МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра програмної інженерії та інформаційних технологій управління

Звіт з “Sprint 01”

з дисципліни «Чисельні методи»

Виконав:

ст. гр. КН-219б

Пономаренко В.С.

Перевірив:

проф. каф. ПІІТУ

Нікуліна О. М.

Харків

2021

ТЕМА: ЗНАХОДЖЕННЯ НУЛІВ ФУНКЦІЇ

ЗАВДАННЯ НА ЛАБОРАТОРНУ РОБОТУ

“

**Task 1**

Написать программу метода Крамера, программу оформить в виде функции. Программы можно писать на любом языке программирования. Входные параметры: А – матрица коэффициентов СЛАУ, В – вектор-столбец свободных членов. Выходные параметры: Х – вектор-столбец корней СЛАУ.

**Task 2**

Написать программу метода Гаусса, программу оформить в виде функции. Программы можно писать на любом языке программирования. Входные параметры: А – матрица коэффициентов СЛАУ, В – вектор-столбец свободных членов. Выходные параметры: Х – вектор-столбец корней СЛАУ.

**Task 3** Написать программу метода Зейделя, программу оформить в виде функции. Программы можно писать на любом языке программирования. Входные параметры: А – матрица коэффициентов СЛАУ, В – вектор-столбец свободных членов. Выходные параметры: Х – вектор-столбец корней СЛАУ.

**Task 4**

Написать программу метода Гаусса-Жордана, программу оформить в виде функции. Программы можно писать на любом языке программирования. Входные параметры: А – матрица коэффициентов СЛАУ, В – вектор-столбец свободных членов. Выходные параметры: Х – вектор-столбец корней СЛАУ.

**Task 5**

Написать программу метода Якоби, программу оформить в виде функции. Программы можно писать на любом языке программирования. Входные параметры: А – матрица коэффициентов СЛАУ, В – вектор-столбец свободных членов. Выходные параметры: Х – вектор-столбец корней СЛАУ.

**Task 6**

Написать программу реализующую проверку найденных корней СЛАУ, программу оформить в виде функции. Входные параметры: Х – вектор-столбец корней СЛАУ. Выходные параметры: flag – булевская переменная (true\false).

**Task 7**

Реализовать графический интерфейс пользователя программного продукта. Предусмотреть возможность выбора любого метода, а также чтения начальных данных из текстовых файлов и с клавиатуры. Содержание текстовых файлов: n – количество уравнений (неизвестных), А – матрица коэффициентов СЛАУ, В – вектор-столбец.

**Task 8**

Доработать и протестировать разработанный программный продукт. В программном продукте должно быть учтены особенности применения разработанных методов:

– размерность СЛАУ (например, метод Крамера не более 4);

– проверять невырожденность матрицы A.

”

МЕТА РОБОТИ

Метою виконання лабораторної роботи є написання програми для вирішення СЛАУ різними методами.

2 ОПИСАННЯ РОЗРОБЛЕНОГО ЗАСТОСУНКУ

Програма була розроблена з використанням таких технологій:

- JavaScript;

- HTML;

- CSS;

Опис розробленого програмного застосунку я почну з “Task 07”, адже саме це завдання є підсумком усіх інших.

Для початку роботи з програмою вам потрібно, запустити Web сервер на вашому комп’ютері з кореневою папкою “t07”. Будь-які тестування рекомендую проводити в браузері “Google Chrome”.

Стартова сторінка проекту має такий вигляд, зображений на рисунку 1.

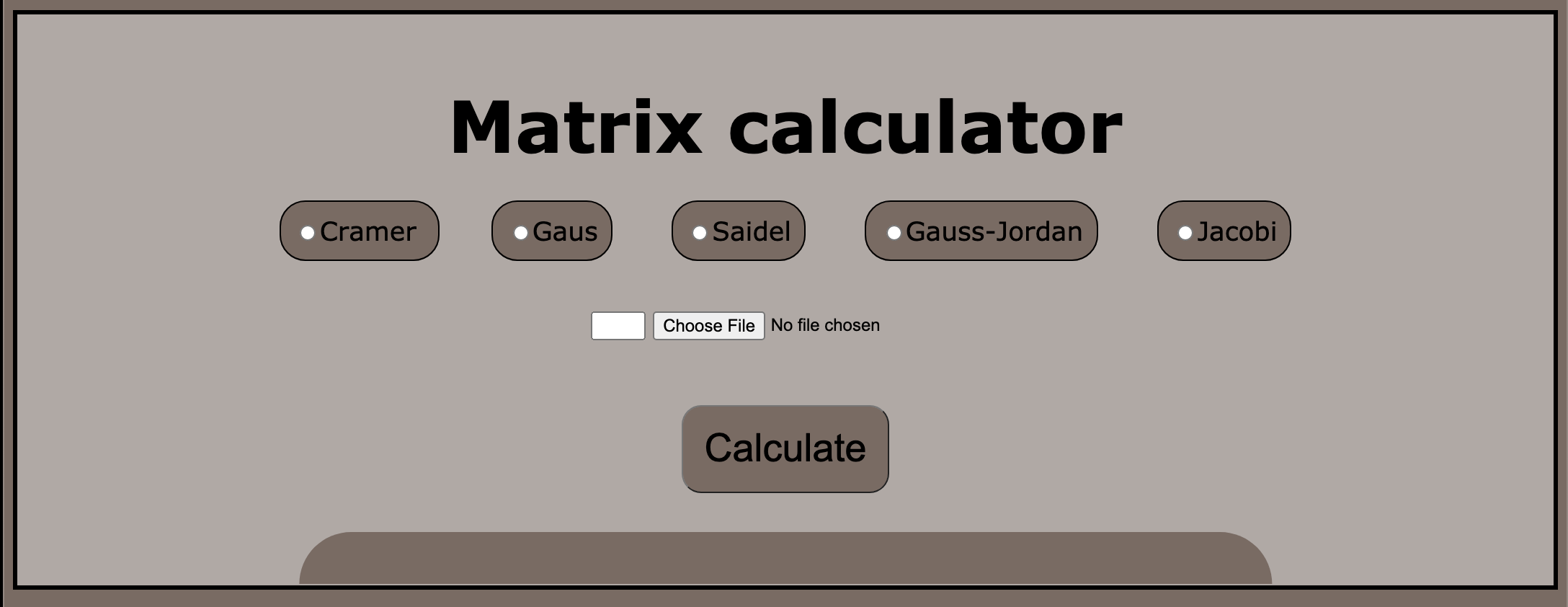


Рисунок 1 — Головна сторінка

З початку вам вибрати метод розв’язання СЛАУ.

1) Метод Крамера, реалізувати цей метод, було завданням 01. Реалізацію методу ви можете знайти в папці «t01/Cramer.js» або «t07/js/Cramer.js»

Код реалізації методу Крамера:

*function formMatrix (slau, type)*

*{*

*for (var matrix = [], j = 0, J = slau.length; j < J; j++)*

*{*

*matrix [j] = slau [j].slice (0, slau [j].length - 1);*

*if (type) matrix [j] [type - 1] = slau [j] [slau [j].length - 1];*

*}*

*return matrix;*

*}*

*function cramer (arr1, arr2) {*

*slau = arr1.slice();*

*for (let index = 0; index < arr2.length; index++) {*

*slau[index].push(arr2[index]);*

*}*

*var d0 = Determinant(formMatrix (slau, 0));*

*console.log("Determinant:" + d0);*

*if (d0 == 0) {*

*throw new UserException("Incorect matrix for this method.");*

*}*

*for (var roots = [], j = 0, J = slau.length; j < J; j++) {*

*roots [j] = Determinant(formMatrix (slau, j + 1)) / d0;*

*}*

*return roots;*

}

2) Метод Гауса, реалізувати цей метод, було завданням 02. Реалізація методу ви можете знайти в папці «t01/Gausa.js» або «t07/js/Gausa.js»

Код реалізації методу Гауса:

*function Iteration(iter\_item, n, m) { //Функция итеррация*

*for(iter\_item=0;iter\_item<(n-1);iter\_item++) { //Цикл выполнения итерраций*

*if (m[iter\_item][iter\_item] == 0) SwapRows(iter\_item,size,m); //Проверка на ноль*

