TI-Concours 2016

Qualifications, 1st category
(z80 and ez80 TI-Basic)

***

Before starting

This is a qualification round, which means that you have until 02.08.2016 11:59 p.m. (UTC + 1) to send your production at info@tiplanet.org. Please put the category (Category 1) in the title, and your first name, last name and complete mailing address. Do not forget to attach your production, which has be sent as a ZIP or RAR archive, whose name is your last name in capital letters without accents, followed by a space and the characters "1Q". This archive will contain all the programs you have made (and nothing else).

If you will to participate in both categories, please send us two separate emails. You can update your production as many times as you like, by sending a new e-mail. For each category, we will judge the last production received in time.

For this qualification round, you will have for each question to write a program, whose name will be: the first 5 letters of your last name, all attached and without accent, followed by the characters "1Q" and question number. For example, if your last name is De Périgny, for Question 4 your program will be named "DEPER1Q4". By doing so, you limit the risk that two distinct programs have the same name. If despite this you realize that someone else whose first five letters of the name are the same as yours is involved in the contest, do not worry, we'll take care of the problem.

All programs must be written entirely in z80 or ez80 TI-BASIC. This means that you can use neither external library nor assembly code. We will also ask you not to use subroutines, and not to protect your code : anyway, judges will have a look at it.

Make sure you have a cable for connection between your calculator and your computer, and the appropriate transfer software. If you have trouble with that, feel free to ask questions on http://tiplanet.org.

Unless you are explicitely told to, you will never have to ask for an input or display anything on the output. The subject will give you variable names, will tell you what each of them represents, and your programs will run under the hypothesis that these variables have been initialized correctly beforehand. Similarly, we will always specify which variables will hold the result sought. You have the right to change the variables used in this entry through your programs, and it is assumed that they are never incorrectly initialized.

The precise scale will not be disclosed, but keep in mind that about half of your score will focus on the accuracy of the results produced, and that the second half will focus on the effectiveness of the implemented algorithms. If you have finished before the deadline, you can always try to make your programs run faster.

The difficulty is growing globally, and it is not necessary to do everything to qualify for the final round. We encourage you to submit what you've done, whatever the number of questions dealt with. Whatever happens, we hope you will have fun searching for answers!

Good luck to everyone !

The escape

You are there, sitting quietly and browsing the Internet (or a TI-Concours subject, as you want) from your computer. At least, that's the last thing you remember ... because after that, it is the total blur, and when you wake up, you are alone, in the dark, without any personal effect on you. While the situation plunges you into misunderstanding, you try despite the total darkness to find an exit.

Question 1

It is assumed that [A] is a matrix having 0 and 1, respectively corresponding to an empty space and a wall. Write a program that, assuming that [A] is a plan of the room where you are located, storages in real variable S the value 1 if you can get out, 0 otherwise. We assume that one can always move freely within a given piece, which means that given two "empty spaces" (represented in the matrix by 0s), there is a way into the room to go from the first one to the second one.

Here is an example :

[[1,1,1,1,1,1,1] S is equal to 1, because there is an exit on the left.

[1,0,0,0,1,1,1] If there were only “1”s on the first column, then S would be equal to 0.

[0,0,1,0,0,0,1]

[1,1,1,1,1,1,1]]

***

After much fumbling, you realize that you are surrounded by four walls, and the only existing door is locked by a secret code! To try to show your presence, you tap on the door and on the walls next to it. In vain ... having nothing else to do, you start typing random codes, hoping that you will find the correct one. But after the 42th try, a synthetic voice, seeming to emanate from the door, asks you:

« What is the first prime number which is strictly greater than 9001 ? ».

Question 2

It is assumed that N is a real variable containing a non-zero integer. Write a program that stores in the real variable P is the smallest prime number strictly greater than N.

***

The door finally opens, and you enter the new room. After a few steps, the door you just opened abruptly closes and the lights turn on. You can’t no go out! In the middle of the room, there is a computer. You turn it on, and a document entitled "Rules of the Game" is displayed on the screen.

In this game, you are opposed to (N-1) other participants. The unfolding is similar to that of a table knockout. At every turn, you encounter an opponent and duel with him. If you win, you are qualified and you play the next round, otherwise you're dead. The game continues until there is one participant left.

