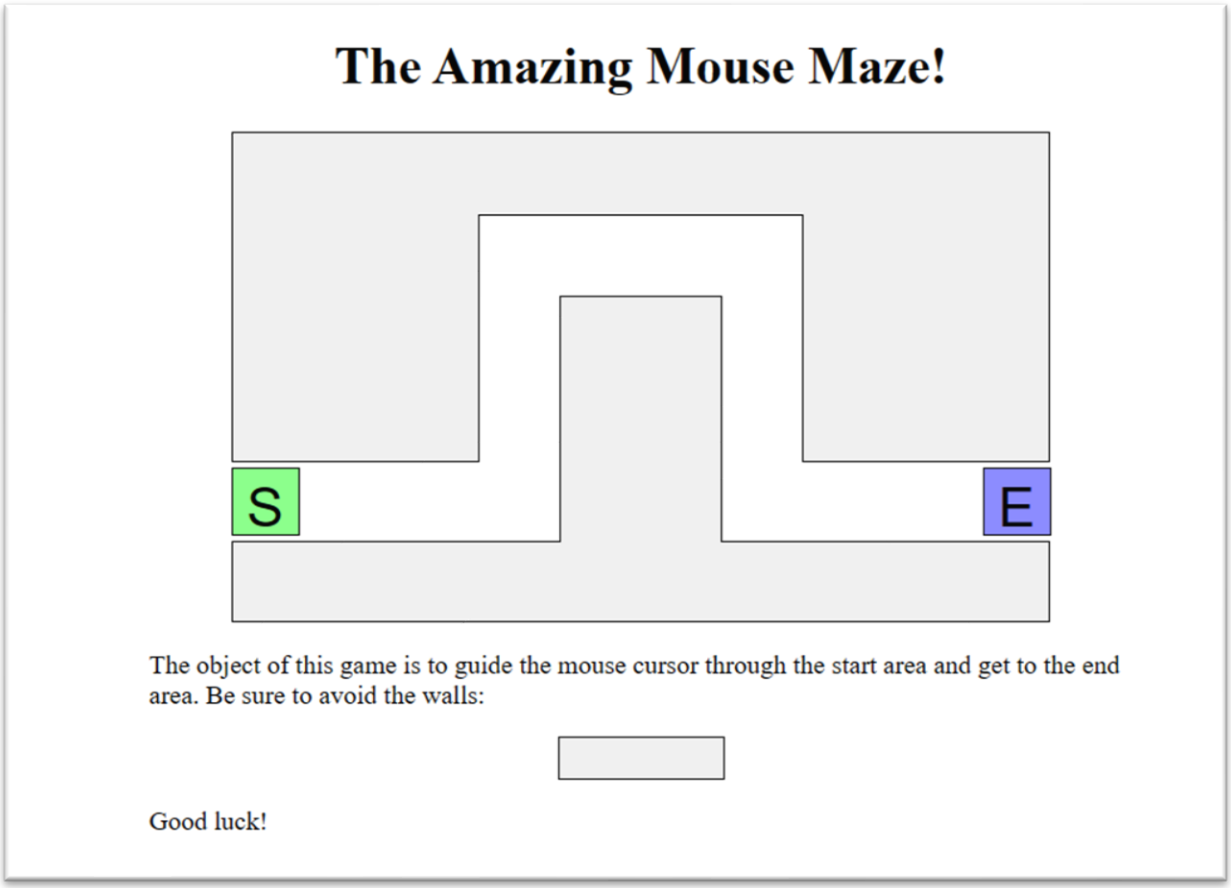
Lab2.Part2 – JS, DOM & Events

The theme of this lab is that we'll be writing a page with a simple "maze" that the user must navigate with the mouse.
The difficulty is not in finding the exit, because the maze is trivial, but rather in having the dexterity to move the mouse through the maze without touching any of the walls. When the user's mouse cursor touches one of the maze walls, the walls turn **red**, and a `"You lose"` message is shown. Touching the Start button with the mouse will remove the red coloring from the maze walls.



You're given files **`Lab2_maze.html`** and **`Lab2_maze.css`**. You should write a script file **`maze.js`** in the same directory as these files that provides all event handling and behavior to make the maze work as specified below. The maze walls are implemented as 5 **`div`** elements on the page. Our provided CSS code already positions these div elements into their proper places on the page. The relevant section of the **`Lab2_maze.html`** code in the page's body that contains these elements is the following:

```
<div id="maze">
 <div id="start">S</div>
 <div class="boundary" id="boundary1"></div>
 <div class="boundary"></div>
 <div class="boundary"></div>
 <div class="boundary"></div>
 <div class="boundary"></div>
 <div id="end">E</div>
</div>
```

- **Runnable solution** to this lab (so you can see how your page is supposed to look and work)

Exercise 1: Single boundary turns red

The first task is to write event-handling code so that when the user moves the mouse onto a single one of the maze's walls, that wall will turn red. Write your JavaScript code, without modifying the maze.html page.

The following are more detailed suggested steps for solving this exercise.

Exercise 2: All boundaries glow red on hover

For your next task, make it so that all walls of the maze turn red when the mouse enters any one of them.

Exercise 3: Alerts on successful completion of maze

Your next task is to make it so that if the user reaches the end of the maze, a congratulatory "You win!" alert message appears.

Exercise 4: Restartable maze

One annoying thing you may be noticing as you test the maze so far is that it can't easily be reset to try again. So, our next task will be to make it so that when the user clicks the mouse on the Start square, the maze state will reset. That is, if the maze boundary walls are red, they will all return to their normal color, so that the user can try to get through the maze again.

Exercise 5: Write your solution and commit it in GitHub

Verify your JavaScript code. Then Upload it to your GitHub CAW_Labs repository.

Challenges:

Exercise 6: On-Page status updates

Alert boxes are generally annoying. Instead of an alert box, make the "You win" and "You lose" messages appear inside the page itself. Place them into the (initially empty) h2 element on the page with an id of status.

Exercise 7: Disallow cheating

It's currently too easy to cheat: Just move your mouse onto the start square, around the outside of the maze, and onto the end square. Fix this by making it so that if the user moves the mouse anywhere outside the maze after clicking the Start area, the walls will light up red and the player will lose the game.

- **Deadline:**

At the end of Lab session (no later than Saturday, October 28 at 23:59)
To: adil.chekati@univ-constantine2.dz

Drive for ***Web Application Design*** *(Conception d'App. Web)*'s materials

SCAN ME!