Unix system programming

Zappy Bibicy

Contact
b-psu-330@epitech.eu

Contents

Instructions

- The project is to make per group of 4 to 6 persons.
- You project MUST be compliant with the norm.
- You can use the whole of the standard C library.
- The turn-in must contain an author file filled with the login of each member of the group, as follows:

```
1 (user@host 42) cat auteur
2 login1
3 login2
4 login3
5 login4
6 login5
7 login6
8 (user@host 43)
```

- Your sources shall be turned-in on the `PSU_year_zappy` directory
  ex: PSU_2013_zappy for the 2013-2014 scolar year

Environment

.1 Geography

The goal of the game is to handle a world and its habitants. This world, named `Trantor` is geographically constituted with plains, without relief: no crater, nor valley, nor mountain. The gameboard represents the entire surface of the world as a planisphere. If a player leaves by the right of the plate, he gets back on the left.

The game is played in teams. The teams who wins is the one in which six players have reached the maximum elevation.

.2 Resources

The environment in which we evolve is pretty rich in resources (whether mineral or food). Thus, we just have to walk on this planet to discover this amazing food or various stones from different natures.

Those stones may be from six different kind:

1. linemate
2. deraumere
3. sibur
4. mendiane
5. phiras
6. thystame

The resources are generated by the server. This generation MUST be random.

.3 Activities

The people from Trantor have two kind of occupation:

- feeding
- looking for stones and grabbing them

The elevation represents an important activity in the life of a Trantorian.

.4 Individuals

The Trantorian is peaceful. He is not violent nor aggressive. He walks cheerfully looking for stones and food to feed himself. He crosses his neighbors without difficulties, even those who are in the same square. He can see as far as his visual capabilities allow it. The Trantorian is immaterial, he is fuzzy and occupies the whole square in which he is located. It's impossible to distinguish where he looks when you meet him. The food that the Trantorian picks up is the only resource that he needs for survival. A food unit enables him to survive $126 \times t$ time units.

.5 The elevation ritual

The goal is that all rise in the Trantorian hierarchy. This ritual, that increases the mental and physical capabilities, must be practiced according to particular rules. There shall be in the same square unit:

- a combination of stones
- a certain number of players of equal level

The player begins the incantation and the elevation is then in progress. It's not necessary for the players to be from the same team. Only their level matters. All players of the group performing the incantation reaches the next level.

Transmitted to every generation, the secret of the elevation is:

elevation	number of players	linemate	deraumere	sibur	mendiane	phiras	thystame
1-2	1	1	0	0	0	0	0
2-3	2	1	1	1	0	0	0
3-4	2	2	0	1	0	2	0
4-5	4	1	1	2	0	1	0
5-6	4	1	2	1	3	0	0
6-7	6	1	2	3	0	1	0
7-8	6	2	2	2	2	2	1

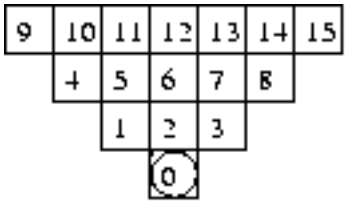
.6 Vision

For various reasons, the player's vision field is limited. Each elevation increases the vision of one unit forward and one on each side of the new row. At first level, the vision unit is equal to 1.

The players needs to send the command voir to know what is around him. The servers will answer a string as follows:

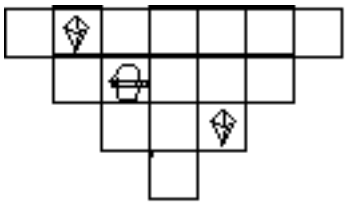
```
voir
{objects-in-square0, objects-in-square1, ..., objects-in-squareP, ... }
```

The following diagram describes the numbering principles.



For example, in this case we obtain:

```
{,,, thystame,, food,,,,, thystame,,,,,}
```



