“The mapmaker” task

In the neighbour of Neverfindland there is a big country, the Potato Empire, many parts of which are still unknown and uninhabited. Its ruler, Empress French Frie, ordered the mapping and settlement of these areas. As a first step, she commissioned you as imperial mapmaker to discover the landscape. The Empress sets missions to determine what landscapes she wants to see in her empire. Help her to fulfil her wishes as much as possible, so that your reputation can grow accordingly!

Contents

Description of the game

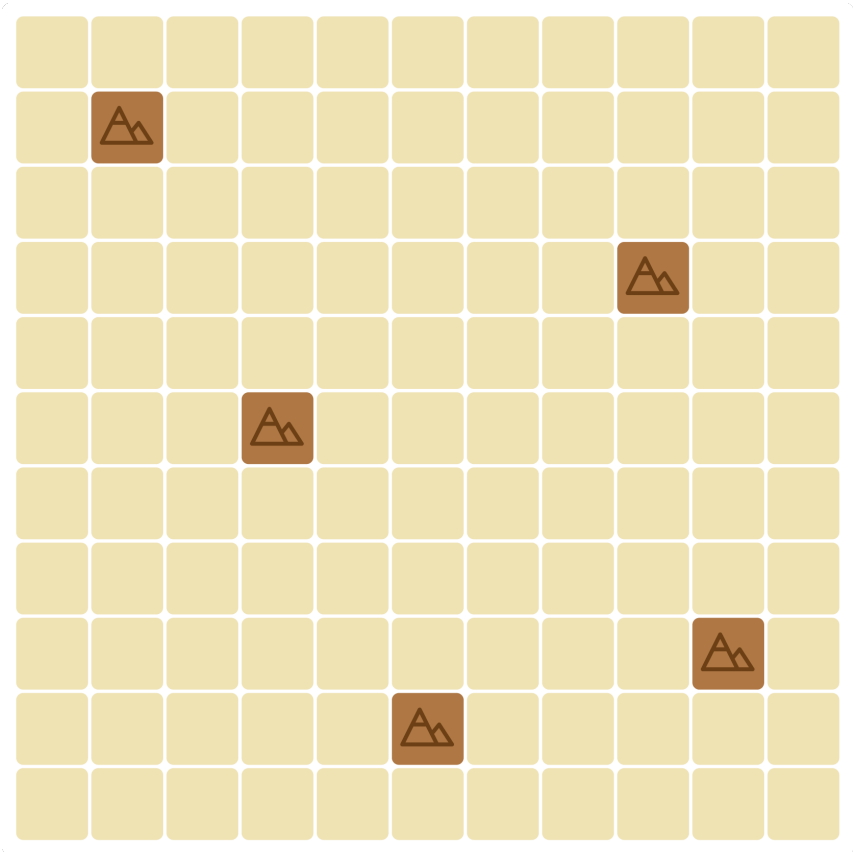
Brief overview

In this single-player game, you have to place map elements of different shapes and terrain types on an 11x11 square grid map. Each element is assigned a time value (1 or 2) and the game consists of 28 time units. At the end (or during) of the game, a number of checks (missions) are performed against the current state of the grid, and the final score is calculated.

Initial state of the map

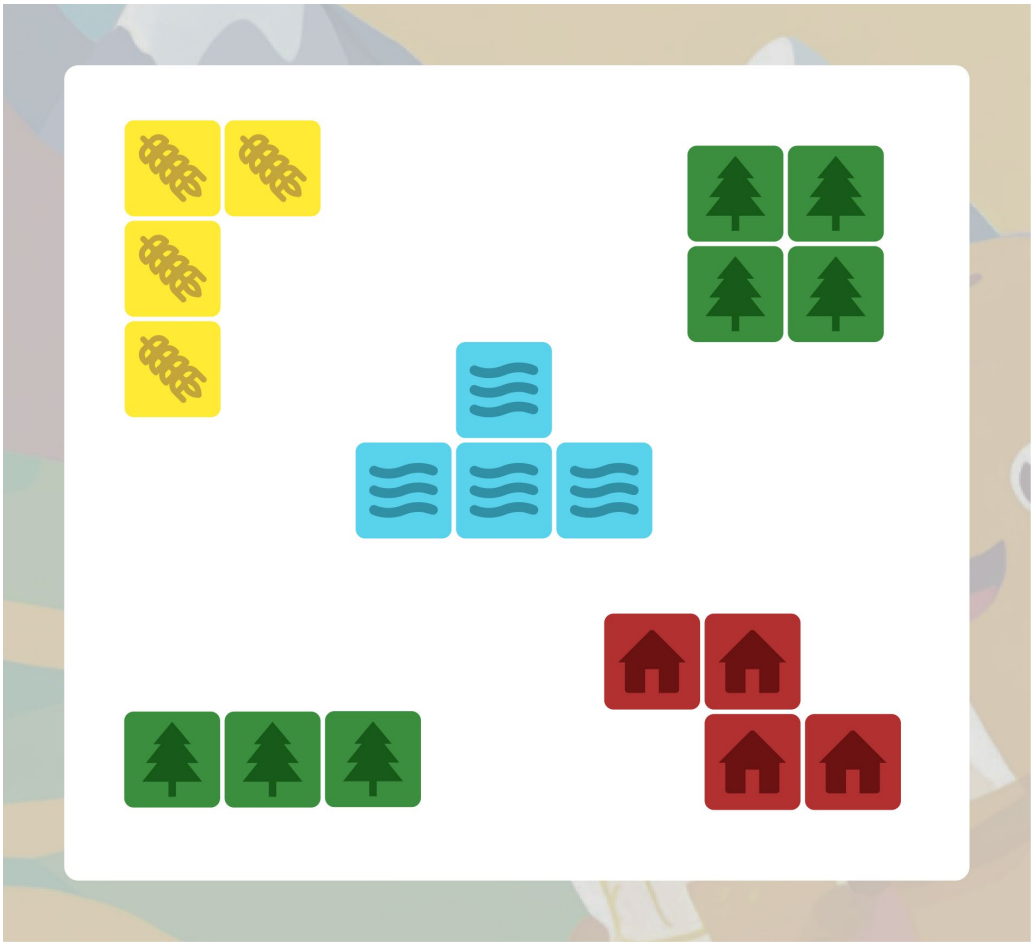
The map is an 11x11 square grid, initially filled with empty cells. The map contains mountain fields in 5 fixed cells. Our mountains are located in the following cells of the map:

