Final round. Cosmos

The year is 3013 and the *CodeIT* tournament has become intergalactic. As the final round is approaching, all nonhuman-finalists are getting ready for a long journey. Unfortunately, getting to the Milky Way is a challenge all by itself and a very hard problem deserving of an intelligent solution. The finalists are worried that they might need to solve two hard problems (how to get to the Milky Way and the final problem itself) and that this would put them into disadvantage compared to their Earthly rivals. The organizers are panicking and rightly so – they do not want the biggest cosmic informatics tournament to finish without nonhuman participation, especially without any competitors from MIT(M31(Andromeda) Institute of Technology). Fortunately for them, their grandcolleagues have foreseen this problem a thousand years ago and have kindly asked the finalists then to solve it.

The nonhuman competitors have a cosmic map which is just a 2D projection of the safest routes throughout the cosmos. The map is a grid composed of ***R*** *x* ***C*** cosmic zones (from here on “zone” and “cell” would be used interchangeably). Each zone contains at most one of the most important objects the nonhumans should know about:

*Note: See Clarifications on the page for definitions of distance, neighborhood of zones and groups of spaceships.*

1. **Inhabited planets** – The spaceships are highly advised not to cause any disruption to the life in the Universe and so they cannot go through a zone containing a planet.
2. **Stars** – Even though the spaceships have enough nuclear energy to power their engines, whenever they are near (see below for exact definition of “near”) a star they can move twice as fast. This applies to a group of spaceships as long as the last spaceship is affected by a star.
3. **Black holes** – The Cosmos can be quite a dangerous place at times. Whenever a spaceship is stuck in the gravitational field of a black hole, it cannot move, nor escape through a teleport unless a group of spaceships rescues it (for how, see below)
4. **Teleports** – Since 2323 a multitude of cosmic teleports have been installed and now the spaceships can use this teleport network to move faster. Of course, everything comes at a price – every time a spaceship uses a teleport to move they have to pay a unit of gold. As the organizers are generous and want the finalists to get to the Milky Way as fast as possible, they have provided for the spaceships an initial amount of gold ***G*** that they can use. Every additional unit of gold that the spaceships use must be gathered from gold planets.
5. **Gold planets** – Each such planet has a finite amount of gold that the finalists can extract. To extract one unit of gold, a spaceship has to be on the cell of the gold planet. Once a gold planet’s resources have been depleted, they are never renewed.
6. **Dark matter areas –** Such an area is a set of zones, such that there is a path between any two zones in the set. Each dark matter area has at least one teleport and additionally, all the teleports in dark matter areas are free (i.e. they don’t cost any gold). A group of spaceships can get out of a dark matter area only if its size is more than 1 (for the sake of this problem, one spaceship is a considered a group of size 1)*. This means that the only way for a single spaceship to leave the dark matter area is by teleportation.*

Your task is to help the future organizers of the CodeIT tournament by writing a program **cosmos**, which helps the nonhumans reach the Milky Way.

**Clarifications:**

All the distances on the map are Manhattan ones, i.e. dist(A,B) = |x(A) – x(B)| + |y(A) – y(B)|. Two cells are considered neighboring if their distance is 1.

**Spaceship movement:** The spaceships move in groups, **group being defined** as a set of ships occupying neighboring cells all with equal x or y coordinates, such that each of those cells contains a spaceship that is a part of the group. (i.e. a group is a row / column segment ). Any set of spaceships satisfying this definition is considered a spaceship group (e.g. you are not tied to moving all 5 spaceships in a row, but may move only the first two as a separate group). A group is defined by the coordinates of the spaceships on its two endpoints (or in case of a group of size 1 – the coordinates of the spaceship twice).

All spaceship groups can move simultaneously a distance of 1 zone in each of the four directions (north, south, east, west), the only exception being when the **last** spaceship of the group is affected by a star. Then, the group can move a distance of 1 zones at a time.

How a star affects the movement of a group is only reduced to how the star affects the last spaceship of the group.

A spaceship can escape the gravitational field of a black hole only if it is a part of a group which has a last spaceship not being frozen by a black hole. Respectively, if the last spaceship of any group is within the range of a black hole, the group as a whole would be unable to move.

**Teleportation:** A teleport can be used by the first spaceship of the group. When a group Is teleported all the spaceships move to the location of the matching teleport on the map. Throughout this process, the distance of each ship of the group to the teleports is conserved (i.e. the first ship after teleportation is on the matching teleport), but the direction is reversed (i.e. the spaceship that was south of the first, is now north of it).

**No spaceship can enter zones** with inhabited planets, stars, black holes or other spaceships, nor can they go outside of the map. **The only exception is when a spaceship** is when a spaceship flies over such a zone (i.e. is moving 2 units and goes above a zone containing a planet).

For more info, please see the example test case.

# Input:

On the first line of the input file ***cosmos.in*** will be three whole numbers – ***R*** (rows)***, C*** (columns)***, G*** (initial amount of gold). ***R*** rows follow, each containing ***C*** characters, each of which is one of:

1. “-“ – empty space,
2. Digit from 1 to 9 – a gold planet with the respective amount of gold on it
3. “t” – a teleport,
4. “b” – a black hole,
5. “s” – a star,
6. “p” – a planet,
7. “h” – nonhuman spaceship.

