МІНІСТЕРСТВО КУЛЬТУРИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КУЛЬТУРИ І МИСТЕЦТВ

ФАКУЛЬТЕТ ЕКОНОМІКИ І ІТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Звіт

з дисципліни «Системний аналіз»

на тему «Обробка стеків»

Виконав:

студент 3 курсу

групи КН-23

Коваль Максим

Київ –2015

ЗМІСТ

[РОЗДІЛ 1 ПОСТАНОВКА ЗАДАЧІ 3](#_Toc431886865)

[РОЗДІЛ 2 ОПИСИ АЛГОРИТМУ РОЗВ’ЯЗАННЯ ЗАДАЧІ 4](#_Toc431886866)

[2.1 Опис алгоритму додавання k-елементів до стеку висоти n після l-того елементу, якщо стеки X, Y, Z мають правосторонній, або лівосторонній спосіб запису 4](#_Toc431886867)

[2.2 Опис алгоритму додавання k-елементів до стеку висоти n після l-того елементу, якщо стеки X, Z мають правосторонній, або лівосторонній спосіб запису, а стек Y має протилежний спосіб запису 4](#_Toc431886868)

[2.3 Опис алгоритму додавання k-елементів до стеку висоти n після l-того елементу, якщо стеки X, Y мають правосторонній, або лівосторонній спосіб запису, а стек Z має протилежний спосіб запису 4](#_Toc431886869)

[2.4 Опис алгоритму додавання k-елементів до стеку висоти n після l-того елементу, якщо стеки Y, Z мають правосторонній, або лівосторонній спосіб запису, а стек X має протилежний спосіб запису 5](#_Toc431886870)

[2.5 Опис алгоритму видалення k-елементів після l-того зі стеку висоти n, якщо стеки X, Z мають правосторонній, або лівосторонній спосіб запису 5](#_Toc431886871)

[2.6 Опис алгоритму видалення k-елементів після l-того зі стеку висоти n, якщо стек X має правосторонній, або лівосторонній спосіб запису, а стек Z має протилежний спосіб запису 5](#_Toc431886872)

[РОЗДІЛ 3 БЛОК-СХЕМИ ПРОГРАМНОГО ПРОДУКТУ 6](#_Toc431886873)

[РОЗДІЛ 4 ЛІСТИНГ КОДУ 9](#_Toc431886874)

[4.1 Файл index.html 9](#_Toc431886875)

[4.2 Файл style.css 10](#_Toc431886876)

[4.3 Файл script.js 13](#_Toc431886877)

[РОЗДІЛ 5 SCREENSHOTS 21](#_Toc431886878)

# РОЗДІЛ 1 ПОСТАНОВКА ЗАДАЧІ

Написати програму обробки стека, яка передбачає:

* Додавання k-елементів до стеку висоти n після l-того елементу.
* Видалення k-елементів після l-того зі стеку висоти n.

При написанні програми врахувати всі можливі варіації запису стеків.

# РОЗДІЛ 2 ОПИСИ АЛГОРИТМУ РОЗВ’ЯЗАННЯ ЗАДАЧІ

Задача була розв’язана за допомогою мови програмування JavaScript і її бібліотеки jQuery. Для візуалізації задачі в браузері були використані такі засоби верстки як HTML5 і CSS3. Переглянути роботу програми можна в будь-якому сучасному браузері.

Початкові умови: є три масиви X, Y, Z. X – початковий стан стека. Y - стек, який потрібно додати до стека Х після l-того елементу. Z – пустий допоміжний стек для результату. Всі стеки були представленні за допомогою масивів.

## 2.1 Опис алгоритму додавання k-елементів до стеку висоти n після l-того елементу, якщо стеки X, Y, Z мають правосторонній, або лівосторонній спосіб запису

1. Переміщаємо в початок масиву Z елементи масиву Х з індексами від 1 до l.
2. Переміщаємо в кінець масиву Z всі k-елементи з масиву Y.
3. Переміщаємо в кінець масиву Z всі елементи з масиву Х з індексами від k+l до n.

## 2.2 Опис алгоритму додавання k-елементів до стеку висоти n після l-того елементу, якщо стеки X, Z мають правосторонній, або лівосторонній спосіб запису, а стек Y має протилежний спосіб запису

1. Перевертаємо масив Y.
2. Переміщаємо в початок масиву Z елементи масиву Х з індексами від 1 до l.
3. Переміщаємо в кінець масиву Z всі k-елементи з масиву Y.
4. Переміщаємо в кінець масиву Z всі елементи з масиву Х з індексами від k+l до n.