(row, column) => (2,2), (4,9), (6,4), (9,10), (10,6)

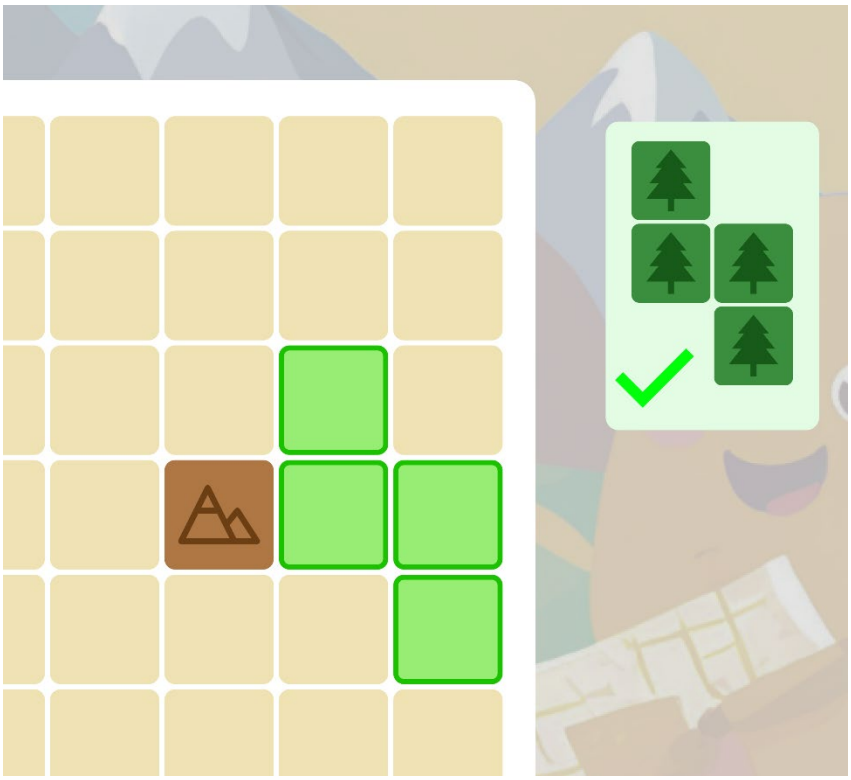


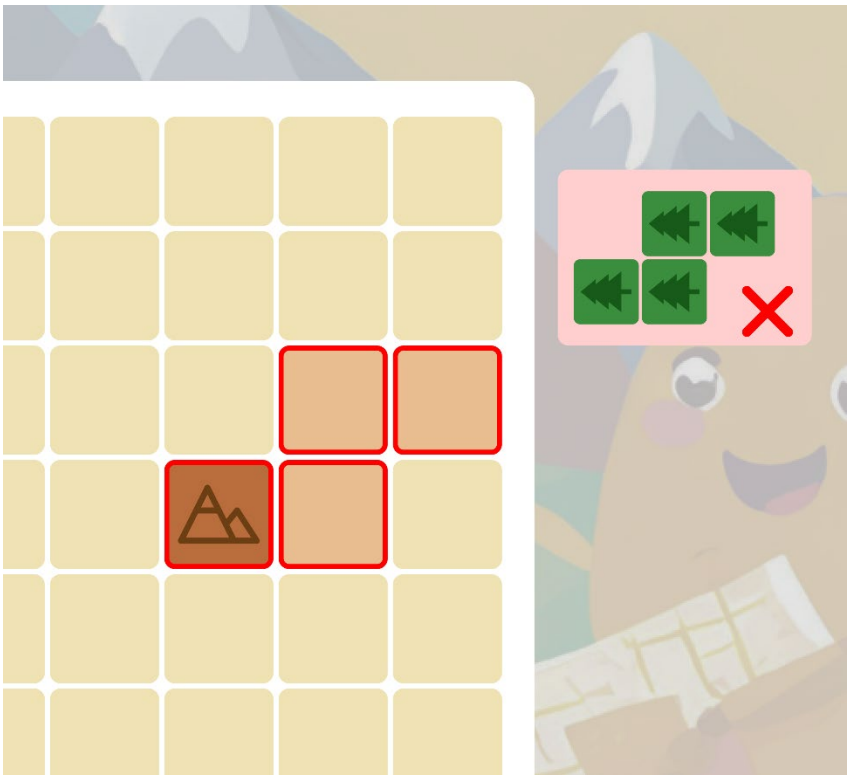
Placing map elements

The types of terrain of map elements that can be placed are: forest, village, farm and water. All possible elements are given below in a JavaScript array, some of them look like this:



The possible elements are shuffled randomly and then you need to place them on the map one by one in sequence. Each map element can be rotated and mirrored, and the map element cannot cover an already reserved field (a mountain is a reserved field), or have any part of it hanging off the map.