Each player (including you) is assigned an input ranking, supposedly representative of your ability to win those games. The table is made in order to let the best player face the worst one, the second the second last, and so on ...

Here are some examples of tables :

Table with two participants :

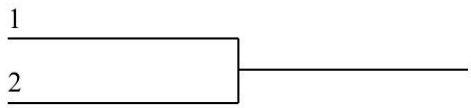


Table with four participants :

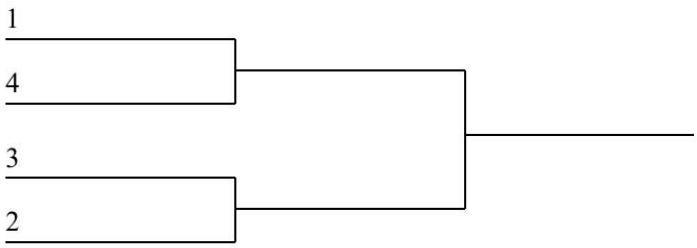
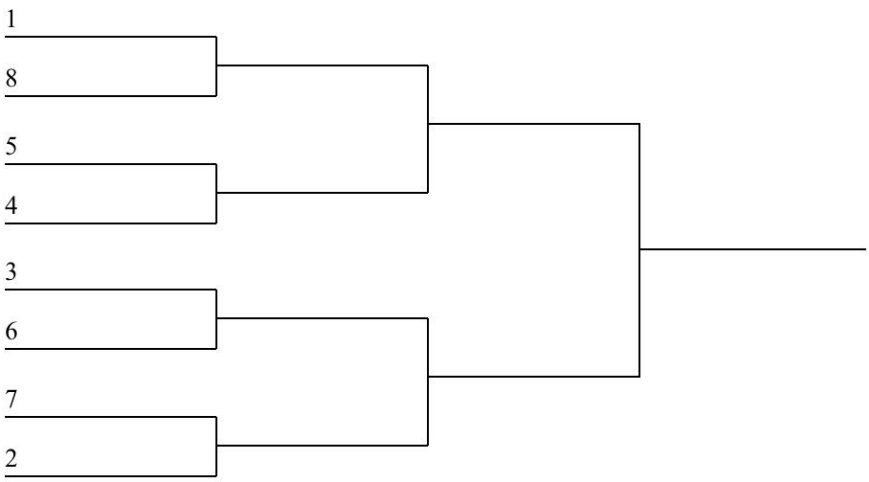


Table with eight participants :



The procedure is the same for 16, 32, 64 ... note that we have each time a power of 2.

Question 3

K being an integer between 1 and 9, write a program that stores in the list L_1 the rankings of the players that are in the table, from top to bottom, in the beginning, if they are 2^K. For example, for K = 3, the answer is $L_1 = \{1, 8, 5, 4, 3, 6, 7, 2\}$.

***

Question 4

Given N the number of participants and C your ranking at the beginning of the game, write a program that, still assuming that N is a power of 2, stores in real variable A the ranking of your first opponent.

***

Obviously, the number of players is not always a power of 2. The best are then exempted from the first round (also called "preliminary round" or "barrage tour"). We can convince ourselves that there exists one and only one table knockout which on one hand checks the definition given above, and on the other hand does not allow all players to be exempted from the first round . For example, if there are 7 players, then the 1st is exempt from the quarter-finals, while the 5th faces the 4th, 3rd the 6th, 7th and 2nd. Thus, the first duel of the 1st will oppose him to the winner of the duel between the 5th and the 4th.

Question 5

Given N the number of participants and C your ranking at the beginning of the game, write a program that stores in the real list L_1 the ranking of your potential opponents for your first duel.

***

Since all your opponents are equipped with some artificial intelligence, you will be unable to defeat anyone without any outside help. Fortunately, the computer made available to you has a database of all the AIs that you are likely to encounter, and their strengths and weaknesses. Thus, by determining in advance your future opponents, you have a chance to get away with exploiting their weaknesses!

It is now assumed that duels are organized round by round, from top to bottom in the table. Thus, in the case where N = 7, the duels will be numbered as follows.

