|  |  |
| --- | --- |
| Computational Problem Solving |  |
| Holi Cow! |  |
|  |  |



1 Introduction

Holi is an annual celebration of color held in India. Reema, a famous artist, is planning to unveil her pi`ece de r´esistance, entitled Field of Dreams, at the festival. A local farmer has agreed to lend Reema a plot of land on which she plans to create her masterpiece. Scattered across the field are the farmer’s prized cows who are all blissfully slumbering.

Reema has brought with her a variety of paint balls of various sizes and colors that she has already placed at random spots in the field. These paint balls are very large and fragile. Whenever one pops, it splatters paint emanating from its position, forming a colorful circle of a specific radius. If there are other paint balls within the radius, these paint balls also pop, which might set o↵other paint balls, and so forth.

Reema would like to create the most colorful scene possible, by popping just a single paint ball. Of course, she wants to maximize the amount of paint that lands on the sleeping cows. That is, for each paint ball, maximize:

∑ # of all paint bombs that painted the cow

all cows

Can you help her figure out which starting paint ball to choose?

1

2 Problem Solving

Reema has written down all the information about the details and locations of each cow and paint ball in the field:

**cow Daisy 2 6**

**paintball BLUE 8 5 1**

**cow Fauntleroy 10 2**

**paintball RED 4 4 3**

**paintball YELLOW 12 3 3**

**cow Babe 8 4**

**cow Milka 6 3**

**paintball GREEN 6 5 2**

Each cow has a unique name and location (e.g. an x and y coordinate). Each paint ball has a unique color, location, and splatter radius.

All cows within the splatter radius (and on it), are painted. Any paint balls within the splatter radius (including on it), are triggered.

2

3 Implementation

1. Main Program

The main program should be named holicow.py. The field of cows and paint balls is provided to it on the command line. The format will be identical to that in the problem solving. If it is not present, you should print the usage statement and exit:

Usage: python3 holicow.py {filename}

If the file name is provided, but does not exist, you should print the following message and exit:

File not found: {filename}

If the file exists, its contents are guaranteed to all be valid.

1. Output

There are three main parts to the output that your program must produce:

1. Displaying the field. Once the input file is read and the graph is built, the field should be displayed as an adjacency list where each vertex indicates what neighboring vertices it is connected to. A typical line would follow the format (where Vertex is either the name of the cow, or the color of the paint ball):

{Vertex} connectedTo: [{Vertex}, {Vertex}, ...]

1. Trigger each starting paint ball in any order. The trigger message should be:

Triggering {COLOR} paint ball...

The chain reaction of triggering other paint balls should use a message that is tabbed in:

{COLOR} paint ball is triggered by {COLOR} paint ball

If a cow is painted by a triggered paint ball, the message should also be tabbed in:

{COW} is painted {COLOR}!

1. Displaying the optimal result. After all starting paintballs have been triggered, you should display which starting paint ball color painted the cows the most. In the case of a tie, choose one. The message should be:

Triggering the RED paint ball is the best choice with 4 total paint on the cows:

After this, the cows that were painted by the best paint ball should be displayed, by name, one per line, along with the color/s they were painted, each tabbed in e.g.:

3

{Cow} colors: {{Color1}, {Color2}, ...}

In the case where no cows are painted, display the following message instead:

No cows were painted by any starting paint ball!

1. Sample Run

This is a sample run using the input from problem solving:

Field of Dreams

---------------

Daisy connectedTo: []

RED connectedTo: [’GREEN’, ’Milka’, ’Daisy’] Milka connectedTo: []

BLUE connectedTo: [’Babe’] Babe connectedTo: []

YELLOW connectedTo: [’Fauntleroy’] Fauntleroy connectedTo: []

GREEN connectedTo: [’Milka’, ’BLUE’]

Beginning simulation...

Triggering RED paint ball...

GREEN paint ball is triggered by RED paint ball

Milka is painted GREEN!

BLUE paint ball is triggered by GREEN paint ball

Babe is painted BLUE!

Milka is painted RED!

Daisy is painted RED!

Triggering BLUE paint ball...

Babe is painted BLUE!

Triggering YELLOW paint ball...

Fauntleroy is painted YELLOW!

Triggering GREEN paint ball...

Milka is painted GREEN!

BLUE paint ball is triggered by GREEN paint ball

Babe is painted BLUE!

**Results:**

**Triggering the RED paint ball is the best choice with 4 total paint on the cows:**

**Daisy’s colors: {’RED’}**

**Milka’s colors: {’GREEN’, ’RED’}**

**Babe’s colors: {’BLUE’}**

**Fauntleroy’s colors: {}**

1. Testing

There are 3 non-trivial test cases that are stored in files named test1.txt, test2.txt and test3.txt. Also included a file test-description.txt that explains what each of the test cases is supposed to test.

4