## 2.3 Опис алгоритму додавання k-елементів до стеку висоти n після l-того елементу, якщо стеки X, Y мають правосторонній, або лівосторонній спосіб запису, а стек Z має протилежний спосіб запису

1. Переміщаємо в початок масиву Z елементи масиву Х з індексами від 1 до l.
2. Переміщаємо в кінець масиву Z всі k-елементи з масиву Y.
3. Переміщаємо в кінець масиву Z всі елементи з масиву Х з індексами від k+l до n.
4. Перевертаємо масив Z.

## 2.4 Опис алгоритму додавання k-елементів до стеку висоти n після l-того елементу, якщо стеки Y, Z мають правосторонній, або лівосторонній спосіб запису, а стек X має протилежний спосіб запису

1. Перевертаємо масив Y.
2. Переміщаємо в початок масиву Z елементи масиву Х з індексами від 1 до l.
3. Переміщаємо в кінець масиву Z всі k-елементи з масиву Y.
4. Переміщаємо в кінець масиву Z всі елементи з масиву Х з індексами від k+l до n.
5. Перевертаємо масив Z.

## 2.5 Опис алгоритму видалення k-елементів після l-того зі стеку висоти n, якщо стеки X, Z мають правосторонній, або лівосторонній спосіб запису

1. Переміщаємо в початок масиву Z елементи масиву Х з індексами від 1 до l.
2. Переміщаємо в кінець масиву Z всі елементи масиву X, які стоять після k+l-елементів масиву.

## 2.6 Опис алгоритму видалення k-елементів після l-того зі стеку висоти n, якщо стек X має правосторонній, або лівосторонній спосіб запису, а стек Z має протилежний спосіб запису

1. Переміщаємо в початок масиву Z елементи масиву Х з індексами від 1 до l.
2. Переміщаємо в кінець масиву Z всі елементи масиву X, які стоять після k+l-елементів масиву.
3. Перевертаємо масив Z.

# РОЗДІЛ 3 БЛОК-СХЕМИ ПРОГРАМНОГО ПРОДУКТУ

C:\Users\Коваль Максим\Downloads\головна блок-схема стек.png

Рис. 1. Головна блок-схема

D:\ThirdCourse15_16\System Analysis\Koval\стек\блок-схема А.png

Рис. 2. Блок-схема А

D:\ThirdCourse15_16\System Analysis\Koval\стек\блок-схема В.png

Рис. 3. Блок-схема В

D:\ThirdCourse15_16\System Analysis\Koval\стек\блок-схема С.png

Рис. 4. Блок-схема С

# РОЗДІЛ 4 ЛІСТИНГ КОДУ

## 4.1 Файл index.html

<!DOCTYPE HTML>

<html>

<head>

<meta http-equiv="content-type" content="text/html; charset=utf-8">

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

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

<script src="script.js" type="text/javascript"></script>

<title>Stack</title>

</head>

<body>

<div id="task">

<div id="enters">

<span>n = </span><input type="text" id="n"/>

<span>k = </span><input type="text" id="k"/>

<span>l = </span><input type="text" id="l"/>

<button id="apply">Ввести</button><br>

</div>

<div id="checkboxs">

<div id="autoSize">

<div id="enterX">

<input type="checkbox" checked="true" id="rex"><label for="rex">Правосторонній спосіб вводу стека Х</label><br>

<input type="checkbox" id="lex"><label for="lex">Лівосторонній спосіб вводу стека Х</label><br>

</div>

<div id="enterY">

<input type="checkbox" checked="true" id="rey"><label for="rey">Правосторонній спосіб вводу стека Y</label><br>

<input type="checkbox" id="ley"><label for="ley">Лівосторонній спосіб вводу стека Y</label><br>

</div>

<div id="result">

<input type="checkbox" checked="true" id="rs"><label for="rs">Правосторонній запис результату</label><br>

<input type="checkbox" id="ls"><label for="ls">Лівосторонній запис результату</label><br>

</div>

</div>

</div>

</div>

<div id="wrapper">

</div>

</body>

</html>

## 4.2 Файл style.css

input {

background: linear-gradient(to right, #FFF, #AAA);

}

body {

min-width: 1000px;

background: linear-gradient(to right, #FFF, #63DD8D);

}

#wrapper {

width: 95%;

margin: 10px auto;

}

#task {

width: 95%;

margin: 0 auto;

height: 100px;

background: linear-gradient(to right, #63DD8D, #007929);

padding-top: 5px;

border-radius: 5px;