End of the game

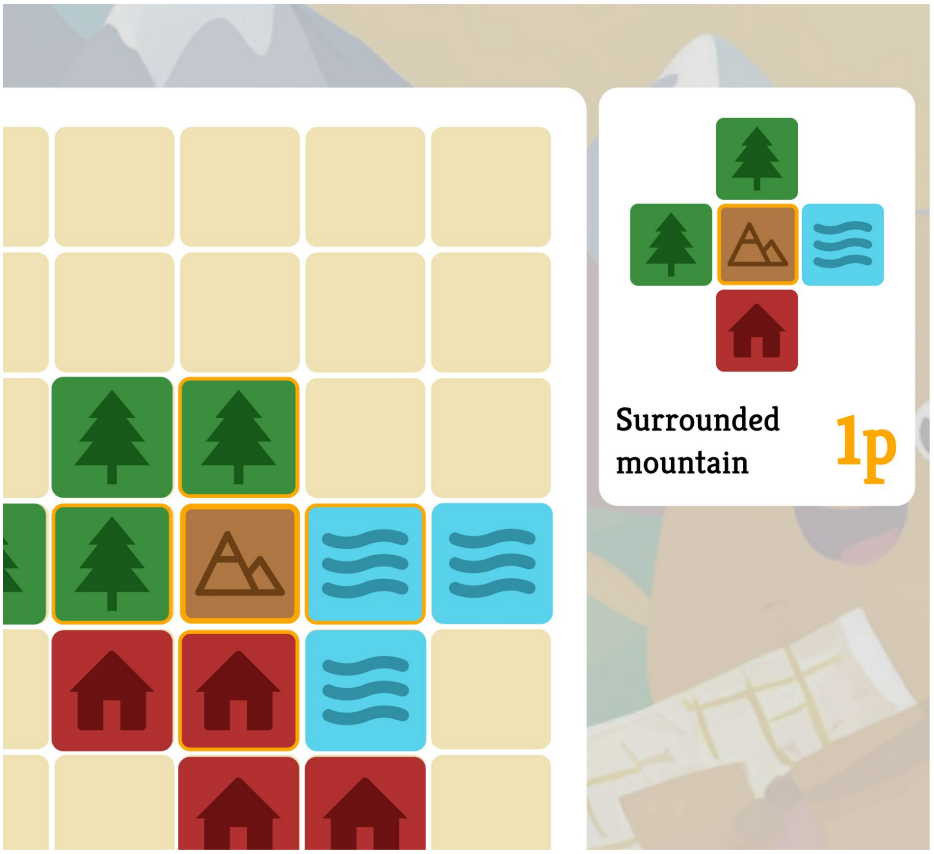
The game lasts up to 28 time units. Each map element is assigned a time unit, which determines how long it takes to explore it. You can draw new map elements until you reach 28 time units. When the total time value reaches or exceeds 28 time units, the game ends. For example, if we have 1 time unit left and we get a map element with two time units, we can still place the map element and then the game ends.

Calculating the score

At the beginning of each game, 4 random mission cards (A,B,C,D) must be selected to score points. An example of a mission card is this:

'You get three points for each of your water fields adjacent to your mountain fields.'

If you surround the mountains on 4 sides, you get 1 point per surrounded mountain.



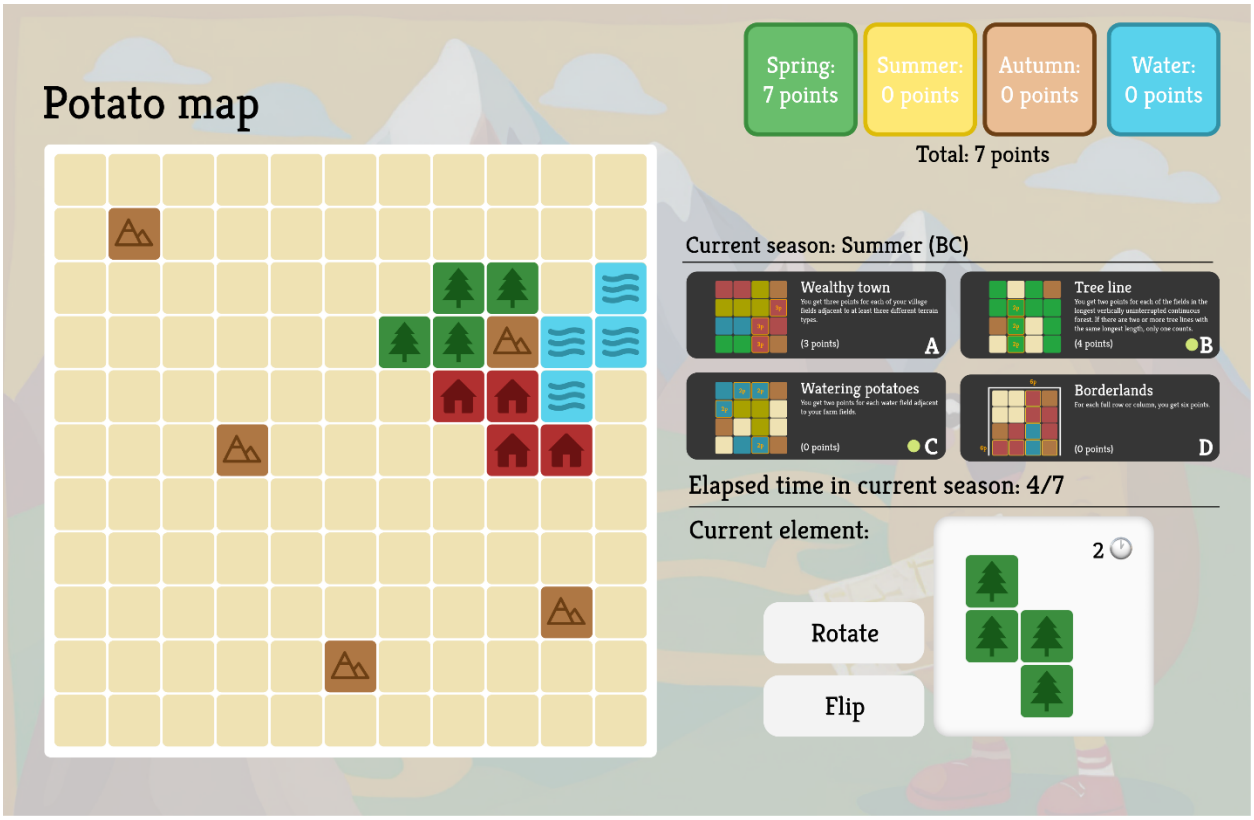
At the end of the game, you have to count the points you got for each mission, and the sum of these will be the final score. For each of the four missions, you must also indicate how many points you have received for each mission!

