9) {
            carry = 1;
            n -= 10;
        }
        else carry = 0;
        S1[i] = n+'0';
    }
    if (carry)
        S1[L] = '1';
}

int main()
{
    int l, k, count = 0;

    scanf("%s %d", S1, &k);
    l = strlen(S1);
    while ( !IsSym(l) ) {
        InverseS1(l);
        Add(l);
        l = strlen(S1);
        count++;
        if (count == k)
            break;
    }
    InverseS1(l);
    printf("%s\n", S);
    printf("%d\n", count);

    return 0;
}
```

2. Python 语言版本

```
def isPalindromic(s):
    l = 0
    r = len(s) - 1
    while l <= r:
        if s[l] != s[r]:
            return False
        l += 1
        r -= 1
```

```
        return True

    v, n = input().split(' ')
    n = int(n)
    for i in range(n + 1):
        if isPalindromic(v) or i == n:
            print(v)
            print(i)
            break
        v = str(int(v) + int(v[::-1]))
```

C. PAT Ranking (25)

1. C 语言版本

```
    #include<stdio.h>
    //#include<math.h>
    #include<stdlib.h>
    #include<string.h>

    #define MAXN 100
    #define MAXK 300
    #define MAXS 30000 /* MAXN*MAXK */

    struct student {
        int score, loc, lrank, frank;
        char id[14];
    } Stu[MAXS];

    int comparS(const void *a, const void *b) {
        return ((const struct student*)a)->score < ((const struct
student*)b)->score ? 1 : 0;
    }

    int comparId(const void *a, const void *b) {
        return strcmp( ((const struct student*)a)->id, ((const struct
student*)b)->id ) > 0 ? 1 : 0;
    }

    void copystu( struct student S1, struct student *S2 )
    {
        int i;
        for (i=0; i<14; i++)
```

```
        (*S2).id[i] = S1.id[i];
    (*S2).score = S1.score;
    (*S2).loc = S1.loc;
    (*S2).lrank = S1.lrank;
    (*S2).frank = S1.frank;
}

void mysort(struct student S[], int n, int (*cmp)())
{
    int i, j;
    struct student tmp;

    for ( i=1; i<n; i++ ) {
        copystu( S[i], &tmp );
        for ( j=i; (j>0) && cmp(&S[j-1], &tmp); j-- ) {
            copystu( S[j-1], &S[j] );
        }
        copystu( tmp, &S[j] );
    }
}

void LRank(struct student S[], int n)
{
    int i, sn;

    S[0].lrank = 1;
    sn = 0;
    for (i=1; i<n; i++) {
        if (S[i].score == S[i-1].score) {
            S[i].lrank = S[i-1].lrank;
            sn++;
        }
        else {
            S[i].lrank = i+1;
            if (sn) {
                mysort(S+i-sn-1, sn+1, comparId);
                sn = 0;
            }
        }
    }
}

int Merge( int SN, struct student S[], int n )
{
```

```
        int StuP, SP, cur;

        StuP = SN-1; SP = n-1; cur = SN+n-1;
        while ( (StuP>=0) && (SP>=0) ) {
            if ( (Stu[StuP].score < S[SP].score) ||
                 ((Stu[StuP].score==S[SP].score)   &&   (strcmp(Stu[StuP].id,
S[SP].id)>0)) ) {
                Stu[cur] = Stu[StuP];  StuP--;
            }
            else {
                Stu[cur] = S[SP];  SP--;
            }
            cur--;
        }
        while ( SP>=0 ) {
            Stu[SP] = S[SP];  SP--;
        }
        return (SN+n);
    }

    int main()
    {
        int N, K, lk;
        struct student S[MAXK];
        int i, j;

        K = 0;
        scanf("%d", &N);
        for (i=0; i<N; i++) {
            scanf("%d", &lk);
            for (j=0; j<lk; j++) {
                scanf("%s", S[j].id);
                scanf("%d", &S[j].score);
                S[j].loc = i+1;
            }
            mysort(S, lk, comparS);
            LRank(S, lk);
            K = Merge( K, S, lk );
        }
        Stu[0].frank = 1;
        for (i=1; i<K; i++) {
            if (Stu[i].score == Stu[i-1].score)
                Stu[i].frank = Stu[i-1].frank;
            else  Stu[i].frank = i+1;
```

```
        }
        printf("%d\n", K);
        for (i=0; i<K; i++)
            printf("%s  %d  %d  %d\n",  Stu[i].id,  Stu[i].frank,  Stu[i].loc,
Stu[i].lrank);

        return 0;
    }
```

2. C++语言版本

```
    #include <iostream>
    #include <string>
    #include <algorithm>
    #include <vector>
    using namespace std;

    struct Student {
        Student(string registrationNumber = "", int score = 0, int locationNumber
=           0):registrationNumber(registrationNumber),           score(score),
locationNumber(locationNumber) {
            finalRank = locationRank = 0;
        }

        bool operator < (const Student &that) const {
            if (score != that.score)
                return score > that.score;
            return registrationNumber < that.registrationNumber;
        }

        string registrationNumber;
        int score;
        int finalRank;
        int locationNumber, locationRank;
    };

    int main() {
        vector < Student > students;
        int m;
        cin >> m;
        for (int i = 1; i <= m; ++i) {
            vector < Student > locals;
            int n;
            cin >> n;
```

```
        for (int j = 0; j < n; ++j) {
            string registrationNumber;
            int score;
            cin >> registrationNumber >> score;
            locals.push_back(Student(registrationNumber, score, i));
        }
        sort(locals.begin(), locals.end());
        for (int i = 0; i < locals.size(); ++i)
            if (0 == i || locals[i].score != locals[i - 1].score)
                locals[i].locationRank = i + 1;
            else
                locals[i].locationRank = locals[i - 1].locationRank;
        students.insert(students.end(), locals.begin(), locals.end());
    }
    sort(students.begin(), students.end());
    for (int i = 0; i < students.size(); ++i)
        if (0 == i || students[i].score != students[i - 1].score)
            students[i].finalRank = i + 1;
        else
            students[i].finalRank = students[i - 1].finalRank;
    printf("%u\n", students.size());
    for (vector < Student >::const_iterator it = students.cbegin(); it !=
students.cend(); ++it)
        printf("%s %d %d %d\n", it->registrationNumber.c_str(), it->finalRank,
it->locationNumber, it->locationRank);
    return 0;
}
```

D. Table Tennis (30)

```
#include <stdio.h>
#include <malloc.h>

#define MaxProc 120
#define MaxWindow 100
#define MaxPlayer 10000
// 选手结构体
struct People {
    int T;
    int P;
    int VIP;
};
// 队列结构体。VIP 选手设计了另一个队列来存储位置信息。
struct QueueRecord {
```

```
    int front;
    int rear;
    int size;
    int VIPfront;
    int VIPrear;
    int VIPsize;
    struct People *Customer;
    int *VIPCustomer;
};

typedef struct QueueRecord *Queue;

