

Contents

1	Section1	1
1.1	basic	1
2	Section2 Math	1
2.1	GCD	1
3	Section3 String	1
3.1	string	1
4	Section4 小工具	1
4.1	permutation	1
4.2	高斯消元	1
5	Section5 Graph	2
5.1	kruskal	2
5.2	floyd	2
5.3	Dijkstra	2
6	Java	2
6.1	java biginterger	2
7	數學公式	3
7.1	thm	3

1 Section1

1.1 basic

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 #define ll long long
4
5 int main() {
6
7     cout<<"for define \n";
8     return 0;
9 }
```

2 Section2 Math

2.1 GCD

```

1 #include<iostream>
2 using namespace std;
3 int GCD(int x,int y){
4     while(y != 0){
5         return GCD(y,x%y);
6     }
7     return x;
8 }
9
10 int main(){
11     int a,b;
12     cin>>a>>b;
13     int gcd = GCD(a,b);
14     int lcm = a*b/gcd;
15
16     cout<<"最大公因數為: "<<gcd<<"\n";
17     cout<<"最小公倍數為: "<<lcm<<"\n";
18     return 0;
19 }
```

3 Section3 String

3.1 string

```

1 #include<iostream>
2 #include<string>
3 using namespace std;
4
5 int main(){
6     //初始化字串
7     string s1 = "",s2 = "";
8     long long a;
9     int b;
10
11     //吃整行(含空格)
12     getline(cin,s1);
13
14     //compare,assign,串接
15     s1 == s2;
16     s1 = s2;
17     s1 += s2[i];
18
19     //字串切割,i:起始位置,len:幾個
20     s1 = s1.substr(i,len);
21
22     //轉成數字或數字轉字串
23     s1 = to_string(a);
24     s2 = to_string(b);
25     a = stoll(s1);
26     b = stoi(s2);
27
28     //判斷數字,字母
29     isdigit(s1[i]);
30     isalpha(s2[i]);
31     return 0;
32 }
```

4 Section4 小工具

4.1 permutation

```

1 #include<iostream>
2 #include<algorithm>
3 using namespace std;
4
5 int main(){
6     string a = "abc";
7     string b = "cba";
8     sort(a.begin(),a.end());
9     do{
10         cout<<a<<"\n"; //把更新的字串印出
11     }while(next_permutation(a.begin(),a.end())); //產生下一個排列
12
13     bool isSamePer =
14         is_permutation(a.begin(),a.end(),b.begin()); //檢查b字串
15     prev_permutation(a.begin(),a.end()); //產生上一個排列結果
16
17     return 0;
18 }
```

4.2 高斯消元

```

1 #define maxn 500+5
2 int A[maxn][maxn];
3 int gaussian_elimination(int m, int n){
4     int r,i,j,k,u;
5     i=j=0;
6     while(i<m && j<n){
7         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     }
```

```

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                 if(A[u][j]){//需要減時，該row才減第一個row
21                     for(k=0; k<n; k++){
22                         A[u][k]^=A[i][k];
23                     }
24                 }
25                 i++;
26             }
27             j++;
28         }
29         return n-i;// free variable數量
30     }

```

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 }

```

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 +
26     "\n");
27 System.out.print("The minus sum is " + minus
28     + "\n");
29 }
```

7 數學公式

7.1 thm

- 中文測試

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