ДОДАТОК А КОД ПРОГРАМИ

index.html

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<meta http-equiv="X-UA-Compatible" content="IE=edge">
	<meta name="viewport" content="width=device-width, initial-scale=1">
	<title>example</title>
	<!-- P5 libraries -->
	<script src="libraries/p5.js"></script>
	<script src="libraries/p5.dom.js"></script>
	<script src="libraries/p5.sound.js"></script>
	<script src="https://unpkg.com/ml5@latest/dist/ml5.min.js"></script>

	<script src="10_bit_nth_round_cypher.js"></script>
	<script src="10_10_test.js"></script>
	<style>
		body {
			margin: 0;
			padding: 0;
			overflow: hidden;
		}
		#suda {
			display: flex;
			flex-direction: column;
		}
	</style>
</head>
<body>
	<div id="suda"></div>
</body>

</html>
```

10_10_test.js

```
let parent;

let data = [{}];

let model;

let counter;
let success;
```

```
let meanErr;

let res = [];

let round = 11;
let hidden = 256;

async function setup() {
  parent = document.getElementById("suda");

  tryRound();
}

async function finishedTraining() {
  classify();
}

async function classify() {
  // for (let i = 0; i < (mes.length * 0.3).toFixed(); i++) {
  for (let i = 0; i < data.length; i++) {
    let input = {
      x1: (data[i].c >> 9) & 1,
      x2: (data[i].c >> 8) & 1,
      x3: (data[i].c >> 7) & 1,
      x4: (data[i].c >> 6) & 1,
      x5: (data[i].c >> 5) & 1,
      x6: (data[i].c >> 4) & 1,
      x7: (data[i].c >> 3) & 1,
      x8: (data[i].c >> 2) & 1,
      x9: (data[i].c >> 1) & 1,
      x10: data[i].c & 1,
    };

    await model.predict(input, handleResults);
  }

  let mean = 0;
  for (let i = 0; i < meanErr.length; i++) {
    mean += meanErr[i];
  }
  console.log(`mean errors: ${mean / meanErr.length}`);
  console.log(`${success}/${mes.length}`);
  let res = {
    roundAmount: round,
    hiddenLayers: hidden,
    error: `${success}/${mes.length}`,
    meanError: mean / meanErr.length,
  };

  writeDown(res, parent);
  round++;
  if (round !== 11) {
    if (hidden === 1024) {
      hidden = 256;
```

```
            round++;
            data.length = 0;
            tryRound();
          } else {
            hidden += 256;
            tryModel();
          }
        }
      }

      async function handleResults(error, result) {
        if (error) {
          console.error(error);
          return;
        }

        let a = 0;
        for (let i = 0; i < 10; i++) {
          a |= Math.round(result[i].value) << (9 - i);
        }

        //checking for 10 bits

        if (a === data[counter].m) success++;
        else {
          let er = 0;
          for (let i = 0; i < 10; i++) {
            if (Math.round(result[i].value) !== ((data[counter].m >> (9 - i)) & 1))
              er++;
          }

          meanErr.push(er);

          let n = [];
          let cyph = [];
          for (let i = 0; i < 10; i++) {
            n.push(Math.round(result[i].value));
            cyph.push((data[counter].m >> (9 - i)) & 1);
          }
        }

        counter++;
      }

      function writeDown(data, parent) {
        let a = document.createElement("a");
        a.innerText = "download";
        parent.appendChild(a);

        a.onclick = function () {
          let text = "";
          text += `cypher rounds: ${data.roundAmount}\nhidden layers:
${data.hiddenLayers}\n\nerrors: ${data.error}\nmean error: ${data.meanError}`;
          let csvData =
```

```
            "data:application/txt;charset=utf-8," + encodeURIComponent(text);
          this.href = csvData;
          this.target = "_blank";
          this.download =
            "Perceptron_" + data.roundAmount + "r_" + data.hiddenLayers + "l.txt";
        };

        // console.log(`cypher rounds: ${dataMas.roundAmount}\nhidden layers:
${dataMas.hiddenLayers}\nerrors: ${dataMas.errorNumbers}/256\neducation time:
${dataMas.educationTime}s\nlearning rate: ${dataMas.learningRate}\n\n\n`);
      }

      async function tryRound() {
        start(round);

        for (let i = 0; i < mes.length; i++) {
          data[i] = new Object();
          data[i].m = mes[i];
          data[i].c = perm[i];
        }

        tryModel();
      }

      async function tryModel() {
        counter = 0;
        success = 0;
        meanErr = [];

        res = [];

        const options = {
          task: "regression",
          learningRate: 0.0009765625,
          debug: true,
          layers: [
            {
              type: "dense",
              units: 10,
              activation: "relu",
            },
            {
              type: "dense",
              units: hidden,
              activation: "relu",
            },
            {
              type: "dense",
              units: 10,
              activation: "relu",
            },
          ],
        };
```