*for(j=n; j>=iter\_item; j--) {*

*m[iter\_item][j] /= m[iter\_item][iter\_item]; //Делим строку i на а[i][i]*

*}*

*for(i=iter\_item+1; i<n; i++) { //Выполнение операций со строками*

*for(j=n;j>=iter\_item;j--) {*

*m[i][j] -= m[iter\_item][j] \* m[i][iter\_item];}}}}*

*function SwapRows(iter\_item, n, m) {*

*for(i=iter\_item+1;i<n;i++) {*

*if(m[i][iter\_item] !== 0) {*

*for(j=0;j<=n;j++) {*

*k = m[i-1][j];*

*m[i-1][j] = m[i][j];*

*m[i][j] = k;*

*}*

*}*

*}*

*}*

*function gausa(arr1, arr2) {//Функция поиска и вывода корней*

*n = arr2.length;*

*m = createCopy(arr1);*

*console.log("le:" + arr2.length);*

*for (let index = 0; index < arr2.length; index++) {*

*m[index].push(arr2[index]);*

*}*

*size = n;*

*Iteration(n, size, m);*

*var l=new Array(size); //Массив ответов*

*l[n-1] = m[n-1][n]/m[n-1][n-1];*

*for(i=n-2;i>=0;i--) {*

*k=0;*

*for(j=n-1;j>i;j--) {*

*k = (m[i][j]\*l[j]) + k;}*

*l[i] = m[i][n] - k;}*

*for (let index = 0; index < l.length; index++) {*

*if (!isFinite(l[index])) {*

*throw new UserException("Incorect matrix for this method.");}}*

*return l;*

*}*

3) Метод Зейделя, реалізувати цей метод, було завданням 03. Реалізація методу ви можете знайти в папці «t01/Saidel.js» або «t07/js/Saidel.js»

Код реалізації методу Зейделя:

*/\** [*https://studfile.net/preview/2377683/page:4/*](https://studfile.net/preview/2377683/page:4/) *\*/*

*function Saidel(a,b) {*

*let size = b.length;*

*x = new Array(b.length)*

*x.fill(0);*

*for (let i = 0; i < 1000; i++) {*

*x = Saidel\_func(a, x ,b)*

*}*

*return x;*

*}*

*function Saidel\_func(a, x ,b) {*

*let n = b.length;*

*for (let j = 0; j < n; j++) {*

*d = b[j]*

*for (let i = 0; i < n; i++) {*

*if(j != i) {*

*d-=a[j][i] \* x[i];*

*}*

*}*

*x[j] = d / a[j][j]*

*}*

*return x;*

*}*

4) Метод Якобі, реалізувати цей метод, було завданням 04. Реалізація методу ви можете знайти в папці «t01/Jacobi .js» або «t07/js/Jacobi.js»

Код реалізації методу Якобі:

*function func(row, number, x) {*

*let result = 0;*

*for (let i = 0; i < row.length - 1; i++) {if (i == number) {*

*result += row[i];} else {*

*result += row[i] \* x[i] \* (-1);}}*

*result = result / row[row.length - 1]*

*return result*

*}*

*function all\_true(e, eps) {*

*for (let i = 0; i < e.length; i++)*

*if (e[i] <= eps)*

*return false*

*return true*

*}*

*function Jacobi(matrix, b) {*

*let n = matrix.length*

*let x\_tmp = new Array(n)*

*let x = new Array(n)*

*let e = new Array(n)*

*let eps = 0.000001*

*for (let i = 0; i < n; i++) {*

*x[i] = 0;x\_tmp[i] = 0;b[i] = +b[i];*

*}*

*for (let i = 0; i < n; i++) {*

*matrix[i].push(b[i])*

*}*

*for (let i = 0; i < n; i++) {*

*let tmp = matrix[i][i];*

*matrix[i][i] = matrix[i][n];*

*matrix[i][n] = tmp;*

*}*

*let counter = 0;*

*do {*

*for (let i = 0; i < n; i++) {*

*x[i] = func(matrix[i], i, x\_tmp)*

*e[i] = Math.abs(x\_tmp[i] - x[i])*

*}*

*for (let i = 0; i < n; i++) {x\_tmp[i] = x[i];}counter++;*

*if (counter > 10000) {*

*throw new UserException("Incorect matrix for this method.");*

*}*

*} while(all\_true(e, eps));*

*return x;*

*}*

4) Метод Гауса-Джордана, реалізувати цей метод, було завданням 05. Реалізація методу ви можете знайти в папці «t01/Gaus-Seidel..js» або «t07/js/Gaus-Seidel..js»

Код реалізації методу Гауса-Джордана:

function Gauss\_Jordan(matrix, b) {

let x = [];

let n = matrix.length;

b = [...b];

matrix = createCopy(matrix);

for (let i = 0; i < n; i++) {

matrix[i].push(parseFloat(b[i]));

}

for (let k = 0; k < n; k++) {

for (let i = k + 1; i < n; i++) {

let mu = parseFloat(matrix[i][k] / matrix[k][k]);

for (let j = 0; j < n + 1; j++) {

matrix[i][j] -= parseFloat(matrix[k][j]) \* parseFloat(mu);

}

b[i] -= b[k] \* mu;

}

}

for (let i = 0; i < n; i++) {

let mu = matrix[i][i]

for (let j = 0; j < n + 1; j++)

matrix[i][j] = parseFloat(matrix[i][j]) / parseFloat(mu)

}

for (let a = n - 1; a > 0; a--)

for (let i = 0; i < a; i++) {

let k = matrix[i][a]

for (let j = 0; j <= n; j++)

matrix[i][j] = matrix[i][j] - k \* matrix[a][j]

}

for (let i = 0; i < n; i++)

x.push(matrix[i][matrix[i].length - 1])

for (let index = 0; index < x.length; index++) {

if (!isFinite(x[index])) {

throw new UserException("Incorect matrix for this method.");

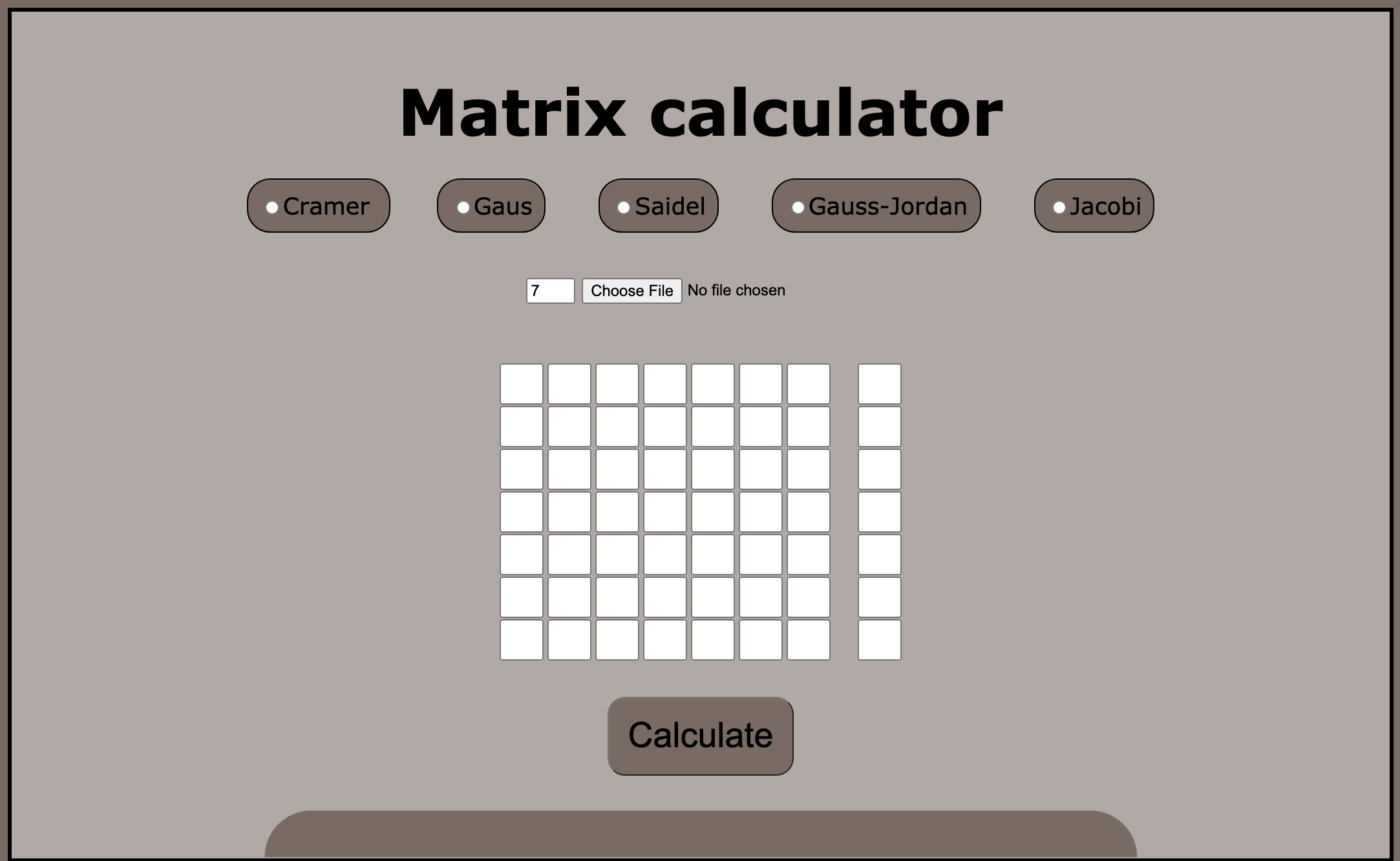
}

}

return x;