Seasons

The 28 time units represent one year. It can be divided into 4 seasons, each season lasting up to 7 units of time. If the total time value reaches or exceeds a multiple of 7 while placing the map elements, the season ends.

At the end of each season, you can score for 2 missions. At the end of spring, you can score points for mission A-B, at the end of summer for mission B-C, at the end of autumn for mission C-D and at the end of winter for mission D-A. For each of the four missions, you need to indicate per season how many points you have earned for each mission!

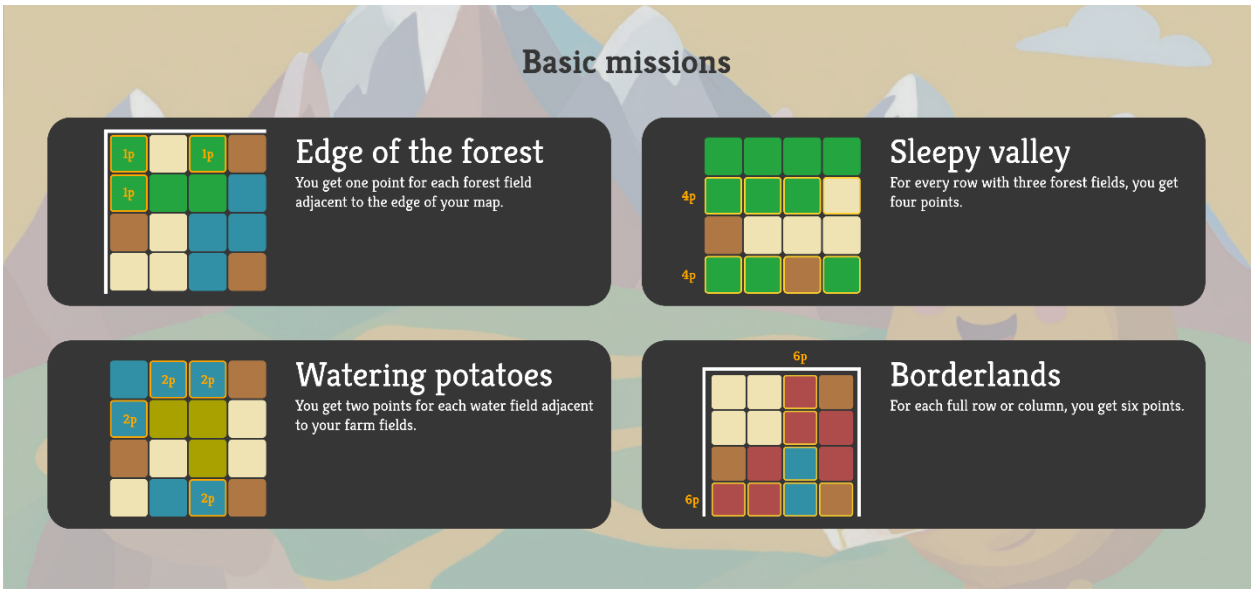
At the end of the game, the points you have earned over the four seasons will be added together to give you your final score.