}

#inputX, #inputY, #inputZ {

width: 10%;

float: left;

background-color: #63DD8D;

margin-right: 4px;

padding-right: 3px;

border-radius: 5px;

}

#inputX input, #inputY input, #inputZ input {

width: 36px;

height: 36px;

text-align: center;

font-size: 20px;

border-radius: 5px;

}

#X, #Y, #Z{

width: 36px;

margin: 0 auto;

padding-bottom: 10px;

}

#checkboxs {

height: 45px;

margin: 5px auto;

width: 90%;

padding-top: 5px;

text-align: enter;

border-radius: 5px;

}

#enterX, #enterY, #result {

float: left;

width: 280px;

background-color: #00BB3F;

margin: 0.5px;

border-radius: 5px;

border: 1px solid #FFF;

}

#X span, #Y span, #Z span {

margin-left: 13px;

border-radius: 5px;

}

#enters {

width: 400px;

margin: 0 auto;

border-radius: 5px;

}

#enters span {

font-size: 20px;

}

#n, #k, #l {

width: 70px;

height: 25px;

font-size: 20px;

border-radius: 5px;

}

#apply {

width: 70px;

height: 31px;

border-radius: 5px;

}

#autoSize {

margin: 0 auto;

width: 850px;

}

#plus, #minus {

width: 50px;

height: 50px;

font-size: 20px;

border-radius: 50%;

}

## 4.3 Файл script.js

var x = [];

var y = [];

var z = [];

$(document).ready(function() {

$("#apply").click(function() {

$("#apply").remove();

$("#wrapper").append("<div id='inputX'></div>");

$("#wrapper").append("<div id='inputY'></div>");

var element = document.createElement('div');

element.id = "X";

inputX.appendChild(element);

var element = document.createElement('div');

element.id = "Y";

inputY.appendChild(element);

var element = document.createElement('span');

element.innerHTML = "X";

X.appendChild(element);

var element = document.createElement('span');

element.innerHTML = "Y";

Y.appendChild(element);

for (var i=0;i<($("#n").val());i++) {

var element = document.createElement('input');

element.classList.add("xelements");

X.appendChild(element);

var element = document.createElement('br');

X.appendChild(element);

}

for (var i=0;i<($("#k").val());i++) {

var element = document.createElement('input');

element.classList.add("yelements");

Y.appendChild(element);

var element = document.createElement('br');

Y.appendChild(element);

}

var element = document.createElement('button');

element.id = "plus";

element.innerHTML = "+";

wrapper.appendChild(element);

var element = document.createElement('button');

element.id = "minus";

element.innerHTML = "-";

wrapper.appendChild(element);

});

});

var show = function(array) {

var element = document.createElement('div');

element.id = "inputZ";

wrapper.appendChild(element);

var element = document.createElement('div');

element.id = "Z";

inputZ.appendChild(element);

var element = document.createElement('span');

element.innerHTML = "Z";

Z.appendChild(element);

for (var i=0; i<array.length; i++) {

var element = document.createElement('input');

element.value = array[i];

Z.appendChild(element);

var element = document.createElement('br');

Z.appendChild(element);

}

var element = document.createElement('img');

element.id = "inputZ";

wrapper.appendChild(element);

}

/////////////////////////////////////////////////////////////////////

$(document).on("click", "#plus", function(){

var l = parseInt($("#l").val());

var n = parseInt($("#n").val());

var k = parseInt($("#k").val());

$("#plus").remove();

$("#minus").remove();

var inputs\_x = document.getElementsByClassName( 'xelements' ),

xelements = [].map.call(inputs\_x, function( input ) {

return input.value;

}).join( ',' );

x = xelements.split(",");

var inputs\_y = document.getElementsByClassName( 'yelements' ),

yelements = [].map.call(inputs\_y, function( input ) {

return input.value;

}).join( ',' );

y = yelements.split(",");

if ($("#rex").prop("checked")) {

if ($("#rey").prop("checked")) {

if ($("#rs").prop("checked")) {

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=0; i<k; i++) {

z[i+l] = y[i];

}

for (var i=0; i<n-l; i++) {

z[l+k+i] = x[l+i];

}

show(z);

}

else if ($("#ls").prop("checked")) {

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=0; i<k; i++) {

z[i+l] = y[i];

}

for (var i=0; i<n-l; i++) {

z[l+k+i] = x[l+i];

}

z.reverse();

show(z);