Queue CreateQueue( int MaxElements );
void AddQ( Queue Q, struct People X );
void AddVIP( Queue Q, int Position );
struct People DeleteQ( Queue Q );
struct People DeleteVIP( Queue Q );
int IsEmpty( Queue Q );
// 按入队时间顺序对所有候选选手进行排序
void mysort( Queue Q, int n)
{
    int i, j, tt, pp, tvip;

    for ( i=1; i<n; i++ ) {
        tt = Q->Customer[i].T;
        pp = Q->Customer[i].P;
        tvip = Q->Customer[i].VIP;
        for ( j=i; (j>0) && (Q->Customer[j-1].T>tt); j-- ) {
            Q->Customer[j].T = Q->Customer[j-1].T;
            Q->Customer[j].P = Q->Customer[j-1].P;
            Q->Customer[j].VIP = Q->Customer[j-1].VIP;
        }
        Q->Customer[j].T = tt;
        Q->Customer[j].P = pp;
        Q->Customer[j].VIP = tvip;
    }
}
// 读入数据并创建候选队列
Queue CreateQueue( int MaxElements )
{
    Queue Q;
    struct People X;
    int i, hh, mm, ss;
```

```
    Q = malloc( sizeof( struct QueueRecord ) );
    Q->Customer = malloc( sizeof( struct People ) * MaxElements );
    Q->VIPCustomer = malloc( sizeof(int) * MaxElements );
    Q->size = 0;
    Q->front = 0;
    Q->rear = -1;
    Q->VIPsize = 0;
    Q->VIPfront = 0;
    Q->VIPrear = -1;

    for ( i=0; i<MaxElements; i++ ) {
        scanf("%d:%d:%d %d %d", &hh, &mm, &ss, &X.P, &X.VIP);
        X.T = ss + 60*( mm + 60*hh );
        if (X.P>MaxProc) X.P = MaxProc;
        X.P *= 60;
        AddQ( Q, X );
    }
    mysort( Q, Q->size );
    for ( i=0; i<MaxElements; i++ )
        if ( Q->Customer[i].VIP ) AddVIP( Q, i );

    return Q;
}
// 将新的普通候选人添加至队列
void AddQ( Queue Q, struct People X )
{
    Q->rear++;
    Q->Customer[Q->rear].T = X.T;
    Q->Customer[Q->rear].P = X.P;
    Q->Customer[Q->rear].VIP = X.VIP;
    Q->size++;
}
// 将新的 VIP 候选人添加至队列
void AddVIP( Queue Q, int Position )
{
    Q->VIPrear++;
    Q->VIPCustomer[Q->VIPrear] = Position;
    Q->VIPsize++;
}
// 从当前队列中取出第一个选手，T 为-1 表示队列已空
struct People DeleteQ( Queue Q )
{
    struct People X;
```

```
        while ( Q->Customer[Q->front].VIP == -1 ) {
            Q->front++;
            Q->size--;
        }
        if ( IsEmpty(Q) ) {
            X.T = -1;
            return X;
        }
        if ( Q->Customer[Q->front].VIP == 1 )
            X = DeleteVIP(Q);
        else {
            X.T = Q->Customer[Q->front].T;
            X.P = Q->Customer[Q->front].P;
            X.VIP = Q->Customer[Q->front].VIP;
        }
        Q->front++;
        Q->size--;

        return X;
    }
    // 从当前队列中取出第一个 VIP 选手
    struct People DeleteVIP( Queue Q )
    {
        struct People X;
        int Position;

        if ( Q->VIPsize ) {
            Position = Q->VIPCustomer[Q->VIPfront];
            Q->VIPfront++;
            Q->VIPsize--;
            Q->Customer[Position].VIP = -1; // 将选手的 VIP 标记置为-1 表示更新为无
效数据
            X.T = Q->Customer[Position].T;
            X.P = Q->Customer[Position].P;
            X.VIP = Q->Customer[Position].VIP;
        }
        else
            X = DeleteQ(Q);

        return X;
    }
    // 判断队列是否为空
    int IsEmpty( Queue Q )
    {
```

```
    return ( Q->size == 0 );
}
// 判断当前时间是否有 VIP 选手到达
int IsVipHere( Queue Q, int CurrentTime )
{
    int Position;

    if ( Q->VIPsize ) {
        Position = Q->VIPCustomer[Q->VIPfront];
        if (CurrentTime >= Q->Customer[Position].T)
            return 1;
    }
    return 0;
}
// 从当前的所有 table 中找出所需剩余时间最少的
int FindNextWindow( int W[], int K, int *WaitTime )
{
    int WinAvail, MinW = MaxProc*60+1;
    int i;

    for ( i=0; i<K; i++ )
        if ( W[i] < MinW ) {
            MinW = W[i];  WinAvail = i;
        }
    *WaitTime = MinW;
    for ( i=0; i<K; i++ )
        W[i] -= MinW;

    return WinAvail;
}
// 按时分秒格式输出时间
void PrintTime( int T )
{
    int hh, mm, ss;

    hh = T/3600;
    mm = (T-hh*3600)/60;
    ss = (T-hh*3600-mm*60);
    if (hh<10) printf("0");
    printf("%d:", hh);
    if (mm<10) printf("0");
    printf("%d:", mm);
    if (ss<10) printf("0");
    printf("%d", ss);
```

```
    }

    void QueueingAtBank( Queue Q, int N )
    {
        struct People Next;
        int CurrentTime, Window[MaxWindow], Count[MaxWindow], WaitTime;
        int i, j, K, nvip, VIPWindow[MaxWindow], WinAvail;

        scanf("%d %d", &K, &nvip);

        for ( i=0; i<K; i++ ) {
            Window[i] = 0;
            VIPWindow[i] = 0;
            Count[i] = 0;
        }
        for (i=0; i<nvip; i++) {
            scanf("%d", &j);
            VIPWindow[j-1] = 1;
        }

        CurrentTime = 28800;
        while ( !IsEmpty(Q) ) {
            WinAvail = FindNextWindow( Window, K, &WaitTime );
            CurrentTime += WaitTime;
            if ( VIPWindow[WinAvail] && (IsVipHere(Q, CurrentTime)) ) // 当前的
VIP table 可用并且有 VIP 候选人到达
                Next = DeleteVIP(Q);
            else
                Next = DeleteQ(Q);
            if ( Next.T < 0 ) break;
            if ( Next.VIP ) {
                for (i=0; i<K; i++)
                    if ((!Window[i]) && (VIPWindow[i])) break;
                if (i<K) WinAvail = i;
            }
            if ( CurrentTime < Next.T ) {
                WaitTime = Next.T - CurrentTime;
                for ( j=0; j<K; j++ ) {
                    Window[j] -= WaitTime;
                    if ( Window[j] < 0 )  Window[j] = 0;
                }
                CurrentTime = Next.T;
            }
            if (CurrentTime > 75599) break;
```

```
            PrintTime(Next.T); printf(" "); PrintTime(CurrentTime);
            printf("                                                    %d\n",
(int)(((double)CurrentTime-(double)Next.T)/60.0+0.5));
            Window[WinAvail] = Next.P;
            Count[WinAvail] ++;
        }
        printf("%d", Count[0]);
        for ( i=1; i<K; i++ )
            printf(" %d", Count[i]);
        printf("\n");
    }

    int main()
    {
        int N;
        Queue Q;

        scanf("%d", &N);
        Q = CreateQueue( N );
        QueueingAtBank( Q, N );

        return 0;
    }
```

PAT20120825

A. Be Unique (20)