}

Далі потрібно вибрати розмір вашої СЛАУ, та заповнити її, або взяти інформацію з файлу, в цьому випадку, матриця заповнится сама (рис. 2 — 4)



Рисукок 2 — Вибір розміру СЛАУ

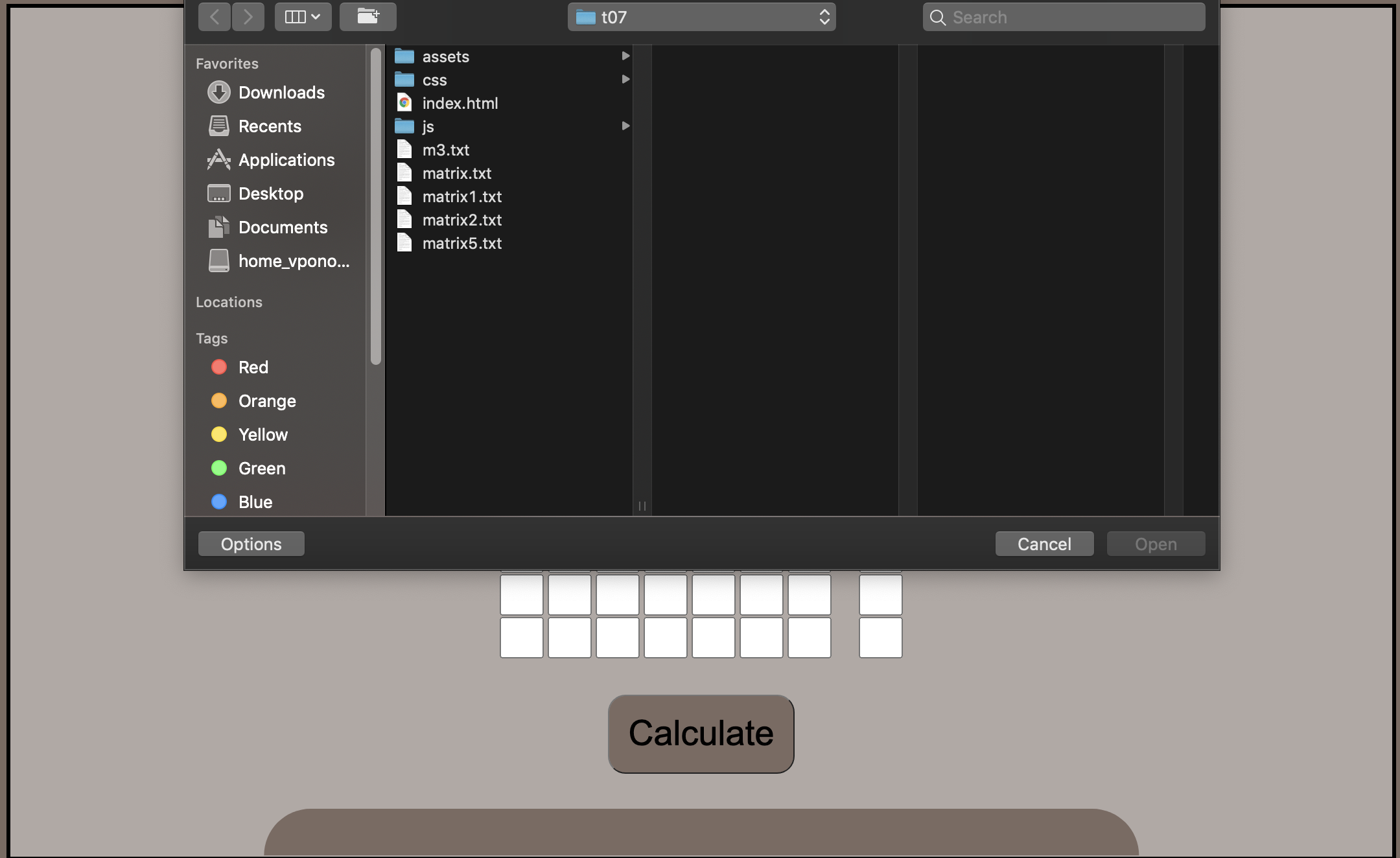


Рисунок 3 — Завантаження файлу

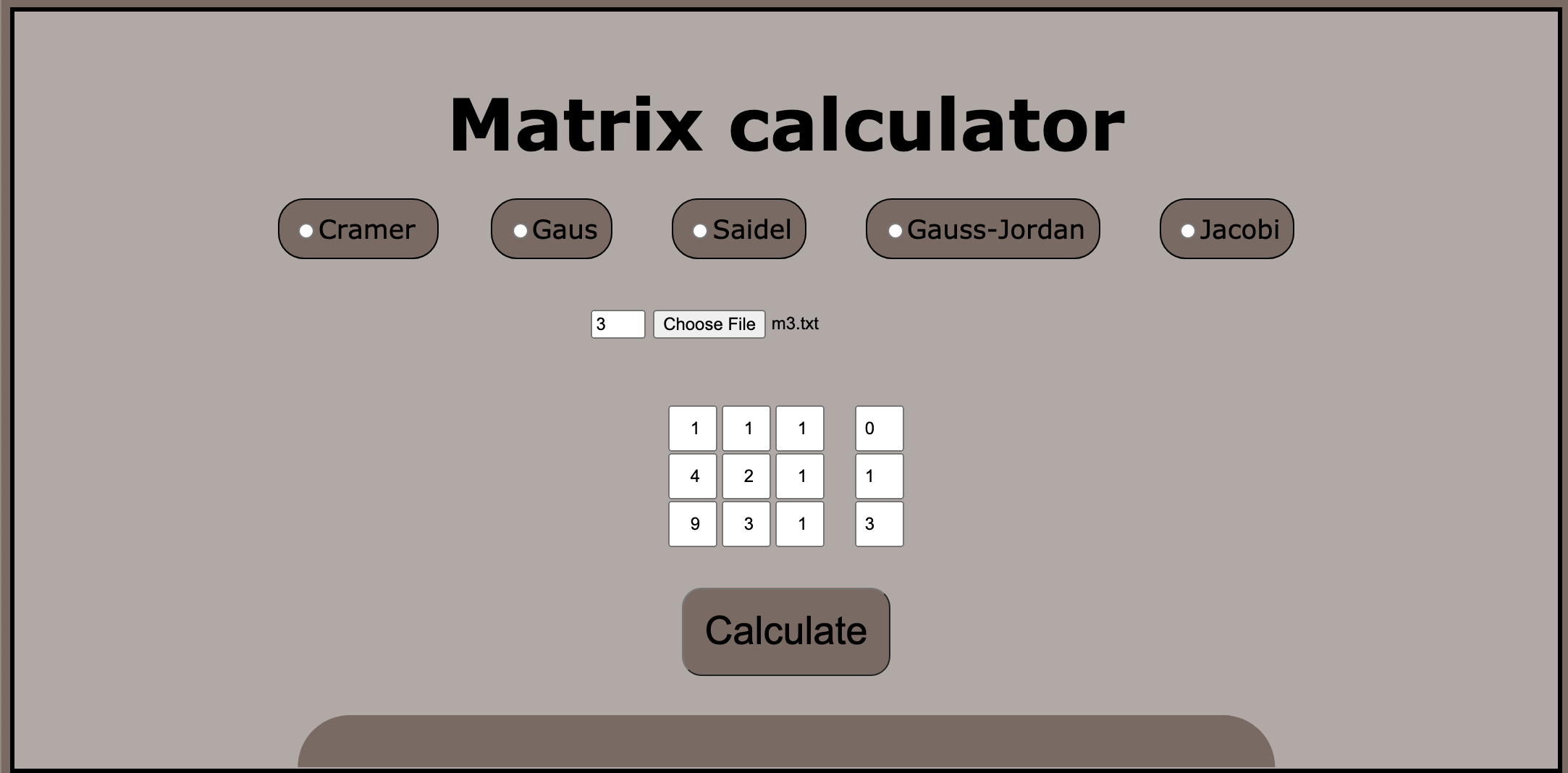


Рисунок 4 — Заповнена СЛАУ

Для того щоб отримати результат, потрібно нажати на «Calculate», після цього виконується логіка, яка, зважаючи на те, що ввів користувач, викликає потрібний метод і виводить результат. Вся ця логіка знаходиться в «t07/js/main.js»