Missions

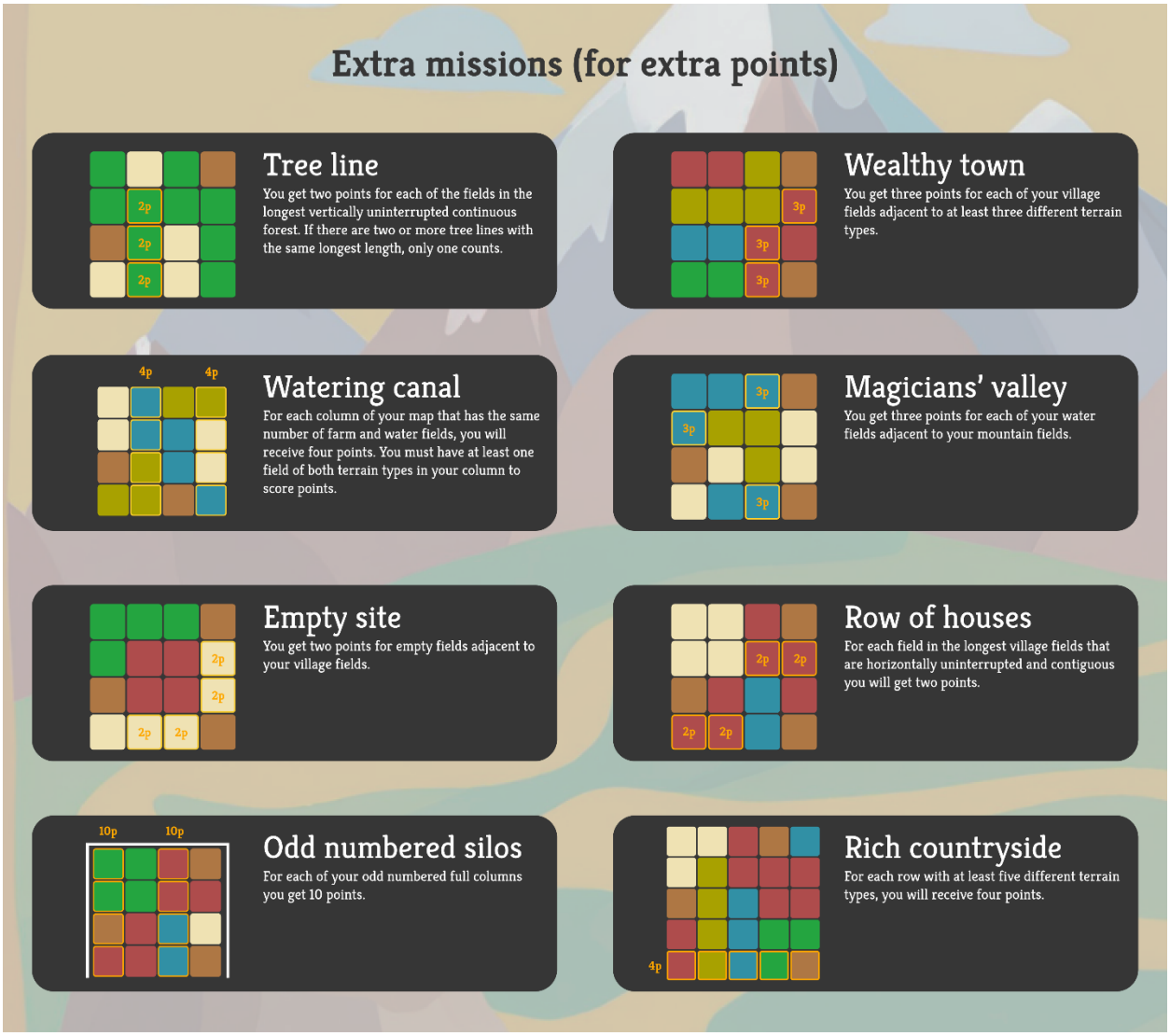
Here you will find the missions to be evaluated in the game and the corresponding pictures.

Basic missions



1. Edge of the forest: you get one point for each forest field adjacent to the edge of your map.
2. Sleepy valley: for every row with three forest fields, you get four points.
3. Watering potatoes: You get two points for each water field adjacent to your farm fields.
4. Borderlands: for each full row or column, you get six points.

Extra missions (for extra points)



1. Tree line: You get two points for each of the fields in the longest vertically uninterrupted continuous forest. If there are two or more tree lines with the same longest length, only one counts.
2. Wealthy town: You get three points for each of your village fields adjacent to at least three different terrain types.
3. Watering canal: For each column of your map that has the same number of farm and water fields, you will receive four points. You must have at least one field of both terrain types in your column to score points.

4. Magicians' valley: You get three points for your water fields adjacent to your mountain fields.
5. Empty site: you get two points for empty fields adjacent to your village fields.
6. Row of houses: for each field in the longest village fields that are horizontally uninterrupted and contiguous you will get two points.
7. Odd numbered silos: for each of your odd numbered full columns you get 10 points.
8. Rich countryside: for each row with at least five different terrain types, you will receive four points.

To help you, we specify the object for the missions:

```
const missions =
{
 "basic": [
  {
   "title": "Edge of the forest",
   "description": "You get one point for each forest field adjacent to the edge of your map."
  },
  {
   "title": "Sleepy valley",
   "description": "For every row with three forest fields, you get four points."
  },
  {
   "title": "Watering potatoes",
   "description": "You get two points for each water field adjacent to your farm fields."
```

```
  },
  {
   "title": "Borderlands",
   "description": "For each full row or column, you get six points."
  }
 ],
 "extra": [
  {
   "title": "Tree line",
   "description": "You get two points for each of the fields in the longest vertically uninterrupted continuous forest. If there are two or more tree lines with the same longest length, only one counts."
  },
  {
   "title": "Watering canal",
   "description": "For each column of your map that has the same number of farm and water fields, you will receive four points. You must have at least one field of both terrain types in your column to score points."
  },
  {
   "title": "Wealthy town",
   "description": "You get three points for each of your village fields adjacent to at least three different terrain types."
  },
  {
   "title": "Magicians' valley",
```

```
    "description": "You get three points for your water fields adjacent to your mountain fields."
   },
   {
    "title": "Empty site",
    "description": "You get two points for empty fields adjacent to your village fields."
   },
   {
    "title": "Terraced house",
    "description": "For each field in the longest village fields that are horizontally uninterrupted and contiguous you will get two points."
   },
   {
    "title": "Odd numbered silos",
    "description": "For each of your odd numbered full columns you get 10 points."
   },
   {
    "title": "Rich countryside",
    "description": "For each row with at least five different terrain types, you will receive four points."
   }
  ],
}
```