```
#include <stdio.h>

#define MAXN 100000
#define MAXkey 10000

struct Key {
    int cnt;
    int no;
} K[MAXkey];

int main()
{
    int i, n, m;
    int ind = MAXN;

    for (i=0; i<MAXkey; i++) {
        K[i].cnt = 0;
        K[i].no = -1;
    }
    scanf("%d", &n);
    for (i=0; i<n; i++) {
        scanf("%d", &m);
        K[m-1].cnt++;
        K[m-1].no = i;
    }
    for (i=0; i<MAXkey; i++) {
        if ((K[i].cnt == 1) && (K[i].no < ind)) {
            ind = K[i].no;
            m = i+1;
        }
    }
    if (ind == MAXN)
        printf("None\n");
    else
        printf("%d\n", m);
    return 0;
}
```

B. Longest Symmetric String (25)

1. C 语言版本

```
#include <stdio.h>

#define MAXL 1000

char S[MAXL];

int IsSym( int begin, int end)
{
    int i, j, mid;

    mid = begin + ((end va- begin + 1) >> 1);
    j = end;
    for (i=begin; i<mid; i++)
        if (S[i] != S[j--]) break;
    if (i<mid) return 0;
    else return 1;
}

int main()
{
    int L, maxl, i, j;

    L = 0; maxl = 1;
    scanf("%c", &S[L]);
    while (S[L] != '\n') {
        L++;
        scanf("%c", &S[L]);
    }
    for (i=0; i<L; i++) {
        if ((L-i) < maxl) break;
        for (j=L-1; j>i; j--)
            if ( (S[j]==S[i]) && IsSym(i, j) && ((j-i+1)>maxl) ) {
                maxl = j-i+1; break;
            }
    }
    printf("%d\n", maxl);

    return 0;
}
```

2.C++语言版本

```
#include<stdio.h>
#include<string.h>
#define N 1010
inline int min(int a,int b){return a>b?b:a;}
inline int max(int a,int b){return a>b?a:b;}
//dp[i] 表示以 i 点为中心向右最大回文长度
int dp[N<<1];
char P[N<<1], T[N];
//在每一个字符间插入# 这样得到的回文串长度一定是奇数（包含#)

int Have_P(){
    int j, len = strlen(T);
    j = 0;
    P[j++] = '$';
    P[j++] = '#';
    for(int i = 0; i < len; i++)
        P[j++] = T[i], P[j++] = '#';
    P[j] = '\0';
    return j;
}

void Manacher(int Plen){
    int mx = 0, id = 0;
    dp[0] = 0;
    for(int i = 1; i < Plen; i++)
    {
        dp[i] = mx>i? min(dp[2*id-i], mx-i) : 1;
        while(P[i + dp[i]] == P[i - dp[i]])dp[i]++;
        if(i + dp[i] > mx)
        {
            mx = dp[i] + i;
            id = i;
        }
    }
}

int main(){
    while(gets(T)){
        int len, ans = 0;
        Manacher(len = Have_P());
        for(int i = 0; i < len; i++)
```

```
            ans = max(ans, dp[i]-1);
        printf("%d\n",ans);
    }
    return 0;
}
```

3. Java 语言版本

```
import java.util.Scanner;

public class Main {
    private static Scanner in = new Scanner(System.in);

    public static void main(String[] args) {
        String s = in.nextLine();
        int ans = 0;
        for (int i = 0; i < s.length(); ++i)
            for (int j = i + ans; j < s.length(); ++j)
                if (isSymmetric(s, i, j))
                    ans = j - i + 1;
        System.out.println(ans);
    }

    public static boolean isSymmetric(String s, int l, int r) {
        while (l < r) {
            if (s.charAt(l) != s.charAt(r))
                return false;
            ++l;
            --r;
        }
        return true;
    }

}
```

C. Course List for Student (25)

```
#include<stdio.h>
#include<string.h>
#include <stdlib.h>

#define MaxK 2500
#define MaxS 200
```

```
#define MaxName 4
#define MaxHashName 26426
#define MaxD 10

typedef  struct  ListNode  *List;
struct ListNode {
    int cNo;
    List Next;
};

struct StudentNode {
    int cnt;
    List crs, tail;
} student[MaxHashName][MaxD];

struct CourseNode{
    int cnt;
    int stu[MaxS];
} course[MaxK];

void Initialize()
{
    int i, j;

    for (i=0; i<MaxHashName; i++)
        for (j=0; j<MaxD; j++) {
            student[i][j].cnt = 0;
            student[i][j].crs = student[i][j].tail = NULL;
        }
}

List NewNode( int cn )
{
    List temp;

    temp = (List)malloc(sizeof(struct ListNode));
    temp->cNo = cn;
    temp->Next = NULL;

    return temp;
}

int NameHash( char name[] )
{
```

```
        int i, j;

        i = name[0] - 'A';
        for (j=1; j<3; j++)
            i = (i<<5) + name[j] - 'A';

        return i;
    }

    void ReadInsert( int N, int K )
    {
        int i, j, cn, sn1, sn2;
        char name[MaxName+1];
        List tmp;

        for (i=0; i<K; i++){
            scanf("%d", &cn);
            scanf("%d", &course[cn-1].cnt);
            for (j=course[cn-1].cnt-1; j>=0; j--) {
                scanf("%s", name);
                course[cn-1].stu[j] = (NameHash(name)<<5) + name[3] - '0';
            }
        }
        for (i=0; i<K; i++){
            for (j=course[i].cnt-1; j>=0; j--) {
                sn2 = course[i].stu[j] % 32;
                sn1 = course[i].stu[j] >> 5;
                student[sn1][sn2].cnt ++;
                tmp = NewNode(i+1);
                if (student[sn1][sn2].cnt==1) {
                    student[sn1][sn2].crs = student[sn1][sn2].tail = tmp;
                }
                else {
                    student[sn1][sn2].tail->Next = tmp; student[sn1][sn2].tail
= tmp;
                }
            }
        }
    }

    void Output ( int N )
    {
        int i, j, sn1, sn2;
        char name[MaxName+1];
```

```
    for (i=0; i<N; i++) {
        scanf("%s", name);
        printf("%s", name);
        sn1 = NameHash(name);
        sn2 = name[3] - '0';
        printf(" %d", student[sn1][sn2].cnt);
        student[sn1][sn2].tail = student[sn1][sn2].crs;
        for (j = student[sn1][sn2].cnt; j>0; j--) {
            printf(" %d", student[sn1][sn2].tail->cNo);
            student[sn1][sn2].tail = student[sn1][sn2].tail->Next;
        }
        printf("\n");
    }
}

int main()
{
    int N, K;

    scanf("%d %d", &N, &K);
    Initialize();
    ReadInsert(N, K);
    Output(N);
    return 0;
}
```

D. Recover the Smallest Number (30)

```
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>
using namespace std;

bool cmp(const string &a, const string &b) {
    return a + b < b + a;
}

int main() {
    vector < string > v;
    int n;
    string s;
    cin >> n;
```

```
    for (int i = 0; i < n; ++i) {
        cin >> s;
        v.push_back(s);
    }
    sort(v.begin(), v.end(), cmp);
    string result = "";
    for (vector < string >::iterator it = v.begin(); it != v.end(); ++it)
        result += *it;
    int pos = 0;
    for (; pos + 1 < result.size() && result[pos] == '0'; ++pos);
    cout << result.substr(pos) << endl;
    return 0;
}
```

PAT20121216

A. Shortest Distance (25)

