When doing this assignment i based it off of my quiz from assignment 3, as a lot of the base structure could be done the same way. I also decided to recreate the quiz example given in the assignment description, but didn't put any effort into the visible quality of the quiz to focus on functionality.

I early on figured that I could reuse the array structure for the questions and answers from uppgift 3, so that all I had to do was create an input field for the question, answer, what kind of questions the user wanted, and buttons to move the game along. I found the input function for html and used the radio type to allow the user to select the kinds of questions they wanted, just like the example image. To only allow one radio button to be selected at one time to prevent logical errors when selecting a question I assigned them the same name, which means that only one radio button is allowed to be selected at once. I also found that input had a button type which I used to allow the player to get a new question and submit their answer, as well as the text and number type, which are editable text fields that take in text respectively numbers, which I used as the problem field and the input field for the user. I could've used any other form of text field for the problem, but to stay true to the example image I used an input field. I also assigned it the text type as I wanted it to be able to include and manage non numerical symbols as well.

When creating the logic I started by restructuring my questions array to incorporate the option to select between different question sets, which meant creating an array of arrays that contained the question types, their questions and their corresponding answers. I then created a function that would be activated whenever the "new questions" button was pressed, that would check which radio button was selected using an if function, that then loaded up the questions of the corresponding type into a new array. I then selected a random question from that array using math.floor() and math.random(), multiplying math.random() with the length of the questions array to allow a random question from the full selected question set to be selected. I then updated the input field for the math problem with the randomly selected question.

To allow the user to submit an answer and get feedback on their math skills I created a function that gets called whenever the submit button was pressed, in which I declared a variable to hold the answer to the current question and then created logic to compare the given solution to the question with the correct answer. To make the player know whether or not they were correct I used an if and else that triggers an alert with appropriate feedback to the player's solution.

In the current system questions can get repeated due to the nature of math.random(). With more questions per dataset repetition is less likely to occur, and to create the logic to prevent the repetition felt too complicated to create for the small scope of the assignment. The user can also edit the question field which doesn't change the correct answer in which the input is checked against as the answer is currently stored in the dataset as a redundancy, and that the comparison logic doesn't take the question to calculate an answer from. This could be done to allow for the user to write their own questions and to simplify the dataset storage by simply changing the questions from being stored as text strings to being stored as statements, and by changing the comparison logic to take in the question statement rather than the answer value. This too felt a bit redundant for the scope of the project.