1. Initialize starting points as variables: current active button (current = 0), score (score = 0), velocity (pace = 2000)
2. Make an array of buttons using betElementById. Four pieces in this task.
3. For each button make onclick event which calls a function that will increase the score and velocity ( buttons[0].onclick = function() {clicked(0)}; ... buttons[4].onclick = function() {clicked(4)}; ). In fact, velocity increase is provided by decreasing of “pace”, which is used in setTimeout function. It gives an effect of velocity increase.
4. The function principles: if one clicks the button and the button index “i” does not equal to “current” value, myStopFunction is called. Otherwise, score becomes higher and pace becomes lower.
5. startGame function assigns “next” with the value returned by startNew function.
6. In startNew function “next” gets random number from an array of buttons using Math.floor-based function. If “next” value is not equal to “current” value, it returns this value to startGame function. Overwise, startNew calls itself once again.
7. After startGame function got next variable value from startNew function, it changes style of button with index equals to “next” value ( target color “tomato” ), buttons with index equals to “current” becomes default color ( “khaki” ).
8. After that, “current” is assignesd with the value of “next”, because the next step is to make the function startGame to work once repeatedly until wrong button is pressed. For this aim, one needs to use setTimeout function, calling startGame function with interval equals to “pace”.
9. myStopFunction stops the game, zeros the timer ( timer = setTimeout(startGame, pace) ), makes all buttons of tomato color and show a message with final score. Button “close” reloads the page.