```
#include <stdio.h>

#define MAXN 100000

int main()
{
    int N, M, i, j, k;
    int D[MAXN+1], Dist;
//数组D表示距离的前缀和,即D[i] 表示1-i点的距离和,则对于区间[i,j] 就可用D[j]-D[i]
表示。

    Dist = 0;
    scanf("%d", &N);
    for (i=0; i<N; i++) {
        D[i] = Dist;
        scanf("%d", &k);
        Dist += k;
    }
    D[N] = Dist;
    scanf("%d", &M);
    for (k=0; k<M; k++){
        scanf("%d %d", &i, &j);
        Dist = D[j-1] - D[i-1];
        if (i>j)  Dist = -Dist;
        if (Dist > D[N]/2) Dist = D[N] - Dist;
        printf("%d\n", Dist);
    }
    return 0;
}
```

B. Student List for Course (25)

```
#include<stdio.h>
#include<string.h>
#include <stdlib.h>

#define MaxName 4
```

```
    #define MaxC 20

    typedef  struct  ListNode  *List;
    struct ListNode {
        char Name[MaxName+1];
        List Next;
    };

    struct StudentNode {
        char Name[MaxName+1];
        int nC;
        int C[MaxC];
    } *Student;

    struct CourseNode {
        int Counter;
        List Ptr;
    } *Course;

    int CmpName(const void *a, const void *b) {
      return  strcmp(((const  struct  StudentNode*)a)->Name,  ((const  struct
StudentNode*)b)->Name);
    }

    void Read_and_Sort( int *N, int *K )
    {
        int i, j;

        scanf("%d %d\n", N, K);
        Student = malloc( sizeof( struct StudentNode ) * (*N) );
        Course = malloc( sizeof( struct CourseNode ) * (*K) );
        for (i=0; i<(*K); i++) {
            Course[i].Counter = 0;
            Course[i].Ptr = NULL;
        }
        for (i=0; i<(*N); i++) {
            scanf("%s %d", Student[i].Name, &Student[i].nC);
            for (j=0; j<Student[i].nC; j++)
                scanf("%d", &Student[i].C[j]);
        }

        qsort(Student, (*N), sizeof(struct StudentNode), CmpName);
    }
```

```
List NewNode( char *name )
{
    List temp;

    temp = (List)malloc(sizeof(struct ListNode));
    strcpy(temp->Name, name);
    temp->Next = NULL;

    return temp;
}

void InsertCourse( int N, int K )
{
    List Node;
    int i, j, CourseIndex;

    for (i=N-1; i>=0; i--)
        for (j=Student[i].nC-1; j>=0; j--) {
            CourseIndex = Student[i].C[j]-1;
            Node = NewNode(Student[i].Name);
            Node->Next = Course[CourseIndex].Ptr;
            Course[CourseIndex].Ptr = Node;
            Course[CourseIndex].Counter ++;
        }
}

void Output( int K )
{
    List Ptr;
    int i;

    for (i=0; i<K; i++) {
        printf("%d %d\n", i+1, Course[i].Counter);
        for (Ptr = Course[i].Ptr; Ptr; Ptr = Ptr->Next)
            printf("%s\n", Ptr->Name);
    }
}

int main()
{
    int N, K;

    Read_and_Sort( &N, &K );
    InsertCourse( N, K );
```

```
    Output( K );

    return 0;
}
```

C. Find Coins (25)

```
#include <stdio.h>
#include <stdlib.h>

#define MAXN 10, 0000
#define MAXV 501

int main()
{
    int N, M, C[MAXN], V[MAXV], V1;
    int i, t1;

    for (i=0; i<MAXV; i++)
        V[i] = 0;
    scanf("%d %d", &N, &M);
    for (i=0; i<N; i++) {
        scanf("%d", &C[i]);
        V[C[i]]++;
    }
    V1 = MAXV;
    for (i=0; i<N; i++) {
        if (V[C[i]]) {
            if ((C[i]<M) && ((M-C[i])<MAXV) && (V[M-C[i]])) {
                if ((C[i]+C[i])==M) {
                    if (V[C[i]]>1)
                        t1 = C[i];
                    else t1 = MAXV;
                }
                else
                    t1 = C[i]<(M-C[i])? C[i]: (M-C[i]);
                if (t1 < V1) V1 = t1;
            }
            V[C[i]] = 0;
        }
    }
    if (V1<MAXV)
        printf("%d %d\n", V1, M-V1);
    else
```

```
        printf("No Solution\n");

    return 0;
}
```

D. Counting Ones (30)

```
#include <stdio.h>

int main()
{
    int num, r, t, D[10], Np[10], N[10], sum;
    int i, j, Kd;

    scanf("%d", &num);
    Kd = 0;
    r = num;
    t = 10;
    while (r) {
        D[Kd] = r%10; Np[Kd] = num%t;
        t *= 10; r /= 10;
        Kd++;
    }
    N[0] = (D[0]>0)? 1:0;
    t = 1;
    for (i=1; i<Kd; i++) {
        for (j=0; j<i; j++)
            N[j] += D[i]*t;
        N[i] = (D[i]==1)? (Np[i-1]+1) : ((D[i])? (t*10):0);
        t *= 10;
    }
    sum = 0;
    for (i=0; i<Kd; i++) sum += N[i];
    printf("%d\n", sum);

    return 0;
}
```

PAT20130310

A. String Subtraction (20)

C++语言代码

```
#include<stdio.h>
#include<string.h>
#define N 10001
char s[N], Del[N];
bool del[128];

int main(){
    while(gets(s)){
        gets(Del);
        memset(del, 0, sizeof(del));
        for(int i = 0; Del[i]; i++)
            del[ Del[i] ] = true;

        for(int i = 0; s[i]; i++)
            if(!del[s[i]])printf("%c", s[i]);
        printf("\n");
    }
    return 0;
}
```

B. Pop Sequence (25)

```
#include <stdio.h>

#define MAXS 1000

int S[MAXS], top;
int O[MAXS];

void InitS()
{
    top = -1;
}

int IsFull。( int M )
```

```
{
    return (top == (M-1));
}

int IsEmpty()
{
    return (top == -1);
}

void Push( int Elm )
{
    S[++top] = Elm;
}

int Pop()
{
    int Elm = S[top--];
    return Elm;
}

int Check( int N, int M )
{
    int i, j;

    j = 0;
    for (i=1; i<=N; i++) {
        if (!IsFull(M))
            Push(i);
        else break;
        while ((!IsEmpty()) && (S[top]==O[j])) {
            Pop(); j++;
        }
    }
    return ((i>N) && IsEmpty());
}

int main()
{
    int M, N, K;
    int i, j;

    scanf("%d %d %d", &M, &N, &K);
    for (i=0; i<K; i++) {
        for (j=0; j<N; j++) scanf("%d", &O[j]);
```

```
            InitS();
            if (Check(N, M)) printf("YES\n");
            else printf("NO\n");
        }

        return 0;
    }
```

C. Linked List Sorting (25)