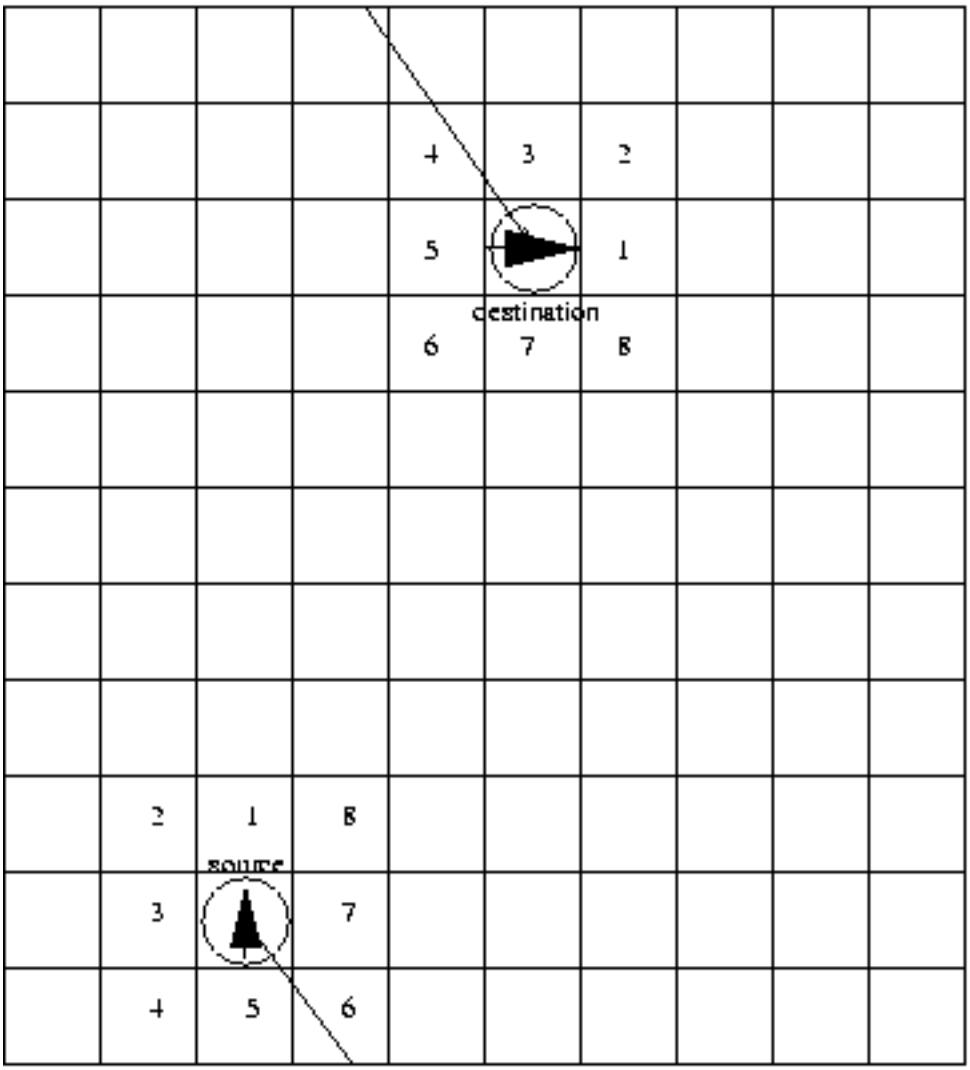
When there is more than an object in a square, they are all indicated and separated using a space. There is an example for a level 1 player with two objects in the square number 1:

```
voir
{, player deraumere,,}
```

.7 Sound transmission

The sound is a wave that propagates in a linear way. The player hears the broadcasts without knowing who broadcasts. They just perceive the direction from which the sound and the emitted message come from. The direction is indicated by the numbering of the square unit from where the sound crosses just before getting to the player. This numbering is done by attributing the number “1” to the square in front of the player. Then we count the squares counterclockwise. The world being spheric, the path we choose for the sound transmission will be the shortest path connecting the transmitter to the player for which it is calculated.

The following example shows the path of the sound you should choose, and the numbering of the boxes around the player. In the case where the broadcast is issued from the same box as the receiver player, he will receive the message from the square 0.


4
3
2
5
1
destination
6
7
8
2
1
8
source
3
7
4
5
6

Your work

.1 The programs

The goal is to create two programs. A server built in C shall handle the world and its people. A client, that you can build in the language that you want, will handle one resident sending orders to the server. The client can also be developed in different platforms (not necessarily Unix).

Here are the server's options:

```
-p port
-x width of the world
-y height of the world
-n name_of_team_1 name_of_team_2 ...
-c number of clients allowed at the game beginning
-t time delay for executing actions.
```

The client syntax:

```
-n team name
-p port
-h hostname, localhost by default
```

The server is constituted of only a single process and a single thread. The client is autonomous, after its launch the user cannot influence its behavior. He drives a drone (player) like in the corewar project.