Код «main.js»:

function UserException(message) {

this.message = message;

}

var matrix\_global;

var free\_members;

var x\_res;

function print\_check\_button() {

let check\_buttom = document.createElement("div");

check\_buttom.classList.add("result\_button");

if(corr\_sol(x\_res)) {

check\_buttom.innerHTML = "true";

check\_buttom.style.color = "green";

} else {

check\_buttom.innerHTML = "false";

check\_buttom.style.color = "red";

}

document.getElementById("result\_box").appendChild(check\_buttom);

}

function get\_user\_input() {

let size = document.getElementById("size\_matrix\_html").value;

// Получаем введенную пользователем основную матрицу

let matrix\_input\_table = document.getElementsByClassName("table\_input");

let matrix = new Array(size);

for (let i = 0; i < size; i++) {

matrix[i] = new Array(3);

}

let count = 0;

for (let i = 0; i < size; i++) {

for (let z = 0; z < size; z++) {

matrix[i][z] = parseInt(matrix\_input\_table[count].value);

count++;

}

}

// Получаем свободные члены

let free\_input\_table = document.getElementsByClassName("free\_member\_input");

free\_members = new Array(size);

for (let i = 0; i < size; i++) {

free\_members[i] = parseInt(free\_input\_table[i].value);

}

console.log(free\_members);

matrix\_global = matrix.slice();

return [size, matrix, free\_members];

}

function get\_solve\_sle(matrix, free\_members) {

let method\_name = document.querySelector('input[name="method"]:checked').value;

console.log(matrix);

if (method\_name == "Cramer") {

console.log("start2: " + matrix);

console.log(matrix);

if (free\_members.length > 4) {

throw new UserException("Max 4-n matrix for Cramer`s method");

}

x\_res = cramer(matrix.slice(), free\_members);

return x\_res;

}

if (method\_name == "Gaus") {

console.log("start2: " + matrix);

x\_res = gausa(createCopy(matrix), free\_members);

return x\_res;

}

if (method\_name == "Saidel") {

console.log("start2: " + matrix);

console.log(matrix.slice());

x\_res = Saidel(createCopy(matrix), free\_members);

return x\_res;

}

if (method\_name == "Jacobi") {

x\_res = Jacobi(createCopy(matrix), free\_members);

return x\_res;

}

if (method\_name == "Gauss-Jordan") {

x\_res = Gauss\_Jordan(createCopy(matrix), free\_members);

return x\_res;

}

return NaN;

}

function print\_result(result) {

console.log("Result: " + result);

if (document.getElementsByClassName("result\_text")[0]) {

document.getElementsByClassName("result\_text")[0].remove();

}

if (document.getElementsByClassName("result\_button")[0]) {

document.getElementsByClassName("result\_button")[0].remove();

}

let result\_html = document.createElement("span");

result\_html.classList.add("result\_text");

let str = "<h1>" + document.querySelector('input[name="method"]:checked').value +"</h1>"+ "<br>";

for (let index = 0; index < result.length; index++) {

str+= "x(" + index +"): " + (Number(result[index]).toFixed(5)) + "<br>";

}

result\_html.innerHTML = str;

document.getElementById("result\_box").appendChild(result\_html);

print\_check\_button();

}

function print\_error(message) {

if (document.getElementsByClassName("result\_text")[0]) {

document.getElementsByClassName("result\_text")[0].remove();

}

if (document.getElementsByClassName("result\_button")[0]) {

document.getElementsByClassName("result\_button")[0].remove();

}

let result\_html = document.createElement("span");

result\_html.classList.add("result\_text");

result\_html.innerHTML = message;

document.getElementById("result\_box").appendChild(result\_html);

}

document.getElementById("size\_matrix\_html").onchange = () => {

if (document.getElementsByClassName("my\_table")[0]) {

document.getElementsByClassName("my\_table")[0].remove();

}

let size = document.getElementById("size\_matrix\_html").value;

let table = document.createElement("table");

table.classList.add("my\_table");

let str\_html = "";

for (let i = 0; i < size; i++) {

str\_html += "<tr>";

for (let z = 0; z < size; z++) {

str\_html += "<td><input class ='table\_input' required></td>";

}

str\_html += "<td><input class ='free\_member\_input' required></td>";

str\_html += "</tr>"

}

table.innerHTML = str\_html;

document.getElementById("table\_html").appendChild(table);

}

document.getElementsByClassName("my\_form")[0].onsubmit = (event) => {

event.preventDefault();

let user\_data = get\_user\_input();

// 0 - size, 1 - matrix, 2 - free\_members

try {

let result = get\_solve\_sle(user\_data[1], user\_data[2]);

print\_result(result);

} catch(e) {

print\_error(e.message);

}

}

function load\_from\_file(object) {

let file = object.files[0];

let reader = new FileReader();

reader.readAsText(file);

reader.onload = function() {

let file\_str = reader.result;

let file\_array = file\_str.split(/(?:\n| )+/);

let size = file\_array[0];

if (document.getElementsByClassName("my\_table")[0]) {

document.getElementsByClassName("my\_table")[0].remove();

}

document.getElementById("size\_matrix\_html").value = size;

let table = document.createElement("table");

table.classList.add("my\_table");

let str\_html = "";

let index = 1;

for (let i = 0; i < size; i++) {

str\_html += "<tr>";

for (let z = 0; z < size; z++) {

str\_html += "<td><input class ='table\_input' required value=\" " + file\_array[index++] + "\"></td>";

}

str\_html += "<td><input class ='free\_member\_input' value=\" " + file\_array[size\*size + i + 1] + "\" required></td>";

str\_html += "</tr>"

}

table.innerHTML = str\_html;

document.getElementById("table\_html").appendChild(table);

};

}

На рисунку 5, ви можете побачити як программа виводить результат.