```
    #include <stdio.h>
    #include <stdlib.h>

    #define MAXN 100000

    struct node {
        int key, addr, next;
    } List[MAXN],,  MapL[MAXN];

    int comparK(const void *a, const void *b)
    {
        return (((const struct node*)a)->key > ((const struct node*)b)->key)?
1:-1;
    }

    void PrintS( int k )
    {
        if (k<10000) printf("0");
        if (k<1000) printf("0");
        if (k<100) printf("0");
        if (k<10) printf("0");
        printf("%d\n", k);
        if (k<10000) printf("0");
        if (k<1000) printf("0");
        if (k<100) printf("0");
        if (k<10) printf("0");
        printf("%d ", k);
    }

    int main()
    {
        int n, i, ad, head;
        scanf("%d %d", &n, &head);
        if (head == -1) {printf("0 -1\n"); return 0;}
```

```
    for (i=0; i<n; i++) {
        scanf("%d", &ad);
        MapL[ad].addr = ad;
        scanf("%d %d", &MapL[ad].key, &MapL[ad].next);
    }
    n = 0;
    List[n] = MapL[head];
    while(List[n].next != -1) {
        List[n+1] = MapL[List[n].next];
        n++;
    }
    n++;
    qsort(List, n, sizeof(struct node), comparK);
    printf("%d ", n);
    for (i=0; i<n; i++) {
        PrintS(List[i].addr);
        printf("%d ", List[i].key);
    }
    printf("-1\n");

    return 0;
}
```

D. Path of Equal Weight (30)

```
#include <stdio.h>
#include <malloc.h>

#define MAXN 100

typedef  struct  TreeNode  *Tree;
struct TreeNode {
    Tree Child;
    int data;
    Tree Sibling;
};

Tree T[MAXN];
int N, M, S;
int path[MAXN], p, cnt;

Tree new_node( int data )
{   /* create a new node */
    Tree temp;
```

```
    temp = (Tree)malloc(sizeof(struct TreeNode));
    temp->Child = temp->Sibling = NULL;
    temp->data= data;
    return temp;
}

void CreateTree()
{
    int i, j, data, id1, id2;
    Tree temp;

    for (i=0; i<N; i++) {
        scanf("%d", &data);
        T[i] = new_node(data);
    }
    for (i=0; i<M; i++) {
        scanf("%d %d %d", &id1, &data, &id2);
        T[id1]->Child = T[id2]; temp = T[id2];
        for (j=1; j<data; j++){
            scanf("%d", &id2);
            if (T[id2]->data >= temp->data) {
                T[id2]->Sibling = temp;
                T[id1]->Child = T[id2];
            }
            else {
                while (temp->Sibling)
                    if (T[id2]->data >= temp->Sibling->data) break;
                    else temp = temp->Sibling;
                T[id2]->Sibling = temp->Sibling;
                temp->Sibling = T[id2];
            }
            temp = T[id1]->Child;
        }
    }
}

void OutputPath()
{
    int i;

    printf("%d", path[0]);
    for (i=1; i<p; i++)
        printf(" %d", path[i]);
```

```
    printf("\n");
}

void Visit(Tree T)
{
    Tree temp;

    cnt += T->data;
    path[p++] = T->data;
    if ((cnt == S) && (!T->Child))
        OutputPath();
    else if ((cnt < S) && (T->Child)) {
        for (temp=T->Child; temp; temp = temp->Sibling)
            Visit( temp );
    }
    p--;
    cnt -= T->data;
}

int main()
{
    scanf("%d %d %d", &N, &M, &S);
    CreateTree();
    cnt = p = 0;
    Visit( T[0] );

    return 0;
}
```

PAT20130830A

A. Dating (20)

```
#include <stdio.h>

int main()
{
    char s1[61], s2[61];
    int d, h, m, i;

    scanf("%s\n%s", s1, s2);
    i = 0;
    d = h = m = -1;
    while (s1[i]!='\0' && s2[i]!='\0') {
        if (s1[i]==s2[i]) {
            if (d<0) {
                switch (d=s1[i]-'A') {
                case 0: printf("MON "); break;
                case 1: printf("TUE "); break;
                case 2: printf("WED "); break;
                case 3: printf("THU "); break;
                case 4: printf("FRI "); break;
                case 5: printf("SAT "); break;
                case 6: printf("SUN "); break;
                default: d=-1; break;
                }
            }
            else {
                h = s1[i]-'0';
                if (h>=0 && h<10) printf("0%d:", h);
                else {
                    h = s1[i]-'A';
                    if (h>=0 && h<14) printf("%d:", h+10);
                    else h = -1;
                }
            }
        }
        if (h<0) i++;
        else break;
    }
    scanf("%s\n%s", s1, s2);
```

```
        i = 0;
        while (s1[i]!='\0' && s2[i]!='\0') {
            if (s1[i]==s2[i]) {
                m = s1[i]-'a';
                if (m>=0 && m<26) m = i;
                else {
                    m = s1[i]-'A';
                    if (m>=0 && m<26) m = i;
                    else m=-1;
                }
            }
            if (m<0) i++;
            else {
                if (m<10) printf("0");
                printf("%d\n", m);
                break;
            }
        }

        return 0;
    }
```

B. Talent and Virtue (25)

```
    #include <stdio.h>
    #include <stdlib.h>

    #define MAXN 100000

    struct Node {
        int id;
        int v, t, g;
    } S[MAXN];

    int CompareG(const void *a, const void *b)
    {
        int k;
        k = ((const struct Node*)a)->g < ((const struct Node*)b)->g ? 1 : 0;
        if (!k) {
            k = ((const struct Node*)a)->g > ((const struct Node*)b)->g ? -1 :
0;
            if (!k) {
                k = ((const struct Node*)a)->v < ((const struct Node*)b)->v ? 1 :
0;
```

```
                if (!k) {
                    k = ((const struct Node*)a)->v > ((const struct Node*)b)->v ?
-1 : 0;
                    if (!k)
                        k = ((const struct Node*)a)->id > ((const struct
Node*)b)->id? 1 : -1;
                }
            }
        }
        return k;
    }

    int main()
    {
        int n, i, j;
        int L, H, cnt;
        int *SS[4], cnts[4];

        cnt = 0;
        scanf("%d %d %d", &n, &L, &H);
        for (i=j=0; i<n; i++) {
            scanf("%d %d %d", &S[j].id, &S[j].v, &S[j].t);
            if (!(S[j].v<L || S[j].t<L)) {
                S[j].g = S[j].v + S[j].t;
                j++;
            }
        }
        n = j;
        if (!n) {
            printf("0\n");
        }
        else {
            printf("%d\n", n);
            for (i=0; i<4; i++) {
                SS[i] = (int *)malloc(sizeof(int) * n);
                cnts[i] = 0;
            }
            qsort(S, n, sizeof(struct Node), CompareG);
            for (i=0; i<n; i++) {
                if (!(S[i].v<H || S[i].t<H))
                    SS[0][cnts[0]++] = i;
                else if (S[i].t<H && !(S[i].v<H))
                    SS[1][cnts[1]++] = i;
                else if (S[i].v<H && S[i].t<H && !(S[i].v<S[i].t))
```

```
                    SS[2][cnts[2]++] = i;
                else SS[3][cnts[3]++] = i;
            }
            for (i=0; i<4; i++)
                for (j=0; j<cnts[i]; j++)
                    printf("%d   %d   %d\n",   S[SS[i][j]].id,   S[SS[i][j]].v,
S[SS[i][j]].t);
        }
        return 0;
    }
```

C. Set Similarity (25)

1、C++语言代码