```
  model = ml5.neuralNetwork(options);

  for (let i = (data.length * 0.3).toFixed(); i < data.length; i++) {
    // for (let i = 0; i < mes.length; i++) {
    const inputs = {
      x1: (data[i].c >> 9) & 1,
      x2: (data[i].c >> 8) & 1,
      x3: (data[i].c >> 7) & 1,
      x4: (data[i].c >> 6) & 1,
      x5: (data[i].c >> 5) & 1,
      x6: (data[i].c >> 4) & 1,
      x7: (data[i].c >> 3) & 1,
      x8: (data[i].c >> 2) & 1,
      x9: (data[i].c >> 1) & 1,
      x10: data[i].c & 1,
    };

    const outputs = {
      y1: (data[i].m >> 9) & 1,
      y2: (data[i].m >> 8) & 1,
      y3: (data[i].m >> 7) & 1,
      y4: (data[i].m >> 6) & 1,
      y5: (data[i].m >> 5) & 1,
      y6: (data[i].m >> 4) & 1,
      y7: (data[i].m >> 3) & 1,
      y8: (data[i].m >> 2) & 1,
      y9: (data[i].m >> 1) & 1,
      y10: data[i].m & 1,
    };

    model.addData(inputs, outputs);
  }

  model.normalizeData();

  const trainingOptions = {
    epochs: 1300,
    batchSize: 32,
  };

  await model.train(trainingOptions, finishedTraining);
}
```

10_bit_nth_round_cypher.js

```
function randomInteger(min, max) {
        // случайное число от min до (max+1)
        let rand = min + Math.random() * (max + 1 - min);
        return Math.floor(rand);
}

function reverseBoxes() {

        for (let i = 0; i < s_box1.length; i++) {
```

```
            s_box1_inv[s_box1[i]] = i;
            s_box2_inv[s_box2[i]] = i;
        }

}

const MAX = 1023;
let s_box1 = [];
let s_box2 = [];
let keyMax = 1023;
let key1, key2;
let s_box1_inv = [];
let s_box2_inv = [];

let mes = [];
let perm = [];
let mes1 = [];

function shuffleArray(array) {
    for (let i = array.length - 1; i > 0; i--) {
        let j = Math.floor(Math.random() * (i + 1));

        let temp = array[i];
        array[i] = array[j];
        array[j] = temp;
    }
}

async function start(rounds) {

    for (let i = 0; i <= MAX; i++) {
        mes[i] = i;
    }

    // console.log(`mes_before: ${mes}`);
    shuffleArray(mes);
    // console.log(`mes_after: ${mes}`);

    for (let i = 0; i <= (MAX >> 5); i++) {
        s_box1[i] = i;
        s_box2[i] = i;
    }

    shuffleArray(s_box1);
    shuffleArray(s_box2);
    // console.log(`s_box1: ${s_box1}`);
    // console.log(`s_box2: ${s_box2}`);

    key1 = randomInteger(0, MAX >> 5)
    key2 = randomInteger(0, MAX >> 5);
    console.log(`key1: ${key1}`);
    console.log(`key2: ${key2}`);

    reverseBoxes();
```

```
		nth_round_encryption(rounds);
		decryption(rounds);
	}

	function nth_round_encryption(rounds) {

		perm = mes.slice();

		// console.log(`perm: ${perm}`);

		for (let c = 0; c < rounds; c++) {
			for (let i = 0; i < mes.length; i++) {

				let a = ((key1 ^ s_box1[perm[i] >> 5]) << 5) | (key2 ^ s_box2[perm[i]
& 0x1f]);

				//циклический сдвиг влево на 7 бит
				let b = (a >> 3);
				a = ((a << 7) % 0x400) | b;

				perm[i] = a;

			}
		}

		// console.log(perm);
	}

	function decryption(rounds) {

		mes1 = perm.slice();
		// console.log(mes[0]);
		for (let c = 0; c < rounds; c++) {
			for (let i = 0; i < mes.length; i++) {

				//циклический сдвиг вправо на 7 бит

				let a = mes1[i];

				let b = (a & 0x7f) << 3;

				a = (a >> 7) | b;

				a = ((s_box1_inv[key1 ^ (a >> 5)] << 5) % 0x400) | s_box2_inv[key2
^ (a & 0x1f)];

				mes1[i] = a;
			}
		}
		console.log(function () {
			for (let i = 0; i < mes1.length; i++) {
				if (mes1[i] !== mes[i]) return false;
			}
```

```
                return true;
            }());
}
```

№ док.	Позначення	Найменування	Дод. відом.
1		Текстові документи	
	ГЮІК.ХХХХХ1412Ст.09ПЗ	Пояснювальна записка	73 с.
2		Графічні документи	
		Презентаційний матеріал	15 слайдів
3		Інші документи	
		Електронна версія ПЗ	1
		Відзив керівника	1 с.
		Рецензія	1 с.

Изм.	Лист	№ докум.	Підпис	Дата	ГЮІК. ХХХХХ1412Ст.09ВД		
Розроб.		Левченко Р. Р.			Криптоаналіз простого шифру заміщення-перестановки за допомогою штучної нейронної мережі	Лит.	Лист 1 / Листів 1
Перевір.		Руженцев В. І.				Т	
Н.контр.		Конєва Н.Ф.				ХНУРЕ Кафедра БІТ	
Затверд.		Халімов Г.З.					