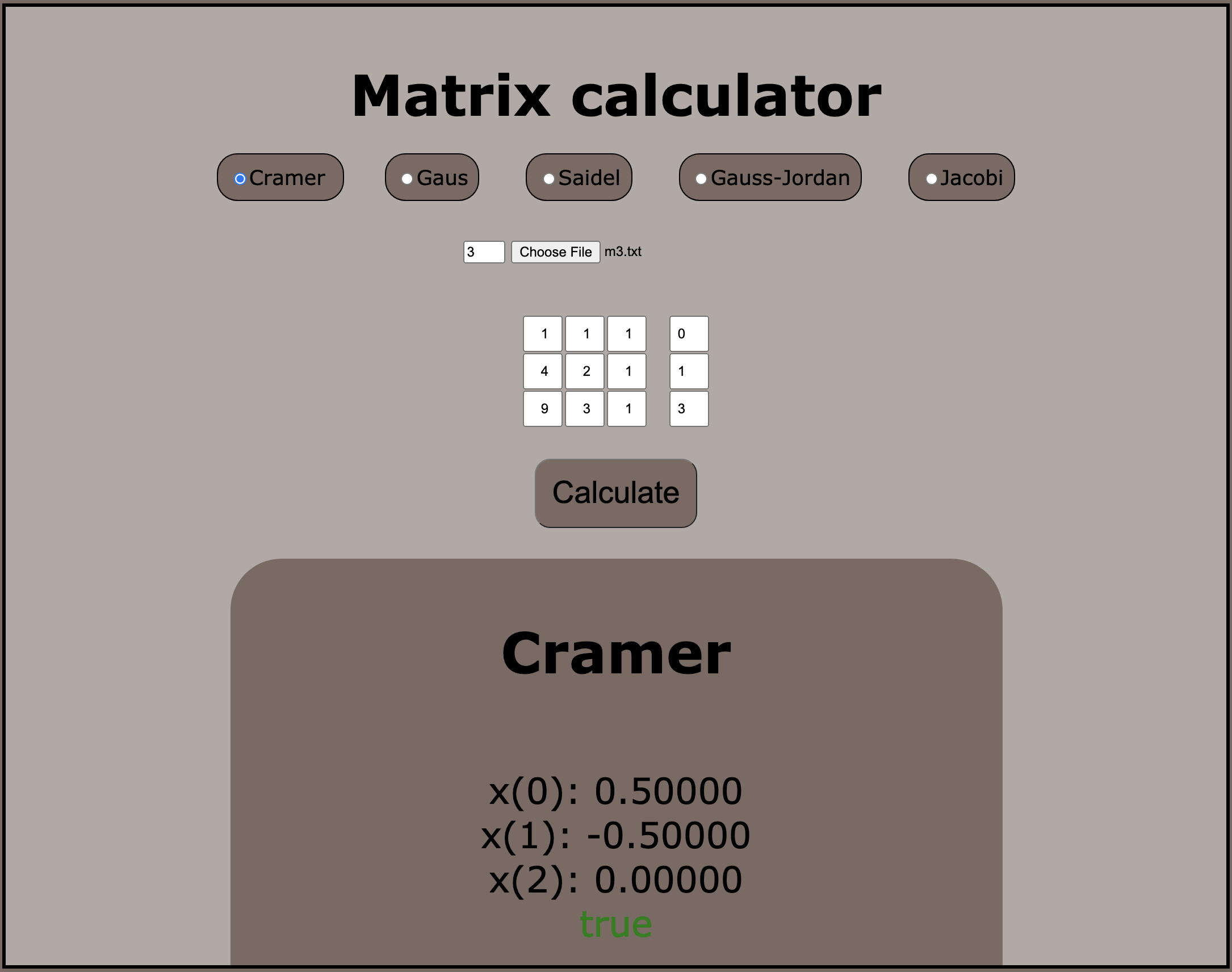


Рисунок 5- Виведення результату

Також була реалізована перевірка на те, чи правильний програма знайшла результат. Код перевірки ви можете знайти в «t06/corr\_solution.js» або в «main.js»

Код «corr.js»:

function corr\_sol(x) {

for (let index = 0; index < free\_members.length; index++) {

let res = 0;

for (let i = 0; i < free\_members.length; i++) {

res += (matrix\_global[index][i] \* x[i]); }

if (res < free\_members[index] + 0.001 && res > free\_members[index] - 0.001) {

res = 0;

continue;

}

else {

return false;

}

}

return true;

}

Приклад, коли программа рахує корені неправильно, зображено на рисунку 6

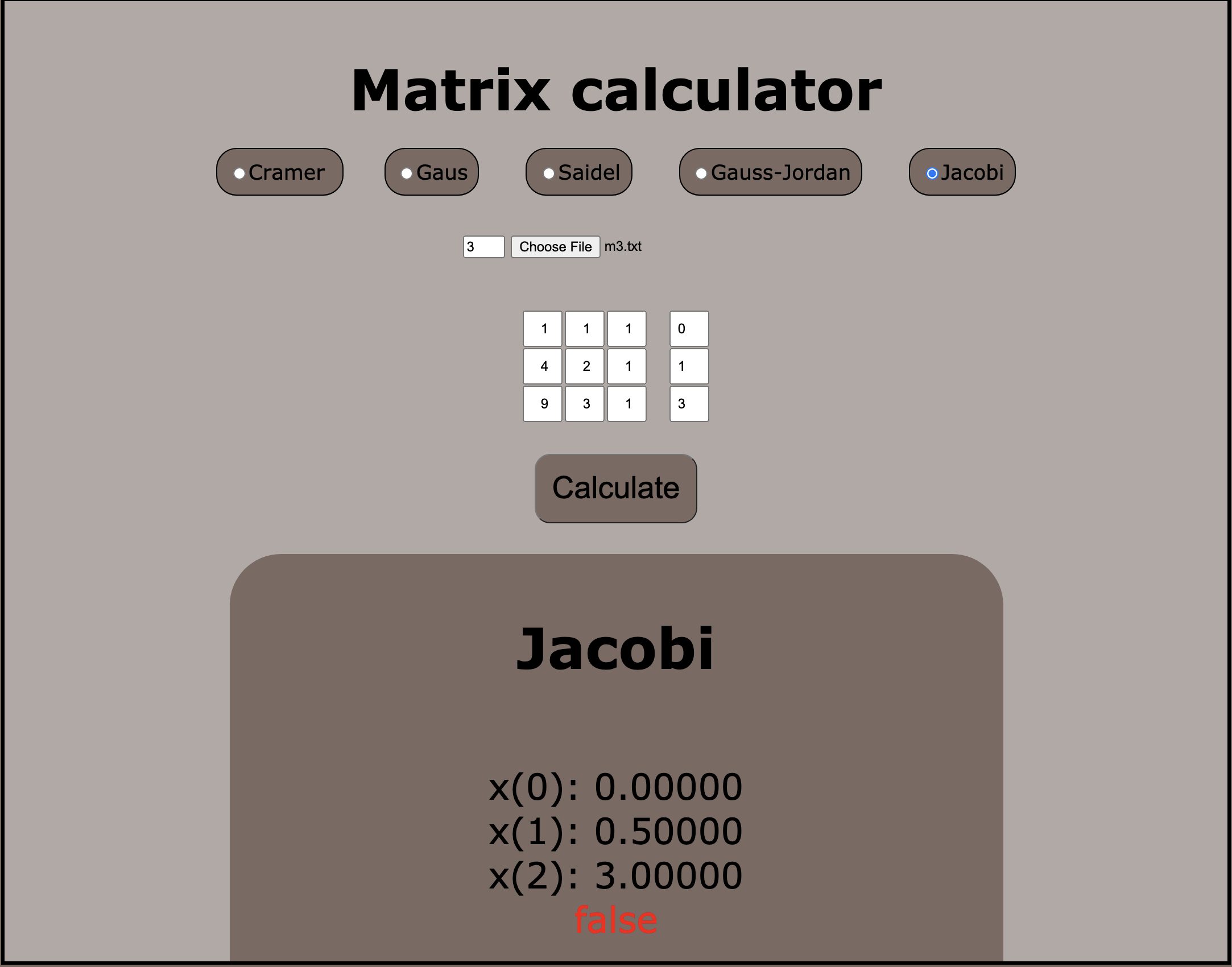


Рисунок 6 — Программа порахувала корені не правильно

В завданні 8 потрібно було, дослідити ситуації, коли корні знайти не можна, і повідомляти про це користувачу, для цього було реалізовано декілька функцій в «t07/js/correct\_func.js» та «t07/js/main.js» був перероблений з використанням виключень.

Код «t07/js/mx\_check\_correct.js»:

function check\_iterible\_diagram(matrix) {

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

let sum = 0;

for (let j = 0; j < matrix.length; j++) {

if (i === j) continue

sum += matrix[i][j]

}

if (Math.abs(matrix[i][i]) < Math.abs(sum)) {

return false;

}}

return true;}

Для тестування виключень, спробуємо знайти корені, для СЛАУ, яка не підходе для методу Крмаера (рис. 7)