Possible map element types

You will find the possible map element types in this array, we have prepared it for you. You have to shuffle it at the beginning of the game and then place it one by one on the map. It is up to you how to do the placing. You could always draw a faint outline of the element you want to place while moving the mouse over the map, but another way is to just click on a cell and it will insert the shape in the given position starting from the top left cell. Objects also have rotation and mirrored data properties, so you can store these in the shapes you draw.

```
const elements = [
  {
    time: 2,
    type: 'water',
    shape: [[1,1,1],
            [0,0,0],
            [0,0,0]],
    rotation: 0,
    mirrored: false
  },
  {
    time: 2,
    type: 'town',
    shape: [[1,1,1],
            [0,0,0],
            [0,0,0]],
    rotation: 0,
    mirrored: false
```

```
},
{
   time: 1,
   type: 'forest',
   shape: [[1,1,0],
           [0,1,1],
           [0,0,0]],
   rotation: 0,
   mirrored: false
},
{
   time: 2,
   type: 'farm',
   shape: [[1,1,1],
           [0,0,1],
           [0,0,0]],
      rotation: 0,
      mirrored: false
   },
{
   time: 2,
   type: 'forest',
   shape: [[1,1,1],
           [0,0,1],
           [0,0,0]],
```

```
    rotation: 0,
    mirrored: false
},
{
    time: 2,
    type: 'town',
    shape: [[1,1,1],
            [0,1,0],
            [0,0,0]],
    rotation: 0,
    mirrored: false
},
{
    time: 2,
    type: 'farm',
    shape: [[1,1,1],
            [0,1,0],
            [0,0,0]],
    rotation: 0,
    mirrored: false
},
{
    time: 1,
    type: 'town',
    shape: [[1,1,0],
```

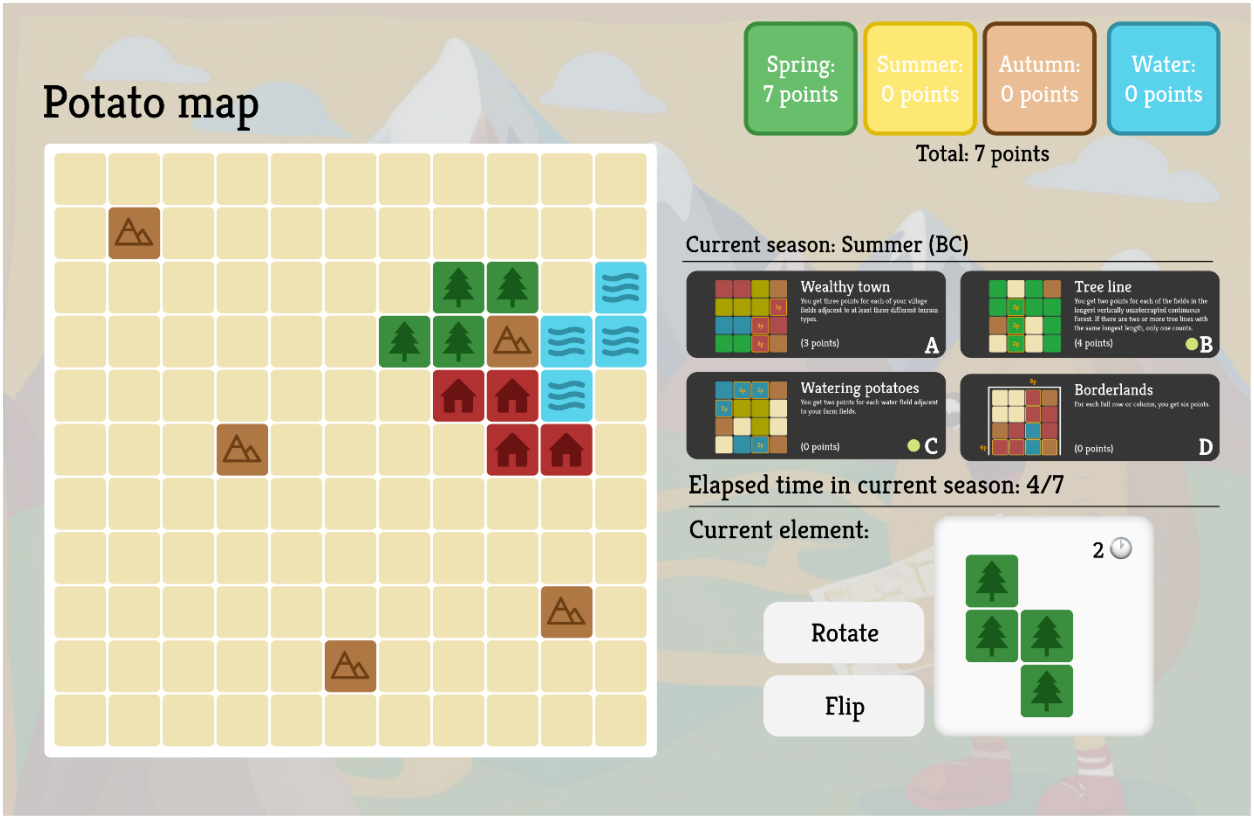
```
        [1,0,0],
        [0,0,0]],
    rotation: 0,
    mirrored: false
},
{
    time: 1,
    type: 'town',
    shape: [[1,1,1],
        [1,1,0],
        [0,0,0]],
    rotation: 0,
    mirrored: false
},
{
    time: 1,
    type: 'farm',
    shape: [[1,1,0],
        [0,1,1],
        [0,0,0]],
    rotation: 0,
    mirrored: false
},
{
    time: 1,
```

```
    type: 'farm',
    shape: [[0,1,0],
            [1,1,1],
            [0,1,0]],
    rotation: 0,
    mirrored: false
},
{
    time: 2,
    type: 'water',
    shape: [[1,1,1],
            [1,0,0],
            [1,0,0]],
    rotation: 0,
    mirrored: false
},
{
    time: 2,
    type: 'water',
    shape: [[1,0,0],
            [1,1,1],
            [1,0,0]],
    rotation: 0,
    mirrored: false
},
```

```
{
    time: 2,
    type: 'forest',
    shape: [[1,1,0],
            [0,1,1],
            [0,0,1]],
    rotation: 0,
    mirrored: false
},
{
    time: 2,
    type: 'forest',
    shape: [[1,1,0],
            [0,1,1],
            [0,0,0]],
    rotation: 0,
    mirrored: false
},
{
    time: 2,
    type: 'water',
    shape: [[1,1,0],
            [1,1,0],
            [0,0,0]],
    rotation: 0,
```

```
    mirrored: false
  },
]
```

