Contents

1 語法

1.1 c++

```cpp
// c++ code
#include <bits/stdc++.h>
lower_bound(a, a + n, k);      //最左邊 ≥ k 的位置
upper_bound(a, a + n, k);      //最左邊 > k 的位置
upper_bound(a, a + n, k) - 1; //最右邊 ≤ k 的位置
lower_bound(a, a + n, k) - 1; //最右邊 < k 的位置
[lower_bound, upper_bound) //等於 k 的範圍
equal_range(a, a+n, k);

// 從小到大
priority_queue<int, vector<int>, greater<int>>pq

insert(it,x)//向vector的任意迭代器it處插入一個元素x，時間複
erase(it)//刪除迭代器爲it處的元素，erase(first,last)
//刪除一個區間[first,last)內的所有元素，時間複雜度均爲O(N)

set
insert(x) //將x插入set中   O(log(n))
count(x)  //回傳x是否存在於set中()  O(log(n))
erase(x)  //刪除在set中的x   O(log(n))
clear() //刪除set中所有元素 O(n)
empty() //回傳是否為空   O(1)
size()  //回傳共有幾個元素   O(1)

map
insert(x) //將x這個pair插入map中   O(log(n))
count(x)  //回傳x這個key是否在map中 O(log(n))
erase(x)  //刪除在map中key為x的 O(log(n))


#include <bits/stdc++.h>
using namespace std;

int main(){
  set<int>s;
  for(int i = 0; i < 10; i++){
    s.insert(i);
  }
  cout << "lower bound: " << *s.lower_bound(5) <<
      '\n';// 5
  cout << "upper bound: " << *s.upper_bound(5) <<
      '\n';// 6

  if(s.lower_bound(20) == s.end()){
    cout << "all elements are less than 20\n";
  }
}
```

1.2 python

```python
sorted((4,1,9,6),reverse=True)
fruits = ['apple', 'watermelon', 'pear', 'banana']
a = sorted(fruits, key = lambda x : len(x))
print(a)
# 輸出：['pear', 'apple', 'banana', 'watermelon']
divmod(a,b)
把除數和餘數運算結果結合起來，返回一個包含商和餘數的元組(a
    // b, a % b)
int()
用於將一個字串或數字轉換為整型。函式語法為:class
    int(x,
    base=10)，x--字串或數字;base--進位制數，預設十進位制。
pow(base, exp[, mod])
回傳 base 的 exp 次方；如果 mod 存在，則回傳 base 的
    exp 次方對 mod 取餘數。兩個引數形式的 pow(exp,
    exp) 等價於次方運算子：base**exp。
>>> pow(38, -1, mod=97)
23
>>> 23 * 38 % 97 == 1
True

eof 寫法
try:
  while True:
    s = input()
except EOFError:
  pass

eval(expression, globals=None, locals=None)
>>>x = 7
>>> eval( '3 * x' )
21
>>> eval('pow(2,2)')
4
>>> eval('2 + 2')
4
>>> n=81
>>> eval("n + 4")
85

list(map(int, input().split()))
 L.append(r)
my_list = ['This' , 'is' , 'a' , 'string' , 'in' ,
    'Python']
my_string = " ".join(my_list)
#This is a string in Python
test = [[0 for j in range(m)] for i in range(n)]
```

2 Graph

2.1 Bellman-Ford

```cpp
#include<iostream>
using namespace std;
const int INF = 1e9;
const int MAXN = 1000;
const int MAXM = 1000;
struct Edge {
    int u;
    int v;
    int w;
};

int n, m;
Edge edges[MAXM];
int dis[MAXN];

// s是起點
bool bellman(int s) {
    for (int i = 0; i < n; i++) {
        dis[i] = INF;
    }
    dis[s] = 0;
    bool relax;
    // 做 n 輪
    for (int i = 0; i < n; i++) {
        relax = false;
        for (int j = 0; j < m; j++) {
```

```
            int u = edges[j].u;
            int v = edges[j].v;
            int w = edges[j].w;
            if (dis[u] == INF) {
                continue;
            }
            if (dis[v] > dis[u] + w) {
                dis[v] = dis[u] + w;
                relax = true;
            }
        }
        if (!relax) {
            break;
        }
    }
    return relax;
}


int main(){

}
```

3 Other

3.1 thm