The dark matter areas are signified by having the corresponding character for each of the above objects transformed as if by pressing the Shift button. For example, a teleport in a dark matter area would be “T”, a star – “S”, a gold planet with three units of gold – (Shift ) + “3” = “#”, etc.

4 integers follow – ***X, Y, W, Z*** - the coordinates of the upper left and lower right corner of the rectangular (since 2893) Milky Way. Each spaceship that is within the boundaries of this rectangle after the journey is considered successful.

*K*/2 lines follow, ***K*** being the number of teleports on the map. Each of those lines contains the coordinates of a pair of matching teleports. For example, 0 0 1 1, tells us that the teleport at (0, 0) is connected to that at (1, 1). All coordinates would be 0-based (i.e. rows are from 0 to R – 1, columns – from 0 to C – 1).

# Output:

On the first line of the output file ***cosmos.out*** your program must output one whole number ***O –*** the number of operations your program would like to execute. Each of those operations would be described by 6 integers on a separate line. The first 2 of those 6 integers describe the coordinates of the spaceship considered first for the group, the next 2 – the coordinates of the spaceship considered last. Next, an integer between 1 and 4 follows. It shows the direction of movement of the group. The directions are as follows: 1 = north, 2 = east, 3 = south, 4 =west. The final, sixth integer can be one of:

* 1 – the group is moved by a distance of 1 cell. If the first ship lands on a teleport, it **won’t** be used.
* 2 – same as the previous options, but this time a boost from a nearby (from the point of view of the last spaceship for the group) star is used. Still, teleports are not used.
* If the number is -1 or -2, the operation is the same as if the number was 1 or 2 (respectively), but teleports are used necessarily. If the first spaceship does not land on a teleport after this operation, it is considered an invalid move.
* If the number is 0 and the first spaceship is already on a teleport, it is used. The direction (the 5th integer) is ignored in this case, but must still be in the 1 to 4 range.

# Constraints:

2 ≤ **R, C** ≤ 100, 0 ≤ **G, K** ≤ 1000, 1 ≤ **N** ≤ 80.

A zone is within the range of a black hole if the distance between the two is at most 1.

For stars, the similar distance is 3.

# Example:

|  |  |  |
| --- | --- | --- |
| **cosmos.in:** | **cosmos.out:** | **Explanation:** |
| 5 5 0  ----b  --t--  \_\_\_\_\_  T\_\_Sh  H\_\_--  0 0 1 4  1 2 3 0 | 3  3 4 3 4 4 2  3 2 3 2 4 2  3 0 4 0 1 0 | At first we have a spaceship at [4,0] which is a part of a dark matter area. The ship can escape the area only by teleporting from [3,0] or by combining with another ship.  With the first two operations in the out file the [4,0] ship is moved on top of the teleport at [3,0]. Note that we are moving 2 zones at a time because of the star at [3,3].    Now that both ships on the map are next to one another (and on top of a teleport) they can use it to move to the Milky Way directly. |

# Scoring:

The scoring would be done using the formula , where **min** is the minimal result, received by any of the competitors, and **yours** is , where:

* ***N*** is the number of spaceships,
* ***s*** are the spaceships in the Milky Way after all the *O* operations,
* ***О*** is the number of operations itself,
* ***g*** = ***max*** (*gold in the end –* ***G****,* 0). The amount of extracted, but unused gold.
* The coefficients ***a****,* ***b****,* ***c***have values: **a** = , **b** = 1, **c** = -5.

Your program will receive 0 points for a test if any of the following is the case:

- An instruction to move a spaceship to an invalid zone was executed;

- It fails to get at least N/2 spaceships to the Milky Way.

- It runs for more than 5 seconds.

# Testing:

The following table gives additional information about the testing:

|  |  |  |  |
| --- | --- | --- | --- |
| **% from the final tests** | **Small (24** %) | **Type 2 (36** %) | **Type 3 (40** %) |
| R,C ≤ | 20 | 60 | 100 |

# Test generator:

For convenience, all the competitors would have access to the generator which was used to create the final tests for this problem. The test generator would be uploaded on the CodeIT website.

The generator accepts 15 integers as given by the table

|  |  |
| --- | --- |
| int \_\_seed\_\_; | The seed for srand() (SEED). |
| int R, C; | The size of the field |
| int G; | Amount of starting gold. |
| int K; | Number of teleports. |
| int N; | Number of nonhuman spaceships. |
| int P; | Number of inhabited planets. |
| int GP; | Number of gold planets. |
| int X, Y, Z, W; | The coordinates of the Milky Way corners. |
| int D; | Number of dark matter areas. Some may overlap so they would be seen as one from the solver’s point of view. |
| int S; | Number of stars. |
| int B; | Number of black holes. |

All of those numbers, with the exception of \_\_seed\_\_, can be set to -1, if you want it to be randomly generated. In any other case, the numbers signifies the **exact value of the corresponding parameters for the generated test.**