The page



The following things are displayed on the page:

- 11x11 grid as the map, showing the mountains and the dropped shapes
- The names and descriptions of randomly selected missions
- Time units remaining in the game
- Which season we are in and the game indicates which missions belong to which
- The number of points we have collected during the seasons

- Our total points and how many points we have earned for each mission.
- The item to place and the associated time period
- Rotate and mirror buttons

Help to solve the task

We recommend the same steps for the solution of the problem as those given in the lecture.

1. Designing the user interface
   - Create the page with static elements (e.g. HTML and CSS).
   - Pay special attention to the implementation of the square grid! Make sure that the elements are square, so that they fit on the page even on lower resolution displays.
   - How do you indicate the field elements? With a background colour, or a background image?
   - Will you have animations? Where, what kind? Get them ready!
2. Add behaviour
   1. The game logic
      - Data
        - What data do you need to run the game?
        - In other words: you should store data that you need to be able to draw the game at any moment!
        - What data is absolutely necessary? And what can you derive, calculate from them? Do not store the latter, always calculate them!
        - For example: the square grid can be represented by a matrix, in which you have to store the field elements in the cells. Store the items to be drawn, the missions (see above). Store the elapsed time. And what else?
      - Operations
        - What changes can occur in the game?
        - What does a rotation mean at the data level? Mirroring? A placement of a map element? Or initializing the game state?

        - What information do you need to extract from the data? For example, calculating missions. Determining the season from the stored time units?
    2. The event handlers
        - Which user actions trigger which events?
        - Where can you click in the interface?
        - What should you do then?
        - What data should be read from the interface?
        - And what changes should be made to the data?
        - How do you display the changes?
            - Do you modify the elements? (imperative)
            - Or do you replace them and make a redraw? (declarative)

Of course, you don't have to make these big blocks in one step. You can go step by step. For example:

1. Drawing a square grid
    - static prototype (e.g. HTML, CSS)
    - matrix on data level
    - generate HTML table from matrix (preparing declarative direction)
2. Placing map elements
    - static prototype (e.g. HTML, CSS)
    - at data level, store elements to be placed (see above prepared), shuffle
    - click on table cell to place, if possible
    - etc. (mirroring, rotating, on mouse-over draw the shapes, etc.)
3. Measuring time
4. Missions
5. Seasons

Design

The good-looking design is important. This doesn't necessarily mean making a fancy-looking page, but it does mean making sure that the layout looks good at 1024x768 resolution and above, and that the game board contains square cells. You can use minimalist design, you can create your own CSS with different background images and graphical elements, or you can use any CSS framework.

There are no requirements as to which technology (table, divs or canvas) you use to solve the task, and there are no strict expectations for the appearance and functionality. The point is that the above tasks should be recognisable and the game should be playable.

Your own work

The work submitted must be essentially your own, individual, independent intellectual product! This does not mean that you cannot, for example, use documentation or search the internet for ideas, but in the case of significant code sharing with other students or a large amount of internet sources, the submission may be rejected without any retake/improvement! (In such a case, the student(s) concerned will not be allowed to complete the course in any way this semester!)

In order to avoid matches, we also strongly request that you do NOT publish your solution on public repository, GitHub or any other platform at least until the submission deadline! In case of a match, we will not search for the source, but we will sanction all students equally, as forwarding or publishing the solution is also considered a study irregularity! (ELTE Student Requirements System, Special Section IK Faculty, § 377/A)

No JavaScript frameworks or libraries can be used to implement the application - for example, any level of jQuery, React, etc. is prohibited!

The completed assignment must be uploaded as a single .zip archive to the Canvas page of the course. The package must include all files needed to run and evaluate the assignment (e.g. HTML, CSS, media elements, JavaScript).

In addition to the application files, it is MANDATORY to submit the README.md file based on the template below, with the declaration and self-checklist included. At the time of submission, the student declares that he/she accepts the possible consequences in case of fraud and the self-checklist will speed up and improve the accuracy of the assessment. In the checklist, put an [X] in front of each task that you have at least partially solved!

If the uploaded archive does not/incorrectly/incompletely contain the declaration, the self-checklist or a required file, the submitted work will not be evaluated and will result in failure to complete the course! It is the student's responsibility to ensure that the submission is complete in time!

Scoring

20 points can be taken for solving the problem. There are minimum requirements without which the submission will not be accepted. Additional assignments may earn an additional 10 points. If you solve everything, you can get up to 30 points for the submission.