}

}

else if ($("#ley").prop("checked")) {

if ($("#rs").prop("checked")) {

y.reverse();

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=0; i<k; i++) {

z[i+l] = y[i];

}

for (var i=0; i<n-l; i++) {

z[l+k+i] = x[l+i];

}

show(z);

}

else if ($("#ls").prop("checked")) {

y.reverse();

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=0; i<k; i++) {

z[i+l] = y[i];

}

for (var i=0; i<n-l; i++) {

z[l+k+i] = x[l+i];

}

z.reverse();

show(z);

}

}

}

else if ($("#lex").prop("checked")) {

if ($("#rey").prop("checked")) {

if ($("#rs").prop("checked")) {

y.reverse();

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=0; i<k; i++) {

z[i+l] = y[i];

}

for (var i=0; i<n-l; i++) {

z[l+k+i] = x[l+i];

}

z.reverse();

show(z);

}

else if ($("#ls").prop("checked")) {

y.reverse();

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=0; i<k; i++) {

z[i+l] = y[i];

}

for (var i=0; i<n-l; i++) {

z[l+k+i] = x[l+i];

}

show(z);

}

}

else if ($("#ley").prop("checked")) {

if ($("#rs").prop("checked")) {

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=0; i<k; i++) {

z[i+l] = y[i];

}

for (var i=0; i<n-l; i++) {

z[l+k+i] = x[l+i];

}

z.reverse();

show(z);

}

else if ($("#ls").prop("checked")) {

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=0; i<k; i++) {

z[i+l] = y[i];

}

for (var i=0; i<n-l; i++) {

z[l+k+i] = x[l+i];

}

show(z);

}

}

}

});

///////////////////////////////////////////////////////////////////////

$(document).on("click", "#minus", function(){

var l = parseInt($("#l").val());

var n = parseInt($("#n").val());

var k = parseInt($("#k").val());

$("#plus").remove();

$("#minus").remove();

var inputs\_x = document.getElementsByClassName( 'xelements' ),

xelements = [].map.call(inputs\_x, function( input ) {

return input.value;

}).join( ',' );

x = xelements.split(",");

$("#inputY").remove();

if (($("#rex").prop("checked"))) {

if ($("#rs").prop("checked")) {

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=l+k; i<n; i++) {

z[i-k] = x[i];

}

show(z);

}

else if ($("#ls").prop("checked")) {

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=l+k; i<n; i++) {

z[i-k] = x[i];

}

z.reverse();

show(z)

}

}

else if ($("#lex").prop("checked")) {

if ($("#rs").prop("checked")) {

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=l+k; i<n; i++) {

z[i-k] = x[i];

}

z.reverse();

show(z);

}

else if ($("#ls").prop("checked")) {

for (var i=0; i<l; i++) {

z[i] = x[i];

}

for (var i=l+k; i<n; i++) {

z[i-k] = x[i];

}

show(z);

}

}

});