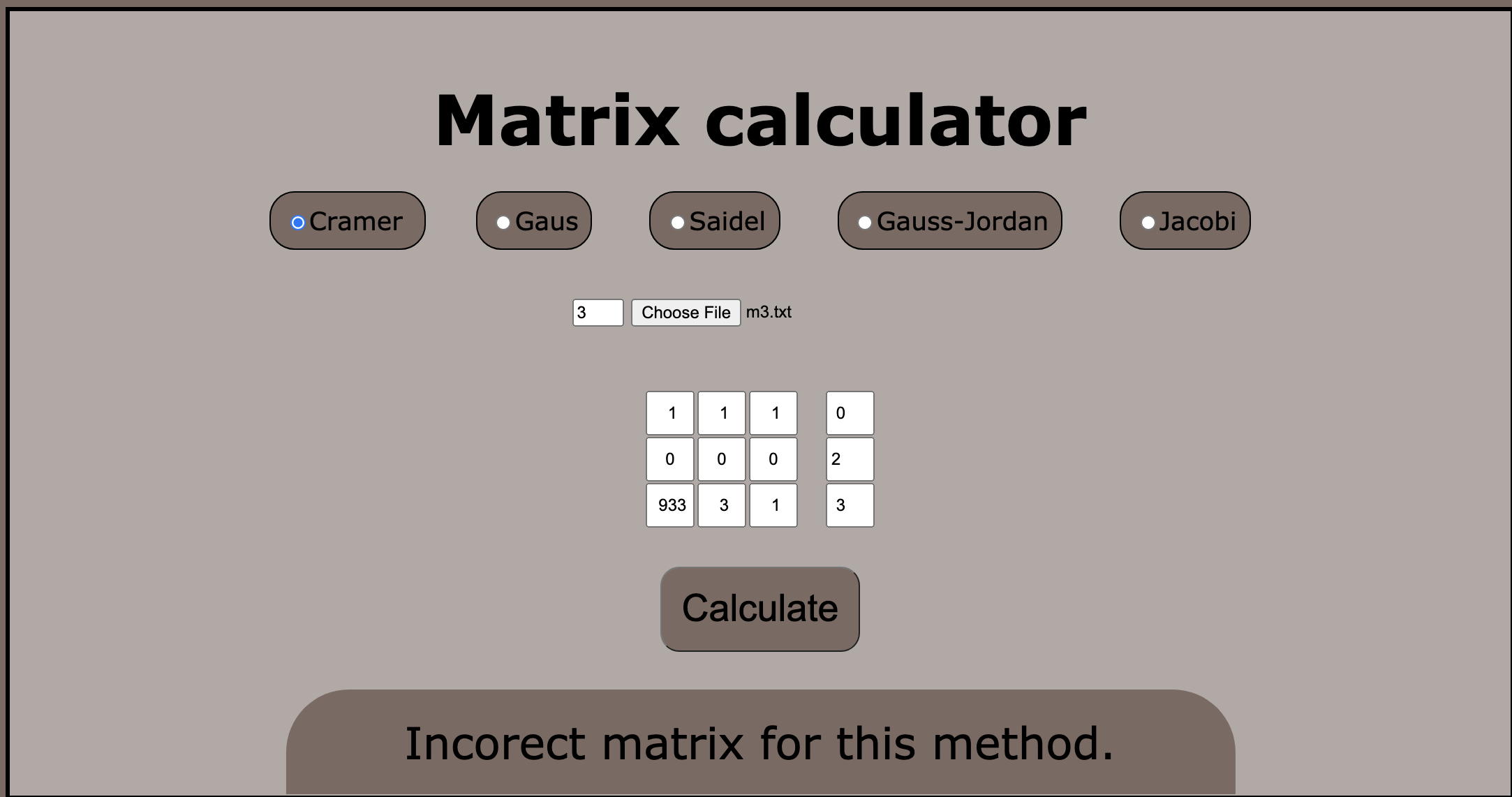


Рисунок 7 — Тестування виключень

Інтерфейс взаємодії та його стиль були описані в файлах «t07/index.html» та «t07/css/style.css» відповідно.

«t07/index.html»:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Solve it</title>

<link rel="stylesheet" href="./css/style.css"

</head>

<body>

<div class = "m\_div">

<form class = "my\_form" method = "post">

<div class = "main\_div">

<div class = "j\_div">

<h1>Matrix calculator</h1>

<span class = "method\_type\_button"><input type= "radio" name = "method" required value="Cramer" id = "Cramer"><label for ="Cramer">Cramer</label> </span>

<span class = "method\_type\_button"><input type= "radio" name = "method" required value="Gaus" id = "Gaus"><label for ="Gaus">Gaus</label></span>

<span class = "method\_type\_button"><input type= "radio" name = "method" required value="Saidel" id = "Saidel"><label for ="Saidel">Saidel</label></span> </span>

<span class = "method\_type\_button"><input type= "radio" name = "method" required value="Jacobi" id = "Jacobi"><label for ="Jacobi">Jacobi</label></span></span>

<span class = "method\_type\_button"><input type= "radio" name = "method" required value="Gaus-Seidel" id = "Gaus-Seidel"><label for ="Gaus-Seidel">Gaus-Seidel</label></span></span>

</div>

<div class = "j\_div">

<input type="number" min="1" max="10" id = "size\_matrix\_html">

<input type= "file" class = "input\_file" onchange="load\_from\_file(this)">

</div>

<div id = "table\_html">

</div>

<div class = "button\_div"><button>Calculate</button></div>

<div id = "result\_box"></div>

</div>

</div>

<script src="js/correct\_func.js"></script>

<script src= "js/Cramer.js"></script>

<script src= "js/Gaus.js"></script>

<script src= "js/Seidel.js"></script>

<script src= "js/Jacobi.js"></script>

<script src= "js/Gaus-Seidel.js"></script>

<script src="js/main.js"></script>

</form>

</body>

</html>

«t07/css/style.css» :

body {

background-color: rgb(138, 93, 63);

}

.m\_div {

text-align: center;

background-color: rgb(177, 169, 165);

}

.table\_input {

text-align: center;

width: 30px;

height: 30px;

}

.free\_member\_input {

width: 30px;

height: 30px;

margin-left: 20px;

}

h1 {

font-family: monospace;

font-size: 55px;

}

.method\_type\_button {

user-select: none;

border: 1px black solid;

background-color: rgb(124, 106, 98);

border-radius: 20px;

margin: 20px;

margin-bottom: 50px;

font-size: 20px;

padding: 10px;

}

.j\_div {

margin: 50px;

}

#result\_box {

margin-top: 30px;

margin-left: 50%;

transform: translateX(-50%);

font-size: 35px;

background-color: rgb(124, 106, 98);

border-top-left-radius: 50px;

border-top-right-radius: 50px;

width: 60%;

padding: 20px;

}

.button\_div button{

background-color: rgb(124, 106, 98);

font-size: 30px;

padding: 15px;

border-radius: 15px;

outline: none;;

}

3 ТЕСТУВАННЯ РОЗРОБЛЕНОГО ЗАСТОСУНКУ

На рисунках 8-12 зображено тестування кожного методу.