Duel's number	Opponents
1	The 5th against the 4th
2	The 3rd against the 6th
3	The 7th against the 2nd
4	The 1er against the first duel's winner
5	The second round's winner against the third one's

6	The two finalists

Question 6

It is assumed that L_1 contains the list of the previous duels' results: the term number i is 1 if the most top opponent in the table won, and is 2 otherwise. C being still equal to your ranking, write a program that stores in L_2 the list of the rankings of your potential opponents in the next round. If you are already dead, L_2 will be equal to {-1}.

***

The game starts. By applying the strategy developed before, you defeat all your opponents. After winning the final duel, an unlocked door appears in the wall. When you open it, the lights go out ... you go blindly in the new room. And here again, the door closes behind you and suddenly the lights on the ceiling turn on. The room is full of zombies! When they see you, all these vile creatures run towards you. While you are wondering if your end has come, you notice a small door on your right. Grabbing this opportunity, you head over to what seems to be your salvation, and you engulf yourself in the opening.

After blocking the door (which this time had a latch), you find stairs in front of you. You climb them and then you arrive in a sort of control room, with windows. As you approach them, you realize that they overlook the room in which you were just before. By wearing your outlook on the rest of the room, you notice the zombie infested room has a defense system in working condition. On the ceiling, there are a large number of vaporizers to disintegrate the enemy in an area you can define yourself. However, several constraints prevent you from doing what you want:

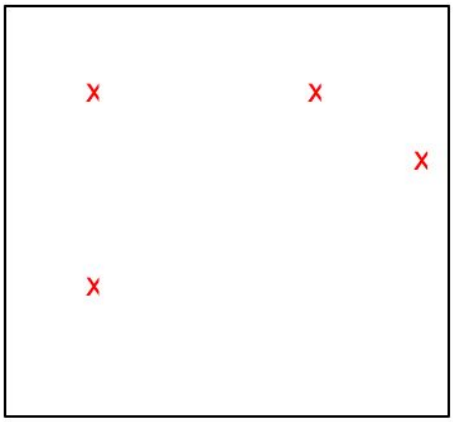
- The start-up time is long and noisy, so if the whole room is not covered by the same time by vaporizers, then the zombies will dodge your attacks;

- The disintegration does not operate if a vaporizer aims an area with two or more zombies;

- If two vaporizers aim the same area, then it causes a fatal bug putting out of service forever the defense system and condemning you to die of hunger in your control room.

The piece being square, you decide to adopt the following strategy: if there is strictly more than a zombie, you cut it into four rooms (in your head, of course). For each area of the room which is still not cut you redo a new division if the condition just mentioned is still not respected. And so on…

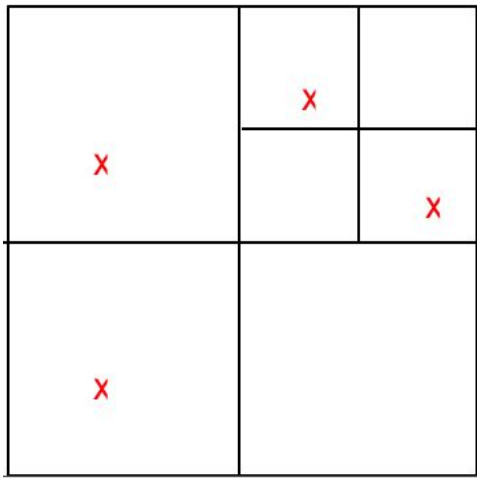
Here is an example. Zombies are in the room as follows.



There are several zombies in the room, so you cut it into four.

In the upper right area there are still two zombies, so we start again, only in this area.

So you define seven distinct areas containing at most a zombie, not overlapping and forming a room partition. This means that if for each zone you use a vaporizer, and you run all the processes together, you can accomplish your goal.



Question 7

It is assumed that the room is a square of side C, and that the summit of the bottom left coordinates are (0,0) in an arbitrary orthonormal reference. Write a program that from both L_1 and L_2 lists representing the abscissas and ordinates of zombies (zombie i has coordinates (L1 (i), L2 (i))), stores in the real variable N the number of vaporizers used by the method described above. In the previous example, N is equal to 7. It is assumed that there is always enough vaporizers.

***

Once all zombies are disintegrated, for a long time nothing happens. Then a door appears in the wall of the room. You open the door, and once again, everything becomes dark. You go along, and then ... you do not remember what happened. What is sure, is that you're back in your room, in front of your computer and that it is 7:00 AM. Maybe it was a bad dream...

THE END

***