```
    #include<stdio.h>
    #include<stdlib.h>

    #define MAXS 50
    #define MAXM 1000
    int S[MAXS][MAXM], cnt[MAXS];

    int Comparen(const void *a, const void *b)
    {
        if ((*(const int*)a) < (*(const int*)b)) return -1;
        else return 1;
    }

    int Find (int X, int sn)
    {
        int m, l, r, t;
        l = 0;
        r = cnt[sn-1]-1;
        while (l<=r) {
            m = (l+r)/2;
            t = S[sn-1][m] - X;
            if (t<0)
                l = m+1;
            else if (t>0)
                r = m-1;
            else
                return 1;
        }
        return 0;
```

```
    }

    int main()
    {
        int N, M, K, s1, s2, Nc, i, j;

        scanf("%d", &N);
        for (i=0; i<N; i++) {
            scanf("%d", &M);
            for (j=0; j<M; j++)
                scanf("%d", &S[i][j]);
            qsort(S[i], M, sizeof(int), Comparen);
            cnt[i] = 1;
            for (j=1; j<M; j++) {
                if (S[i][cnt[i]-1] != S[i][j])
                    S[i][cnt[i]++] = S[i][j];
            }
        }
        scanf("%d", &K);
        for (i=0; i<K; i++) {
            scanf("%d %d", &s1, &s2);
            if (cnt[s1-1] > cnt[s2-1]) {
                j=s1; s1=s2; s2=j;
            }
            Nc = 0;
            for (j=cnt[s1-1]-1; j>=0; j--)
                if (Find(S[s1-1][j], s2)) Nc++;
            printf("%.1f%c\n",
(double)Nc*100.0/(double)(cnt[s1-1]+cnt[s2-1]-Nc), '%');
        }

        return 0;
    }
```

2. C++语言(STL 做法)

```
    #include<stdio.h>
    #include<set>
    using namespace std;
    set<int>myset[51];
    set<int>::iterator p;
    int main(){
        int n, m, query, u, v;
        while(~scanf("%d", &n)){
            for(int i = 1; i <= n; i++)
```

```
            {
                myset[i].clear();
                scanf("%d",&m);
                while(m--){
                    scanf("%d",&u);
                    myset[i].insert(u);
                }
            }
            scanf("%d",&query);
            while(query--)
            {
                scanf("%d %d", &u, &v);
                int com = 0;
                for(p = myset[u].begin(); p != myset[u].end(); p++)
                    if(myset[v].find(*p) != myset[v].end())com++;
                printf("%.1lf%%\n",            (double)(com*100)            /
(double)(myset[u].size()+myset[v].size()-com));
            }
        }
        return 0;
    }
```

D. Complete Binary Search Tree (30)

```
    #include <stdio.h>
    #include <math.h>
    #include <stdlib.h>

    #define MAXN 1000
    int A[MAXN], B[MAXN];

    int comparK(const void *a, const void *b)
    {
        return (*((const int*) a) > *((const int*) b))? 1:-1;
    }

    void Solve(int left, int right, int root)
    {
        int n, x, h, t, L;
        n = right - left + 1;
        if (!n) return;
        h = (int)(log((double)n+1.0)/log(2.0));
        t = 1<<h;
        x = n + 1 - t;
```

```
    t = t>>1;
    if (x>t) L = t;
    else L = x;
    L += t - 1;
    B[root] = A[left + L];
    Solve(left, left+L-1, (root<<1)+1);
    Solve(left+L+1, right, (root<<1)+2);
    return;
}

int main()
{
    int n, i;

    scanf("%d", &n);
    for (i=0; i<n; i++)
        scanf("%d", &A[i]);
    qsort(A, n, sizeof(int), comparK);
    Solve(0, n-1, 0);
    printf("%d", B[0]);
    for (i=1; i<n; i++)
        printf(" %d", B[i]);
    printf("\n");
    return 0;
}
```

PAT20130830B

A. A+B 和 C (15)

```
#include <stdio.h>

int main(){
    int i=0,n=0,ret=0;
    double a, b, c;
    scanf("%d",&n);
    for(i=0;i<n;i++){
        scanf("%lf%lf%lf",&a,&b,&c);
        ret = (a+b)>c? 1:0;
        printf("Case #%d: ",i+1);
        if (ret) printf("true\n");
        else printf("false\n");
    }
    return 0;
}
```

B. 数字分类 (20)

```
#include <stdio.h>

int main()
{
    int n, i, k, cnt2, cnt3, cnt4;
    int s1, s2, s4, a5, one;

    scanf("%d", &n);
    cnt2=cnt3=cnt4=0;
    s1=s2=s4=0;
    a5=0;
    one = 1;
    for(i=0; i<n; i++){
        scanf("%d", &k);
        switch(k%5) {
        case 0:
            if (!(k%2)) s1+= k;
            break;
        case 1:
```

```
            cnt2++;
            s2 += k*one;
            one *= -1;
            break;
        case 2:
            cnt3++;
            break;
        case 3:
            cnt4 ++;
            s4 += k;
            break;
        case 4:
            a5 = (k>a5)? k:a5;
            break;
        default: break;
        }
    }
    if (s1) printf("%d", s1);
    else printf("N");
    if (cnt2) printf(" %d", s2);
    else printf(" N");
    if (cnt3) printf(" %d", cnt3);
    else printf(" N");
    if (cnt4) printf(" %.1f", (double)s4/(double)cnt4);
    else printf(" N");
    if (a5) printf(" %d\n", a5);
    else printf(" N\n");
    return 0;
}
```

C. 数素数 (20)

```
#include<stdio.h>
#include<math.h>

int prime[10001],primenum;
void PRIME(){
    primenum = 0;
    prime[ ++primenum ]=2;
    for(int i = 3; primenum < 10000; i += 2)
    {
        for(int j = 1; j <= primenum; j++)
            if(i % prime[j] == 0) break;
            else if(prime[j] > sqrt((double)i) || j==primenum-1)
```

```
            {
                prime[ ++primenum ]=i;
                break;
            }
        }
    }
    int main(){
        PRIME();
        int n, m;
        while(~scanf("%d %d", &m, &n)){
            int cnt = 1;
            for(int i = m; i < n; i++)
            {
                printf("%d", prime[i]);
                if(cnt == 10)cnt = 1, printf("\n");
                else cnt++, printf(" ");
            }
            printf("%d\n", prime[n]);
        }
        return 0;
    }
```

D. 福尔摩斯的约会 (20)

```
#include <stdio.h>

int main()
{
    char s1[61], s2[61];
    int d, h, m, i;

    scanf("%s\n%s", s1, s2);
    i = 0;
    d = h = m = -1;
    while (s1[i]!='\0' && s2[i]!='\0') {
        if (s1[i]==s2[i]) {
            if (d<0) {
                switch (d=s1[i]-'A') {
                case 0: printf("MON "); break;
                case 1: printf("TUE "); break;
                case 2: printf("WED "); break;
                case 3: printf("THU "); break;
                case 4: printf("FRI "); break;
                case 5: printf("SAT "); break;
```