.2 The teams

At the beginning, a team is composed of n players. Each player is controlled by a client. The clients cannot communicate or exchange data between them outside of the game.

A client has 10 units of life at the beginning, thus he can survive 1260 time units, which represents $1260 \times t$ seconds.

.3 The commands

Each player answers to the following actions, and only with the following syntax:

Action	Command	Delay	Answer
go ahead	avance	$\frac{7}{t}$	ok
right turn of 90 degrees	droite	$\frac{7}{t}$	ok
left turn of 90 degrees	gauche	$\frac{7}{t}$	ok
see	voir	$\frac{7}{t}$	case1, case2, case3 ...
inventory	inventaire	$\frac{1}{t}$	linemate n, sibur n, ...
pick up an object	prend objet	$\frac{7}{t}$	ok/ko
put object down on the ground	pose objet	$\frac{7}{t}$	ok/ko
expels players in the same square unit	expulse	$\frac{7}{t}$	ok/ko
broadcast text	broadcast texte	$\frac{7}{t}$	ok
begin the incantation	incantation	$\frac{300}{t}$	elevation en cours niveau actuel : K
fork a player	fork	$\frac{42}{t}$	ok
number of slots not used by the team	connect_nbr	0	valeur
death of a player	-	-	mort

All the commands are transmitted using a string terminated with a newline character.

.4 client/server dialog

The dialog between the client and the server is done using tcp sockets. The port is passed as parameter. The client sends requests without waiting for their accomplishment, the server returns a message confirming the proper conduct of the execution of the requests. The connection of the client to the server is done as follows: the client opens a socket in the server's port, then the server and the client communicate as follows:

```
<-- BIENVENUE\n
--> TEAM-NAME\n
<-- CLIENT-NUMBER\n
<-- X Y\n
```

The CLIENT-NUMBER indicates the number of clients that can still be accepted by server for the TEAM-NAME team. If this number is greater than or equal to 1 a new player can connect.

The client can send up to ten successive requests without receiving answer from the server. Beyond ten requests, the server will stop taking them into account. The server executes the requests of the clients in order of reception. The requests are buffered and the delay of execution of a command only blocks the affected player. X and Y are the dimensions of the world.

.5 The time

Active waiting is not tolerated. It shall not be blocking when clients are stopped or in any phase of the game.

The Trantorians have adopted an international time unit. The time unit is the second. The time of the execution of an action is calculated with the following formula: $action/f$ where f is a frequency (in hertz).

If f = 1, 'avance' takes 7/1 seconds. We'll choose by default, $f = 100$. F will always be an integer. The time referential is the absolute time.

.6 Objects handling

Only the class of the object can be identified. Thus it's impossible to distinguish two objects from the same class. For example, two siburs will have the same denomination because they are part of the same class.

.7 Player's reproduction

A player can reproduce using the fork command.The execution of this command leads to the production of an egg. After spawning, the player who spawned the egg can go about its business. When the eggs hatches, a new player is created, with a random orientation. This operation enables a new client to connect. The connect_nbr command returns the number of active and authorized connections for this family.

Period of egg laying: $\frac{42}{t}$. Delay between spawn and hatch: $\frac{600}{t}$.

.8 Inventory

This commands enables to see what objects the drone has and how much time he can survive. The server will send, for example, the following line:

```
1 {food 345, sibur 3, phiras 5, ..., deraumere 0}
```

.9 The Broadcast

To emit a message, the client shall send the following command to the server:

```
broadcast text
```

The server will send to all the clients the following line:

```
message K,text
```

with K indicating where the sound comes from.

.10 Expulsion

The drone can expel all the drones in the same square unit. He expels them in the direction in which he's looking. When a client sends to the server the expulse command, all the clients sharing the square unit receive the following line:

```
deplacement: K\n
```

with K indicating where the drone comes from.

.11 Graphical interface

The project shall have a graphic visualization available. It shall represent the world. The interface can be integrated to the server, or to a client. The choice is yours, but it's easier to have it in a specific client.

This interface shall integrate at least a 2D visualization using icons, allowing a world representation. A 3D or any other kind will be very appreciated but keep in mind that the graphical interface shall be functional before being beautiful.

To develop this interface, you are free to use the following libraries:

- In C, using the `SDL` library.
- In C++, using the graphical part of the `GDL`, SFML, or Qt libraries.

If you wish to use another language, you shall obtain the agreement of the professor in charge of your Unix System module. Without this agreement, obtained by mail only, any use of an unauthorized library will be considered as cheating.