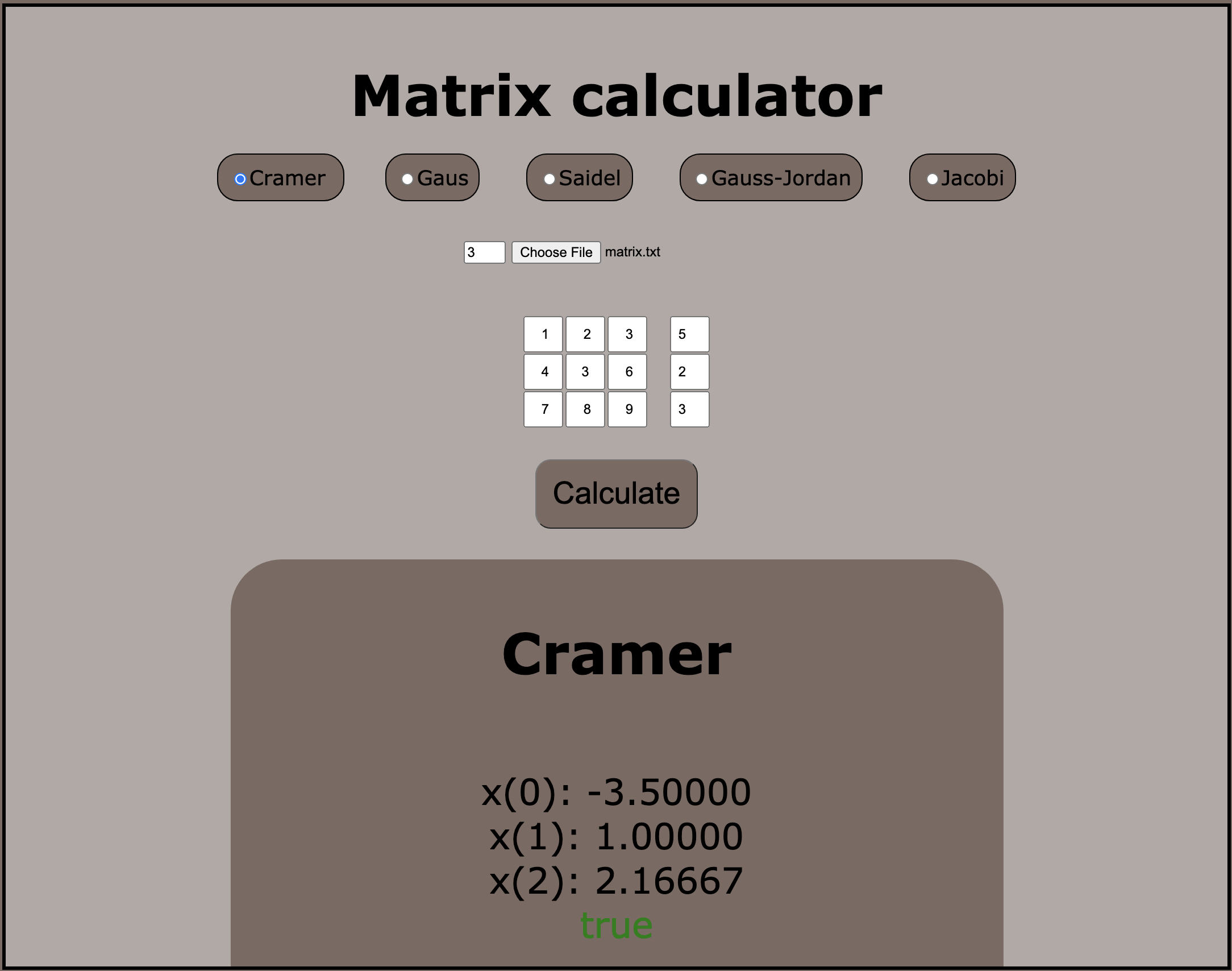


Рисунок 8 — Тестування методу Крамера

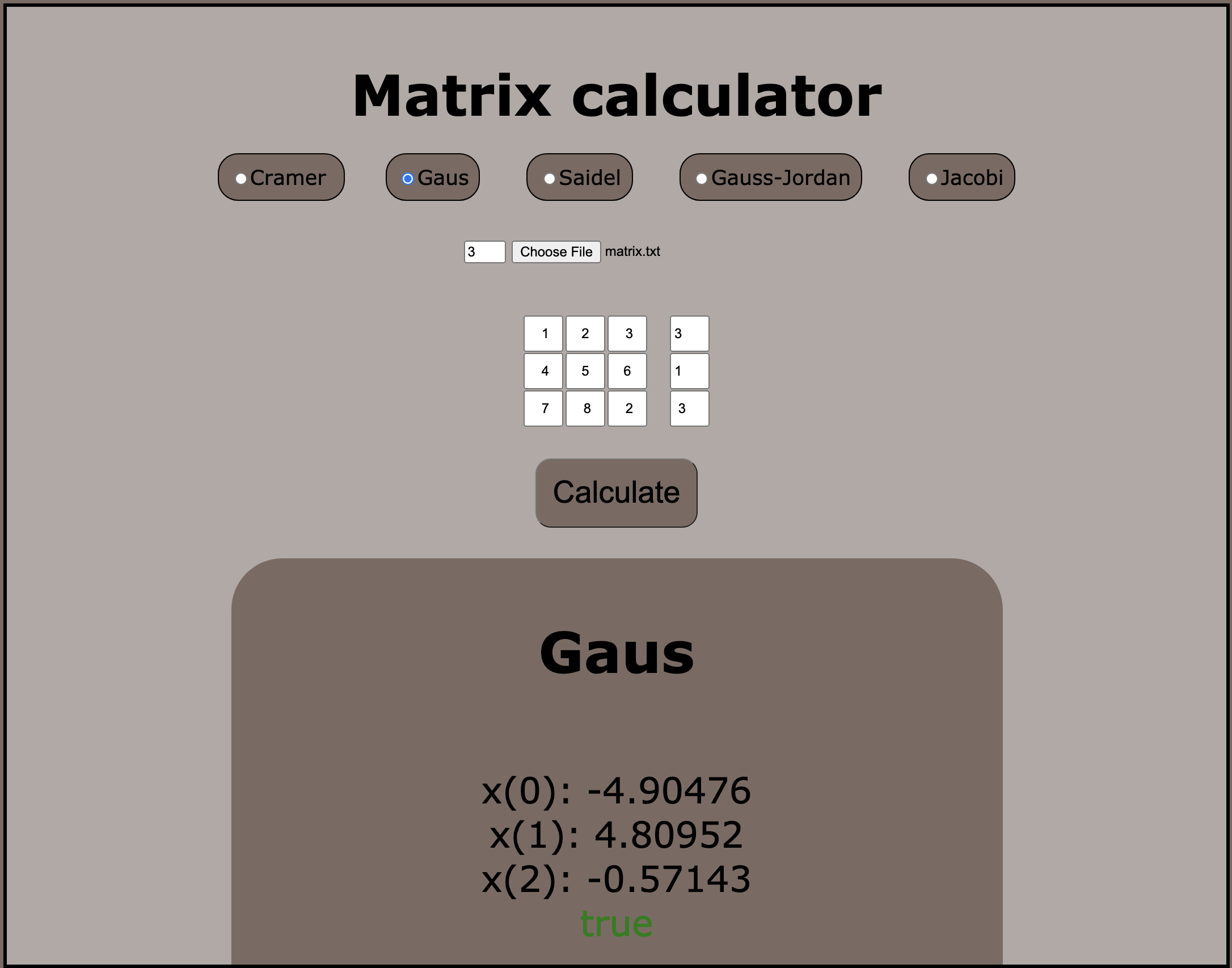


Рисунок 9 — Тестування методу Гауса

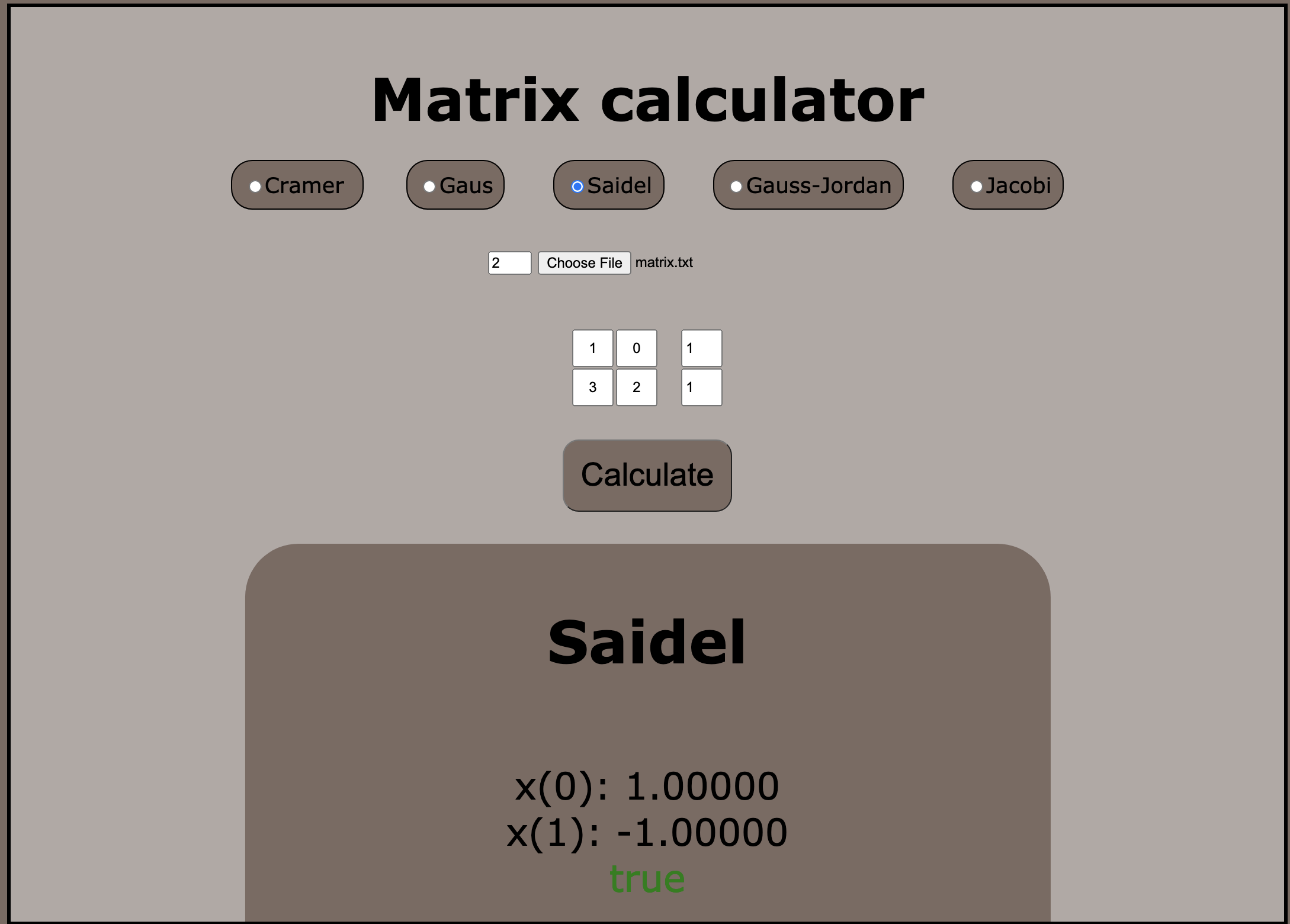


Рисунок 10 — Тестування методу Зельделя

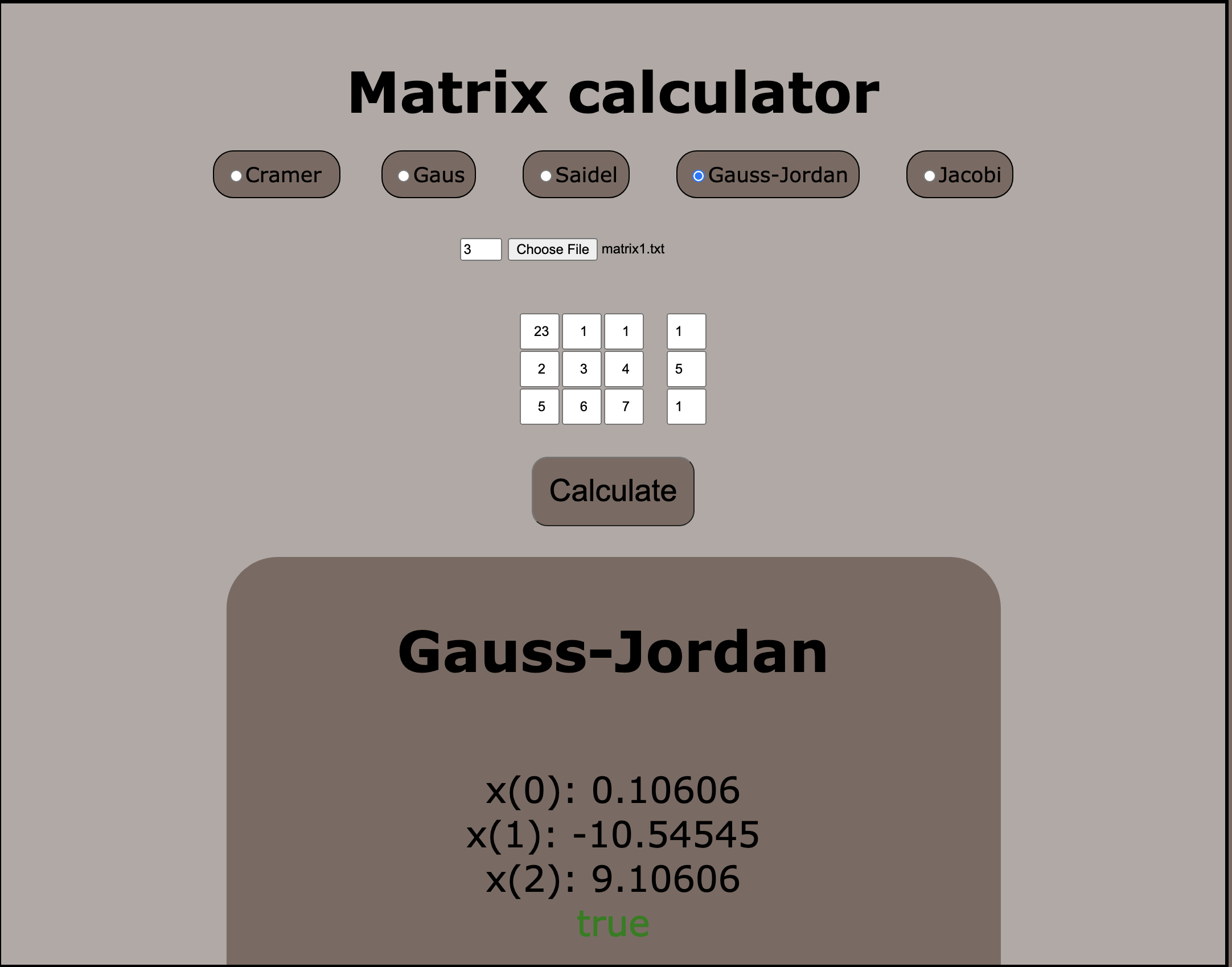


Рисунок 11 — Тестування методу Гауса-Джордона

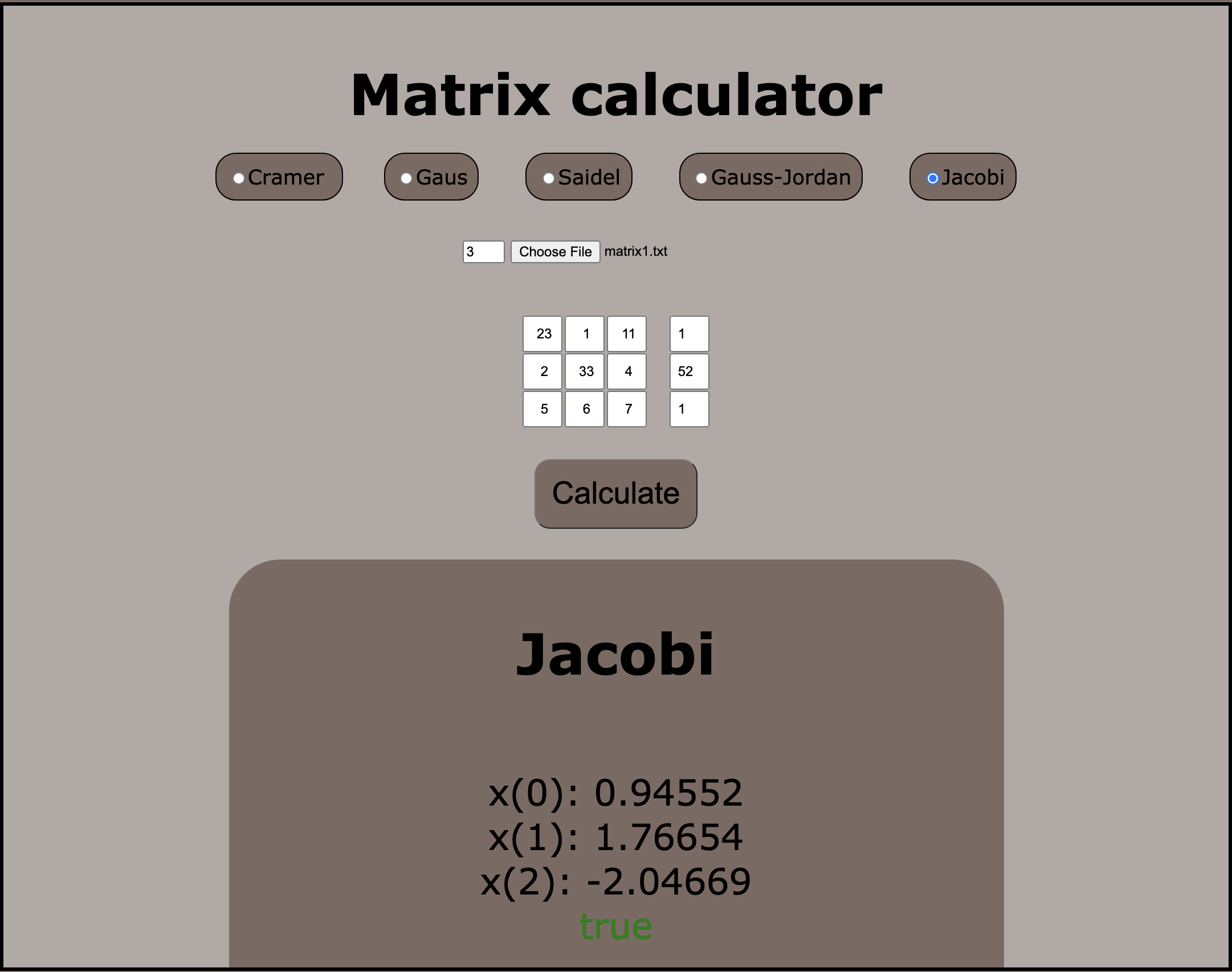


Рисунок 12 — Тестування методу Якобі

ВИСНОВКИ

Під час виконання даної лабораторної роботи була створена програма для знаходження коренів СЛАУ різними методами. В програмі були пророблені випадки, коли корені знайти не можливо, і тд. Під час виконання цього спрінта, здобув навички при роботі з різними методами для розв’язвування СЛАУ. З тяжкими трудостями не зіткнувся.

СПИСОК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Конспект лекцій з дисципліни «Чисельні методи».
2. https://uk.wikipedia.org/wiki/Нуль\_функції, 21.02.2021.
3. https://uk.wikipedia.org/wiki/Метод\_Ньютона\_в\_оптимізації, 21.02.2021.
4. https://uk.wikipedia.org/wiki/Метод\_хорд, 21.02.2021.
5. https://uk.wikipedia.org/wiki/Метод\_бісекції, 21.02.2021.