```
                case 6: printf("SUN "); break;
                default: d=-1; break;
                }
            }
            else {
                h = s1[i]-'0';
                if (h>=0 && h<10) printf("0%d:", h);
                else {
                    h = s1[i]-'A';
                    if (h>=0 && h<14) printf("%d:", h+10);
                    else h = -1;
                }
            }
        }
        if (h<0) i++;
        else break;
    }
    scanf("%s\n%s", s1, s2);
    i = 0;
    while (s1[i]!='\0' && s2[i]!='\0') {
        if (s1[i]==s2[i]) {
            m = s1[i]-'a';
            if (m>=0 && m<26) m = i;
            else {
                m = s1[i]-'A';
                if (m>=0 && m<26) m = i;
                else m=-1;
            }
        }
        if (m<0) i++;
        else {
            if (m<10) printf("0");
            printf("%d\n", m);
            break;
        }
    }

    return 0;
}
```

E. 德才论 (25)

```
#include <stdio.h>
#include <stdlib.h>
```

```
    #define MAXN 100000

    struct Node {
        int id;
        int v, t, g;
    } S[MAXN];

    int CompareG(const void *a, const void *b)
    {
        int k;
        k = ((const struct Node*)a)->g < ((const struct Node*)b)->g ? 1 : 0;
        if (!k) {
            k = ((const struct Node*)a)->g > ((const struct Node*)b)->g ? -1 :
0;
            if (!k) {
                k = ((const struct Node*)a)->v < ((const struct Node*)b)->v ? 1 :
0;
                if (!k) {
                    k = ((const struct Node*)a)->v > ((const struct Node*)b)->v ?
-1 : 0;
                    if (!k)
                        k = ((const struct Node*)a)->id > ((const struct
Node*)b)->id? 1 : -1;
                }
            }
        }
        return k;
    }

    int main()
    {
        int n, i, j;
        int L, H, cnt;
        int *SS[4], cnts[4];

        cnt = 0;
        scanf("%d %d %d", &n, &L, &H);
        for (i=j=0; i<n; i++) {
            scanf("%d %d %d", &S[j].id, &S[j].v, &S[j].t);
            if (!(S[j].v<L || S[j].t<L)) {
                S[j].g = S[j].v + S[j].t;
                j++;
            }
```

```
        }
        n = j;
        if (!n) {
            printf("0\n");
        }
        else {
            printf("%d\n", n);
            for (i=0; i<4; i++) {
                SS[i] = (int *)malloc(sizeof(int) * n);
                cnts[i] = 0;
            }
            qsort(S, n, sizeof(struct Node), CompareG);
            for (i=0; i<n; i++) {
                if (!(S[i].v<H || S[i].t<H))
                    SS[0][cnts[0]++] = i;
                else if (S[i].t<H && !(S[i].v<H))
                    SS[1][cnts[1]++] = i;
                else if (S[i].v<H && S[i].t<H && !(S[i].v<S[i].t))
                    SS[2][cnts[2]++] = i;
                else SS[3][cnts[3]++] = i;
            }
            for (i=0; i<4; i++)
                for (j=0; j<cnts[i]; j++)
                    printf("%d   %d   %d\n",   S[SS[i][j]].id,   S[SS[i][j]].v,
S[SS[i][j]].t);
        }
        return 0;
    }
```

PAT20131102A

A. The Black Hole of Numbers (20)

```
    #include <stdio.h>

    int next ( int n )
    {
        int d[4], m, i, j;
        m = n;
        i = 0;
        while (n) {
            d[i++] = n%10;
            n /= 10;
        }
        if (i==4) {
            for (j=1; j<4; j++)
                if (d[0]!=d[j]) break;
            if (j==4) {
                printf("%d - %d = 0000\n", m, m);
                return 6174;
            }
        }
        while (i<4) d[i++] = 0;
        for (i=1; i<4; i++) {
            m = d[i];
            for (j=i-1; j>=0 && m<d[j]; j--)
                d[j+1] = d[j];
            d[j+1] = m;
        }
        n = d[3];
        for (i=2; i>=0; i--) n = n*10 + d[i];
        m = d[0];
        for (i=1; i<4; i++) m = m*10 + d[i];
        j = n-m;
        printf("%d%d%d%d - %d%d%d%d = ", d[3], d[2], d[1], d[0], d[0], d[1], d[2],
d[3]);
        if (j<1000) printf("0");
        if (j<100) printf("0");
        if (j<10) printf("0");
        printf("%d\n", j);
        return j;
```

```
}

int main()
{
    int n;

    scanf("%d", &n);
    n = next(n);
    while (n!=6174)
        n = next(n);

    return 0;
}
```

B. Mooncake (25)

```
#include <stdio.h>
#include<stdlib.h>

#define MAXN 1000

struct MK {
    double W, P;
} Cake[MAXN];

int compareP(const void *a, const void *b)
{
    double ap1, ap2;

    ap1 = ((const struct MK*)a)->P / ((const struct MK*)a)->W;
    ap2 = ((const struct MK*)b)->P / ((const struct MK*)b)->W;

    return ap1 > ap2 ? -1 : 1;
}

int main()
{
    double D, Sp;
    int N, i;

    scanf("%d %lf", &N, &D);
    for (i=0; i<N; i++)
        scanf("%lf", &Cake[i].W);
    for (i=0; i<N; i++)
```

```
        scanf("%lf", &Cake[i].P);
    qsort(Cake, N, sizeof(struct MK), compareP);
    Sp = 0.0;
    for (i=0; i<N; i++) {
        if (Cake[i].W > D) {
            Sp += D * Cake[i].P / Cake[i].W;
            break;
        }
        else {
            Sp += Cake[i].P;
            D -= Cake[i].W;
        }
    }
    printf("%.2f\n", Sp);

    return 0;
}
```

C. Speech Patterns (25)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

char text[1048580];
char*words[524290];

int less(const char** a,const char** b){return strcmp(*a,*b);}

int main(){
    int n=0,i=0,w=0,best=0;
    char*sbest="there is no word";
    gets(text);
    for(i=0;text[i];i++){
        int ch=text[i];
        if(ch>='A'&&ch<='Z'){
            ch+=0x20;
        }
        if(ch>='a'&&ch<='z'||ch>='0'&&ch<='9'){
            if(!i||!text[i-1]){
                words[n++]=text+i;
            }
        }else{
            ch=0;
```

```
        }
        text[i]=ch;
    }
    qsort(words,n,sizeof(char*),less);
    w=0;
    for(i=0;i<=n;i++){
        if(!i||i==n||strcmp(words[i-1],words[i])!=0){
            if(i-w>best){
                best=i-w;
                sbest=words[w];
            }
            w=i;
        }
    }
    printf("%s %d\n",sbest,best);
    return 0;
}
```

D. Gas Station (30)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAXV 1010
#define InfD 10000000

int N, M, K, D;

struct Table {
    int known, dist;
} T[MAXV];

typedef struct GNode *Graph;
struct GNode {
    int V, dist;
    Graph next;
};
Graph G[MAXV];

int GetV()
{
    int v;
    char str[5];
```

```
    scanf("%s", str);
    if (str[0]=='G')
        v = atoi(str+1) + N;
    else v = atoi(str);
    return v-1;
}

void Insert(int v1, int v2, int d)
{
    Graph t = (Graph)malloc(sizeof(struct GNode));
    t->V = v2; t->dist = d;
    t->next = G[v1];
    G[v1] = t;
    t = (Graph)malloc(sizeof(struct GNode));
    t->V = v1; t->dist = d;
    t->next = G[v2];
    G[v2] = t;
}

void ReadG()
{
    int i, v1, v2, d;

    for (i=N+M-1; i>=0; i--)
        G[i] = NULL;
    for (i=0; i<K; i++) {
        v1 = GetV();
        v2 = GetV();
        scanf("%d", &d);
        Insert(v1, v2, d);
    }
}

void InitialT(int start)
{
    int i;

    for (i=N+M-1; i>=0; i--) {
        T[i].known = 0;
        T[i].dist = InfD;
    }
    T[start].dist = 0;
}
```