The conditions for obtaining a practice grade for a JavaScript submission are: meeting all the minimum requirements, i.e. obtaining at least 8 points (40%).

Remember that the deadline for the assignment is fixed, we do not accept late ones!

Minimal requirements (not accepted without them, 8 points)

- Square grid: After starting the game, a 11x11 map with the mountains in the right place is drawn (1 point)
- Placement: One of the map elements is randomly displayed with the corresponding time units (1 point)
- Placement: We can place the map element on the grid (anywhere) (2 points).
- Time: The game lasts up to 28 units of time, and by placing a map element down, you subtract the unit of time associated with that map element. (1 point)
- Mission: you can calculate the score of the mission "Borderlands" (1 point).
- End of game: for each mission, it calculates how many points have been scored for that mission (1 point)
- End of game: At the end of the game, after the 28 time units have elapsed, it calculates the score for the basic mission "Borderlands" and displays the number of points scored (1 point)

Normal requirements (12 points)

- Placement: You can place the map element correctly (2 points).
- Placement: The displayed map element can be rotated and placed in this way (1 point)
- Placement: The displayed map element can be mirrored and placed in this way (1 point)
- Mission: the mission "Edge of the forest" is displayed and can be scored (1 point)
- Mission: the mission "Sleepy valley" is displayed and can be scored (1 point)
- Mission: the mission "Watering potatoes" is displayed and can be scored (1 point)
- Season: the game is played over 4 seasons, each season lasts for 7 time units, the mission cards for each season are highlighted. (1 point)
- Season: At the end of each season, the end-of-season score is calculated from the corresponding mission cards and the game continues to the next season. (1 point)

- Mission: 1 extra point can be earned by completely encircling the mountains, which will be added to your score at the end of each season (or game) (1 point)
- End of game: at the end of the game, the total score over the four seasons is displayed (1 point).
- Good-looking appearance (1 point)

Extra requirements (10 points)

- Mission: Tree line (1 point)
- Mission: Watering canal (1 point)
- Mission: Wealthy town(1 point)
- Mission: Magicians' valley (1 point)
- Mission: Empty site (1 point)
- Mission: Terraced House (1 point)
- Mission: Odd silos (1 point)
- Mission: Rich countryside (1 point)
- Save: The game saves its state continuously to localStorage. When loading a page, if there is such a saved state, it is loaded from there, otherwise a new game is started. At the end of the game, the saved state is deleted (2 points).

README.md

n the README.md file, include the following statement (Replace the <> placeholders with your own data). In the same file, replace the space between each [ ] with an x for the subtasks that you have solved (even partially). Without a properly filled in README.md file, the solution will not be accepted!

<Student's name>

<Neptun code>

Web programming - assignment

This solution was submitted and created by the student above for the Web Programming course.

I declare that this solution is my own work. I did not copy or use it from a third party

solutions from third parties. I did not forward my solution to my fellow students, nor did I publish it.

Eötvös Loránd University Student Requirements System

(Organizational and Operational Regulations of ELTE, Volume II, § 74/C) states that as long as,

as long as a student has been working on the work - or at least a significant part of it - of another student

of another student's work as his or her own, it is a disciplinary offence.

The most serious consequence of a disciplinary offence is dismissal from the university.

Minimal requirements (not accepted without them, 8 points)

[ ] Square grid: After starting the game, a 11x11 map with the mountains in the right place is drawn (1 point)

[ ] Placement: One of the map elements is randomly displayed with the corresponding time units (1 point)

[ ] Placement: We can place the map element on the grid (anywhere) (2 points).

[ ] Time: The game lasts up to 28 units of time, and by placing a map element down, you subtract the unit of time associated with that map element. (1 point)

[ ] Mission: you can calculate the score of the mission "Borderlands" (1 point).

[ ] End of game: for each mission, it calculates how many points have been scored for that mission (1 point)

[ ] End of game: At the end of the game, after the 28 time units have elapsed, it calculates the score for the basic mission "Borderlands" and displays the number of points scored (1 point)

Normal requirements (12 points)

[ ] Placement: You can place the map element correctly (2 points).

[ ] Placement: The displayed map element can be rotated and placed in this way (1 point)

[ ] Placement: The displayed map element can be mirrored and placed in this way (1 point)

[ ] Mission: the mission "Edge of the forest" is displayed and can be scored (1 point)

[ ] Mission: the mission "Sleepy valley" is displayed and can be scored (1 point)

[ ] Mission: the mission "Watering potatoes" is displayed and can be scored (1 point)

[ ] Season: the game is played over 4 seasons, each season lasts for 7 time units, the mission cards for each season are highlighted. (1 point)

[ ] Season: At the end of each season, the end-of-season score is calculated from the corresponding mission cards and the game continues to the next season. (1 point)

[ ] Mission: 1 extra point can be earned by completely encircling the mountains, which will be added to your score at the end of each season (or game) (1 point)

[ ] End of game: at the end of the game, the total score over the four seasons is displayed (1 point).

[ ] Good-looking appearance (1 point)

Extra requirements (10 points)

[ ] Mission: Tree line (1 point)

[ ] Mission: Watering canal (1 point)

[ ] Mission: Wealthy town(1 point)

[ ] Mission: Magicians' valley (1 point)

[ ] Mission: Empty site (1 point)

[ ] Mission: Terraced House (1 point)

[ ] Mission: Odd silos (1 point)

[ ] Mission: Rich countryside (1 point)

[ ] Save: The game saves its state continuously to localStorage. When loading a page, if there is such a saved state, it is loaded from there, otherwise a new game is started. At the end of the game, the saved state is deleted (2 points).