**Buổi 08 Ứng dụng giải thuật Dijkstra, Floyd**

1. **Bài 1** : **Bài toán đi ra biên**

Cho bảng kích thước 𝑛 × 𝑚 (𝑛, 𝑚 ≤ 1000) các số tự nhiên. Từ một ô có thể di chuyển sang một ô kề cạnh với nó, nhưng không được đi ra ngoài bảng. Hãy tìm một cách đi từ ô (𝑥, 𝑦) cho trước đến một ô biên sao cho tổng số của các ô đi qua là nhỏ nhất.

|  |  |
| --- | --- |
| Input : Table.txt | Output |
| 7 6  19 24 23 18 20 16  19 01 01 05 01 16  17 16 01 13 01 20  18 01 01 17 04 21  14 20 17 04 02 28  01 02 01 02 01 59  14 04 09 60 24 18 | Xuất ma trận (Table.txt) lên màn hình (để tiện theo dõi)  Nhập tọa độ (x,y) của điểm xuất phát ( ví dụ nhập : x = 2, y = 3)  Xuất lên đường đi ngắn nhất ra biên  Đường ra biên gần nhất, xuất phát từ s(2,3) :  15 -> 16 -> 22 -> 28 -> 34 -> 33 -> 32 -> 31 -> 30 .Với độ dài = 14 |

Dòng đầu : 7 dòng, 6 cột

***Hướng dẫn*** : (tất cả viết theo dạng hàm trong Program.cs)

* Để sử dụng các thành phần của class WeightMatrix.cs, cần khai báo thêm các thuộc tính như sau :

public int n;

public int[,] a;

// Các array là biến toàn cục chỉ phục vụ cho giải thuật

int[] pre; int[] dist; bool[] visited;

int[,] p; int[,] d;

// propeties

public int N { get => n; set => n = value; }

public int[,] A { get => a; set => a = value; }

public int[] Dist { get => dist; set => dist = value; }

public int[] Pre { get => pre; set => pre = value; }

public int[,] D { get => d; set => d = value; }

public int[,] P { get => p; set => p = value; }

* File Table.txt 🡪 đọc ra ma trận mt[,]

// Đọc file Matran.txt -> trả về ma trận mt[,]

static int[,] FileToMatrix(string fileName)

{

int[,] mt;

// Doc file Matran.txt -> Ma tran mt

...

return mt;

}

* Nhập tọa độ ô xuất phát : (x,y) 🡪 đỉnh xuất phát : s = x\*col + y
* Từ ma trận mt[,] 🡪 đồ thị ma trận kề có trọng số

Mỗi một ô của ma trận mt là một đỉnh của đồ thị, số hiệu của đỉnh s tương ứng với ô(x,y) được đánh số từ trái qua phải, từ trên xuống dưới và công thức tương ứng :

s = x\*col + y (với row, col là số hàng, số cột ma trận mt)

// Đọc ma trận a --> WeightList g. Hàm trả về một đồ thị danh sách kề

static WeightMatrix MatrixToWeightMatrix(int[,] mt)

{

...

return g;

}

* Với đồ thị ma trận kề 🡪 giải thuật Dijkstra(s) 🡪 pre[] và dist[]
* Từ pre[], dist[] 🡪 xuất đường đi ra biên gần nhất và cho biết độ dài
* Ô ở biên tương ứng với đỉnh có bậc < 4

1. **Bài 2** : Chọn thành phố để tổ chức họp

Có 𝑛 (𝑛 ≤ 100) thành phố được đánh số từ 1 đến 𝑛. Khoảng cách giữa hai thành phố 𝑖 và 𝑗 là 𝑎𝑖𝑗. Người ta muốn tổ chức một cuộc họp quy tụ 𝑛 lãnh đạo thành phố. Hãy tìm một thành phố để tổ chức cuộc họp sao cho khoảng cách của người đi xa nhất là nhỏ nhất có thể.

|  |  |  |
| --- | --- | --- |
| Đồ thị | Input : SelectCty.txt | Output : xuất lên màn hình |
|  | 4  0 3 6 9  3 0 2 0  6 2 0 2  9 0 2 0 | Thành phố được chọn : ?  Khoản cách xa nhất và gần nhất có thể : > |

***Hướng dẫn*** : (tất cả viết theo dạng hàm)

* Thực thi thuật toán Floyd() 🡪 ma trận d[,]
* Mỗi một dòng i của d : d[i, ..] lưu trữ đường đi ngắn nhất từ i đến các đỉnh còn lại 🡪 tìm đường đi dài nhất trong d[i, ..]
* Trong các d[i, ..] dài nhất kể trên 🡪 chọn d[i, ..] nhỏ nhất 🡪 kết quả

1. **Bài 3** : Đường tròn.

Trên mặt phẳng cho 𝑛 đường tròn, các đường tròn được đánh số từ 1 đến 𝑛. Đường tròn thứ 𝑖 được cho bởi bộ ba số thực (𝑥𝑖 , 𝑦𝑖 , 𝑟𝑖 ), trong đó (𝑥𝑖 , 𝑦𝑖 ) là toạ độ của tâm đường tròn và 𝑟𝑖 là bán kính của đường tròn. Một đối tượng nằm trong đường tròn có thể di chuyển tự do trong đường tròn đó với chi phí bằng 0. Nhưng để di chuyển đối tượng đến vị trí ngoài đường tròn thì trước tiên phải di chuyển đường tròn đang chứa đối tượng đến đường tròn nào đó chứa điểm cần đến. Chi phí di chuyển giữa hai đường tròn bằng khoảng cách giữa chúng. Một đối tượng đang ở trong đường tròn 𝑠, hãy tìm phương án di chuyển đối tượng đến đường tròn 𝑡 sao cho tốn chi phí ít nhất.

Dữ liệu vào : File văn bản DuongTron.INP

* Dòng đầu tiên chứa ba số nguyên 𝑛, 𝑠,𝑡.
* 𝑛 dòng tiếp theo, dòng 𝑖 chứa ba số nguyên 𝑥𝑖 , 𝑦𝑖 , 𝑟𝑖 .

Dữ liệu ra : xuất lên màn hình chi phí ít nhất tìm được (xuất 2 số lẻ thập phân)

1. **Bài 4** : Bài toán điều khiển việc thực hiện các dự án lớn.

Một công trình được thực hiện qua nhiều công đoạn. Có những công đoạn mà việc thực hiện nó phải sau khi hoàn thành một số công đoạn nào đó. Mỗi công đoạn i có thời gian hoàn thành là t[i]. Dự án và các công đoạn được mô tả như sau :

|  |  |  |  |
| --- | --- | --- | --- |
| Công đoạn | t[i] | Các công đoạn phải hoàn thành trước | Tên công đoạn |
| 1 | 15 | Không có |  |
| 2 | 30 | 1 |  |
| 3 | 80 | Không có |  |
| 4 | 45 | 2, 3 |  |
| 5 | 24 | 4 |  |
| 6 | 15 | 2, 3 |  |
| 7 | 15 | 5, 6 |  |
| 8 | 19 | 5 |  |

Cho biết thời gian sớm nhất để công trình hoàn thành