```
int Dijkstra()
{
    int i, j, minD, v, w;
    Graph t;

    for (;;) {
        minD = InfD+1;
        for (j=N+M-1; j>=0; j--)
            if (!T[j].known && T[j].dist<minD) {
                minD = T[j].dist; v = j;
            }
        if (minD > InfD) break;
        if (v<N && T[v].dist>D) return -1;
        T[v].known = 1;
        for (t = G[v]; t; t = t->next) {
            w = t->V;
            if (!T[w].known)
                if (T[v].dist+t->dist < T[w].dist)
                    T[w].dist = T[v].dist+t->dist;
        }
    }
    minD = InfD;
    for (i=0; i<N; i++)
        if (T[i].dist < minD) minD = T[i].dist;
    return minD;
}

int main()
{
    int i, j, GD, GavgD, BestG, maxD, avgD;

    scanf("%d %d %d %d\n", &N, &M, &K, &D);
    ReadG();
    BestG = 0; maxD = 0;
    for (i=0; i<M; i++) {
        InitialT(N+i);
        GD = Dijkstra();
        if (GD > 0) {
            if (GD >= maxD) {
                GavgD = 0;
                for (j=0; j<N; j++)
                    GavgD += T[j].dist;
                if ((GD>maxD) || ((GD==maxD) && (GavgD<avgD))) {
                    BestG = i+1;
```

```
                    maxD = GD;
                    avgD = GavgD;
                }
            }
        }
    }
    if (!BestG) printf("No Solution\n");
    else {
        printf("G%d\n", BestG);
        printf("%.1f %.1f\n", (double)maxD, (double)avgD / (double)N);
    }

    return 0;
}
```

PAT20131102B

A. 部分 A+B (15)

```
#include <stdio.h>
#include <string.h>

long long pn(long long a, long long d)
{
    long long r, p = 0;

    while (a) {
        r = a%10;
        if (r == d) p = p*10+d;
        a /= 10;
    }
    return p;
}

int main()
{
    long long a, d, p1, p2;

    scanf("%lld %lld", &a, &d);
    p1 = pn(a, d);
    scanf("%lld %lld", &a, &d);
    p2 = pn(a, d);
    printf("%lld\n", p1+p2);

    return 0;
}
```

B. A 除以 B (20)

```
#include <stdio.h>
#include <string.h>

#define MAXN 1000

int main()
{
```

```
    char a[MAXN+1];
    int b, n, q, r, flag, i;

    flag = 0;
    scanf("%s", a);
    scanf("%d", &b);
    n = strlen(a);
    r = 0;

    for (i=0; i<n; i++){
        q = r*10 + a[i]-'0';
        if (!flag) flag = q/b;
        if (flag) printf("%d", q/b);
        r = q%b;
    }
    if (!flag) printf("0");
    printf(" %d\n", r);

    return 0;
}
```

C. 锤子剪刀布 (20)

```
#include <stdio.h>

int getin()
{
    char c, t;
    scanf("%c%c", &c, &t);
    switch(c) {
    case 'B': return 3; break;
    case 'C': return 2; break;
    case 'J': return 1; break;
    default: return 0; break;
    }
}

void prnt(int id)
{
    switch(id) {
    case 0: printf("J"); break;
    case 1: printf("C"); break;
    case 2: printf("B"); break;
    default: break;
```

```
    }
}

int main()
{
    int i, N, v1, v2, p1[2], w1[3], w2[3], maxw, maxid;

    scanf("%d\n", &N);

    p1[0]=p1[1]=0;
    for (i=0; i<3; i++)
        w1[i]=w2[i] = 0;

    for (i=0; i<N; i++) {
        v1 = getin();
        v2 = getin();
        if ((v1>v2 && v1!=v2+2) || v2==v1+2) {
            p1[0]++;
            w1[v1-1]++;
        }
        else if (v1 == v2) p1[1]++;
        else w2[v2-1]++;
    }
    printf("%d %d %d\n", p1[0], p1[1], N-p1[0]-p1[1]);
    printf("%d %d %d\n", N-p1[0]-p1[1], p1[1], p1[0]);
    maxw=w1[2]; maxid=2;
    for (i=1; i>=0; i--)
        if (w1[i]>maxw) {
            maxw = w1[i]; maxid = i;
        }
    prnt(maxid);
    printf(" ");
    maxw=w2[2]; maxid=2;
    for (i=1; i>=0; i--)
        if (w2[i]>maxw) {
            maxw = w2[i]; maxid = i;
        }
    prnt(maxid);
    printf("\n");

    return 0;
}
```

D. 数字黑洞 (20)

```
#include <stdio.h>

int next ( int n )
{
    int d[4], m, i, j;
    m = n;
    i = 0;
    while (n) {
        d[i++] = n%10;
        n /= 10;
    }
    if (i==4) {
        for (j=1; j<4; j++)
            if (d[0]!=d[j]) break;
        if (j==4) {
            printf("%d - %d = 0000\n", m, m);
            return 6174;
        }
    }
    while (i<4) d[i++] = 0;
    for (i=1; i<4; i++) {
        m = d[i];
        for (j=i-1; j>=0 && m<d[j]; j--)
            d[j+1] = d[j];
        d[j+1] = m;
    }
    n = d[3];
    for (i=2; i>=0; i--) n = n*10 + d[i];
    m = d[0];
    for (i=1; i<4; i++) m = m*10 + d[i];
    j = n-m;
    printf("%d%d%d%d - %d%d%d%d = ", d[3], d[2], d[1], d[0], d[0], d[1], d[2],
d[3]);
    if (j<1000) printf("0");
    if (j<100) printf("0");
    if (j<10) printf("0");
    printf("%d\n", j);
    return j;
}

int main()
{
```

```
    int n;

    scanf("%d", &n);
    n = next(n);
    while (n!=6174)
        n = next(n);

    return 0;
}
```

E. 月饼 (25)

```
#include <stdio.h>
#include<stdlib.h>

#define MAXN 1000

struct MK {
    double W, P;
} Cake[MAXN];

int compareP(const void *a, const void *b)
{
    double ap1, ap2;

    ap1 = ((const struct MK*)a)->P / ((const struct MK*)a)->W;
    ap2 = ((const struct MK*)b)->P / ((const struct MK*)b)->W;

    return ap1 > ap2 ? -1 : 1;
}

int main()
{
    double D, Sp;
    int N, i;

    scanf("%d %lf", &N, &D);
    for (i=0; i<N; i++)
        scanf("%lf", &Cake[i].W);
    for (i=0; i<N; i++)
        scanf("%lf", &Cake[i].P);
    qsort(Cake, N, sizeof(struct MK), compareP);
    Sp = 0.0;
    for (i=0; i<N; i++) {
```

```
        if (Cake[i].W > D) {
            Sp += D * Cake[i].P / Cake[i].W;
            break;
        }
        else {
            Sp += Cake[i].P;
            D -= Cake[i].W;
        }
    }
    printf("%.2f\n", Sp);

    return 0;
}
```