02393 C++ Programming Exercises

Assignment 4

To be handed in via Autolab — https://autolab.compute.dtu.dk/courses/02393-E23/assessments

1 The Maze

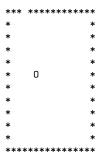
Consider the following code that creates an (empty) version of the maze seen during the lecture, having 12×16 tiles:

```
typedef enum { wood, stone } material;
struct tile {
 int x, y;
 bool isWall;
 material type;
};
#define NROWS 12
#define NCOLS 16
int main() {
 tile playground[NROWS][NCOLS];
 for (int i = 0; i < NROWS; i++) {</pre>
    for (int j = 0; j < NCOLS; j++) {
      playground[i][j].x = j;
      playground[i][j].y = i;
      playground[i][j].isWall = (j==0 || i==(NROWS-1) || (i==0 && j!=3) || j==(NCOLS-1));
      if (playground[i][j].isWall) {
        playground[i][j].type = stone;
      } else {
        playground[i][j].type = wood;
    }
 }
}
```

We want to make a mini text-based game out of this.

- Use two variables x and y to represent the position of a player in the playground, and let's have initially x = y = 5.
- Write a function that uses x and y above, and displays the current situation on the standard output (cout), by writing:
 - "*" for every tile that is a wall
 - "" (a single space) for every tile that is empty
 - "0" for the tile where the player is located

(The function can assume that the player is always within the playground.) Thus, the initial state is displayed as follows: (see next page)



- Read a value of type char from the standard input (cin), representing a user command:
 - if the character is 'q', the game ends (without producing any further output);
 - if the character is '1' (lowercase L), then the player should make one step to the left
 if possible, i.e., if the target tile exists and is not a wall. If it is not possible, then the
 player remains in the same position;
 - similarly, if the character is 'r' for right, 'u' for up, or 'd' for down, the player should be moved one step in the corresponding direction, if possible.

After each step, the playground is displayed again and the user can enter the next command (until 'q' is given).

2 The Maze With Dynamic Allocation

This is a variation of the previous exercise where the size of the maze is specified by the user, at run-time. Therefore, the program should:

- \bullet read two integers n and m representing the number of rows and columns of the maze, respectively;
- dynamically allocate a maze of size $n \times m$;
- start with the player's initial position at coordinates $x = \lfloor \frac{m}{2} \rfloor$ and $y = \lfloor \frac{n}{2} \rfloor$;
- display the maze and the player position, and
- read user commands and display the results, until 'q' is given just as in the previous exercise.