Contents

1 Section1

1.1 basic

```cpp
#include <bits/stdc++.h>
using namespace std;
#define ll long long

int main() {

    cout<<"for define \n";
    return 0;
}
```

2 Section2 Math

2.1 GCD

```cpp
#include<iostream>
using namespace std;
int GCD(int x,int y){
    while(y != 0){
        return GCD(y,x%y);
    }
    return x;
}

int main(){
    int a,b;
    cin>>a>>b;
    int gcd = GCD(a,b);
    int lcm = a*b/gcd;

    cout<<"最大公因數為: "<<gcd<<'\n';
    cout<<"最小公倍數為: "<<lcm<<'\n';
    return 0;
}
```

3 Section3 String

3.1 string

```cpp
#include<iostream>
#include<string>
using namespace std;

int main(){
//初始化字串
    string s1 = "",s2 = "";
    long long a;
    int b;

//吃整行(含空格)
    getline(cin,s1);

//compare,assign,串接
    s1 == s2;
    s1 = s2;
    s1 += s2[i];

//字串切割,i:起始位置,len:幾個
    s1 = s1.substr(i,len);

//轉成數字或數字轉字串
    s1 = to_string(a);
    s2 = to_string(b);
    a = stoll(s1);
    b = stoi(s2);

//判斷數字,字母
    isdigit(s1[i]);
    isalpha(s2[i]);
    return 0;
}
```

4 Section4 小工具

4.1 permutation

```cpp
#include<iostream>
#include<algorithm>
using namespace std;

int main(){
    string a = "abc";
    string b = "cba";
//一定要先排序，才會有全部的組合
    sort(a.begin(),a.end());

//產生組合的迴圈
    do{
        cout<<a<<"\n";
    }while(next_permutation(a.begin(),a.end()));

//檢查b字串是否為a字串可排出結果
    bool isSamePer =
        is_permutation(a.begin(),a.end(),b.begin());

//產生上一個排列結果
    prev_permutation(a.begin(),a.end());

    return 0;
}
```

4.2 高斯消元

```cpp
#define maxn 500+5
int A[maxn][maxn];
int guassian_elimination(int m, int n){
    int r,i,j,k,u;
    i=j=0;
    while(i<m && j<n){
        r=i;
```

```
8        for(k=i; k<m; k++){//找為1的值
9          if(A[k][j]){
10           r=k;
11           break;
12         }
13       }
14       if(A[r][j]){
15         if(r!=i){//換到first row
16           for(k=0; k<n; k++)
17             swap(A[r][k],A[i][k]);
18         }
19         for(u=i+1; u<m; u++){
20 //需要減時，該row才減第一個row
21           if(A[u][j]){
22             for(k=0; k<n; k++)
23               A[u][k]^=A[i][k];
24           }
25         }
26         i++;
27       }
28       j++;
29     }
30     return n-i;// free variable數量
31 }
```

4.3 最大流

```
1 #define N 105
2 int path[N],adj[N][N];
3
4 memset(adj,0,sizeof(adj));
5 //建雙向邊
6 for(int i=0,u,v,w; i<c; i++){
7     scanf("%d %d %d",&u,&v,&w);
8     adj[u][v] += w;
9     adj[v][u] += w;
10 }
11
12 int flow = 0;
13 while(true){
14     memset(path,0,sizeof(path));
15     queue<int> Q;
16
17     path[s] = s;
18     Q.push(s);
19 //BFS找路徑
20     while(!Q.empty() && !path[t]){
21         int now = Q.front();
22         Q.pop();
23         for(int i=1; i<=n; i++){
24             if(!path[i] && adj[now][i]>0){
25                 Q.push(i);
26                 path[i] = now;
27             }
28         }
29     }
30 //完全沒有路到t就break
31     if(!path[t])
32         break;
33     int min_flow = 1e9;
34 //找最窄的路
35     for(int from=path[t],to=t; from!=to;
       from=path[to=from]){
36         min_flow = min(min_flow,adj[from][to]);
37     }
38 //更新該路徑所有邊的額度
39     for(int from=path[t],to=t; from!=to;
       from=path[to=from]){
40         adj[from][to] -= min_flow;
41         adj[to][from] += min_flow;
42     }
43     flow += min_flow;
44 }
```

5 Section5 Graph

5.1 kruskal

```
1 #define maxn 200005
2 #define MP make_pair
3 int N,M;
4 int par[maxn],Rank[maxn];
5 vector<pair<int,int>> G[maxn*2];//雙向邊，所以X2
6
7 struct edge{
8     int x,y,w;
9     bool operator<(const edge& rhs) const{
10        return w<rhs.w;
11    }
12 }e[maxn*2];//雙向邊，所以X2
13
14 int Find(int a){
15    return par[a]==a?a:(par[a] = Find(par[a]));
16 }
17
18 bool Union(int a, int b){
19    a = Find(a);
20    b = Find(b);
21    if(a==b) return false;
22    int tmp = Rank[a] + Rank[b];
23    if(Rank[a]>=Rank[b]){
24        Rank[a] = tmp;
25        par[b] = a;
26    }
27    else{
28        par[a] = b;
29        Rank[b] = tmp;
30    }
31    return true;
32 }
33
34 int kruskal(){
35    for(int i=0; i<N; i++){
36        G[i].clear();
37        par[i] = i;
38        Rank[i] = 1;
39    }
40    int m = 0, tot = 0;
41    for(int i=0,u,v,w; i<M; i++){
42        scanf("%d %d %d",&u,&v,&w);
43        e[m++] = edge{u,v,w};
44        e[m++] = edge{v,u,w};
45        tot += w;
46    }
47    sort(e,e+m);
48
49    int mst = 0, cost = 0;
50    for(int i=0,u,v,w; i<m; i++){
51        u = e[i].x;
52        v = e[i].y;
53        w = e[i].w;
54        if(Union(u,v)){
55            cost += w;
56            mst++;
57            G[u].push_back(MP(v,w));
58            G[v].push_back(MP(u,w));
59        }
60        if(mst==N-1)
61            break;
62    }
63    return cost;
64 }
```

5.2 floyd

```
1 //N為點的個數，G為記錄路徑長的二維振烈
2 for(int k=0; k<N; k++){
3     for(int i=0; i<N; i++){
```

```
4          for(int j=0; j<N; j++){
5              G[i][j]=min(G[i][j],G[i][k]+G[k][j]);
6          }
7      }
8  }
```

5.3 Dijkstra

```
1  struct Data{
2      int u,w;
3      bool operator<(const Data&rhs) const
4      {
5          return w>rhs.w;
6      }
7  };
8
9  void sol(int s){
10     memset(d,0x3f,sizeof(d));
11     memset(vis,0,sizeof(vis));
12     d[s] = 0;
13     priority_queue<Data> pq;
14     pq.push(Data{s,0});
15
16     while(!pq.empty()){
17         Data k = pq.top();
18         pq.pop();
19         int u = k.u;
20         if(vis[u]) continue;
21         vis[u] = 1;
22
23         for(int i=0; i<G[u].size(); i++){
24             int v = G[u][i].first, w = G[u][i].second;
25             if(d[v]>d[u]+w){
26                 d[v] = d[u] + w;
27                 pq.push(Data{v,d[v]});
28             }
29         }
30     }
31 }
```

5.4 SPFA

```
1  #define N 1005
2  #define MP make_pair
3  typedef pair<int,int> PII;
4
5  int n,m;
6  int dis[N],cnt[N];
7  vector<PII> G[N];
8  bool inq[N];
9
10 bool SPFA(){
11     memset(dis,0x3f,sizeof(dis));
12     memset(inq,false,sizeof(inq));
13     memset(cnt,0,sizeof(cnt));
14
15     queue<int> Q;
16     dis[0] = 0;
17     Q.push(0);
18     inq[0] = true;
19     while(!Q.empty()){
20         int u = Q.front();
21         Q.pop();
22         inq[u] = false;
23         for(int i=0; i<G[u].size(); i++){
24             int v = G[u][i].first, w = G[u][i].second;
25             if(dis[v]>dis[u]+w){
26                 dis[v] = dis[u] + w;
27                 if(!inq[v]){
28 //如果鬆弛超過n次，代表有負環
29                     if(++cnt[v]>=n)
30                         return true;
31                     inq[v] = true;
```

```
32                     Q.push(v);
33                 }
34             }
35         }
36     }
37     return false;
38 }
```

6 Java

6.1 java biginterger

```
1  import java.io.*;
2  import java.util.*;
3  import java.math.BigInteger;
4
5  public class bigint {
6      public static void main(String args[]) {
7          Scanner cin = new Scanner(System.in);
8  //Java大數運算宣告BigInteger
9          //首先宣告plus代表做加法運算
10         BigInteger plus = BigInteger.valueOf(0);
11         //首先宣告minus代表做減法運算
12         BigInteger minus = BigInteger.valueOf(0);
13         while ( cin.hasNext() ) {
14 //接下來讀入一整行字串
15             String str = cin.next();
16             //宣告num代表讀入進來的一整行數字
17             //然後把str丟到BigInteger num裡面
18             BigInteger num = new BigInteger(str);
19             if ( str.equals("0") ) break;
20             else {
21                 plus = plus.add(num);
22                 minus = minus.subtract(num);
23             }
24         }
25         System.out.print("The plus sum is " + plus +
                "\n");
26         System.out.print("The minus sum is " + minus
                + "\n");
27     }
28 }
```

7 數學公式

7.1 thm

- 中文測試

- $\sum\limits_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$