# РОЗДІЛ 5 SCREENSHOTS

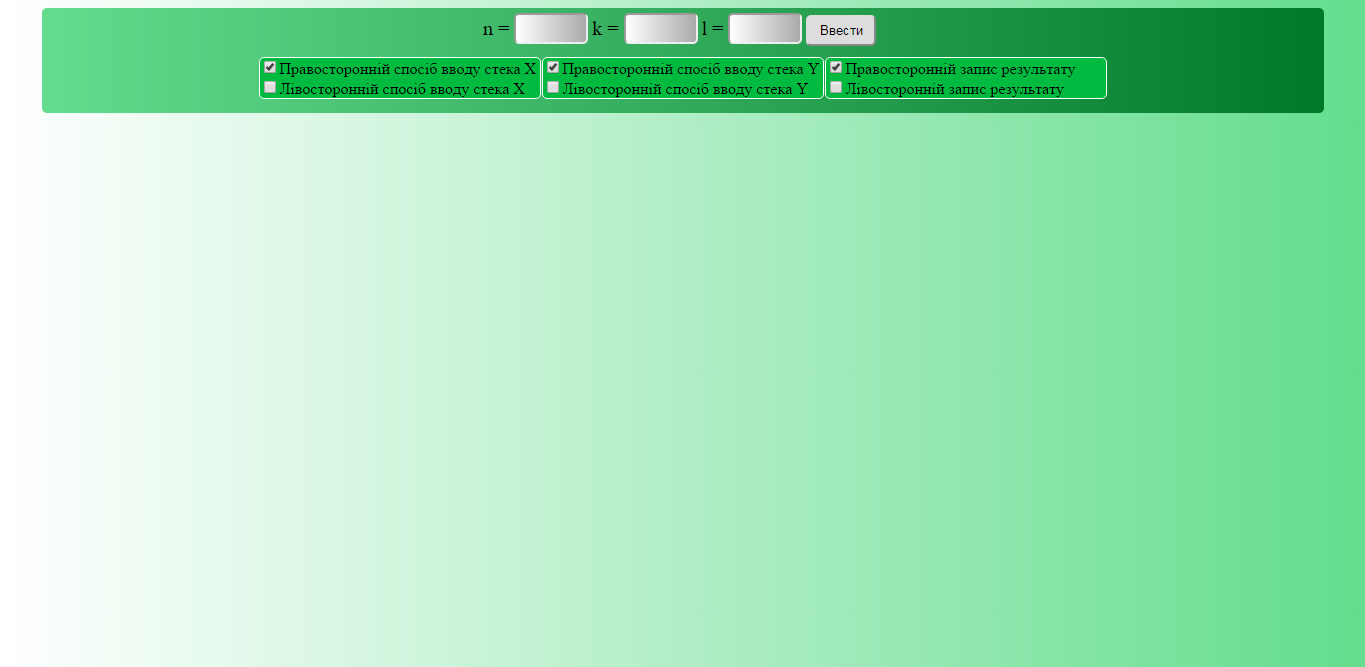


Рис. 5. Початкове вікно програми

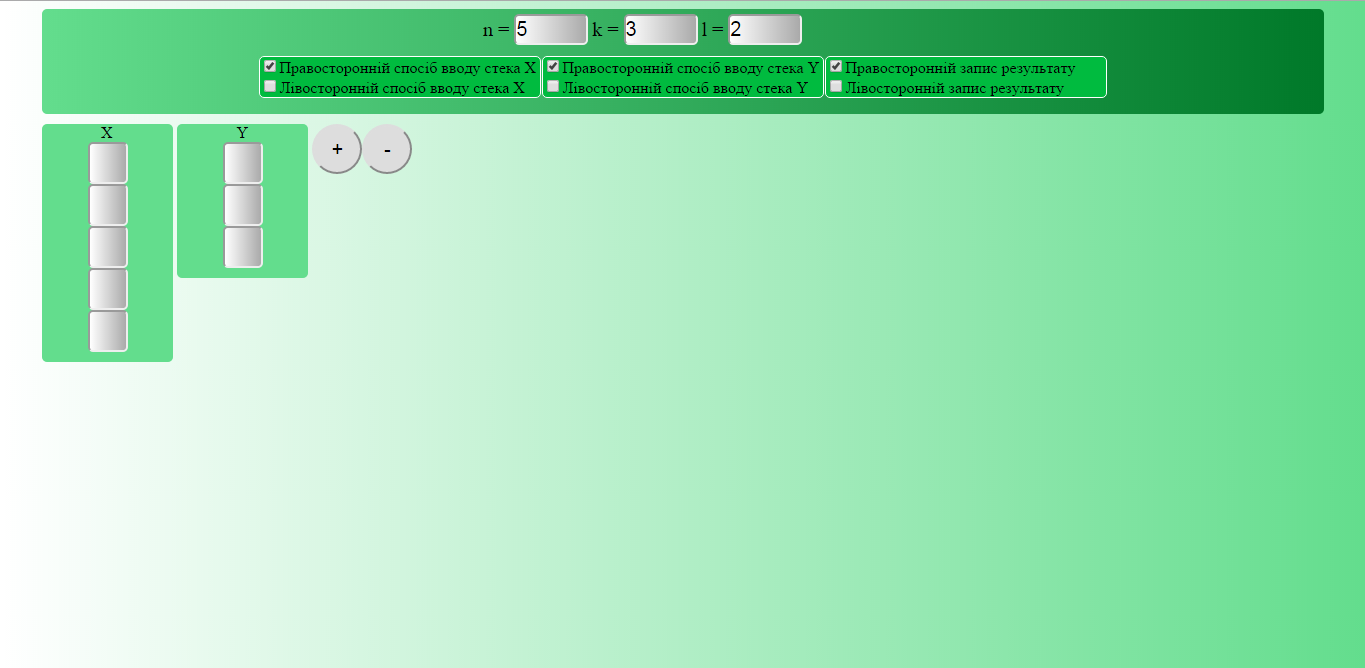


Рис. 6. Вікно вводу умови

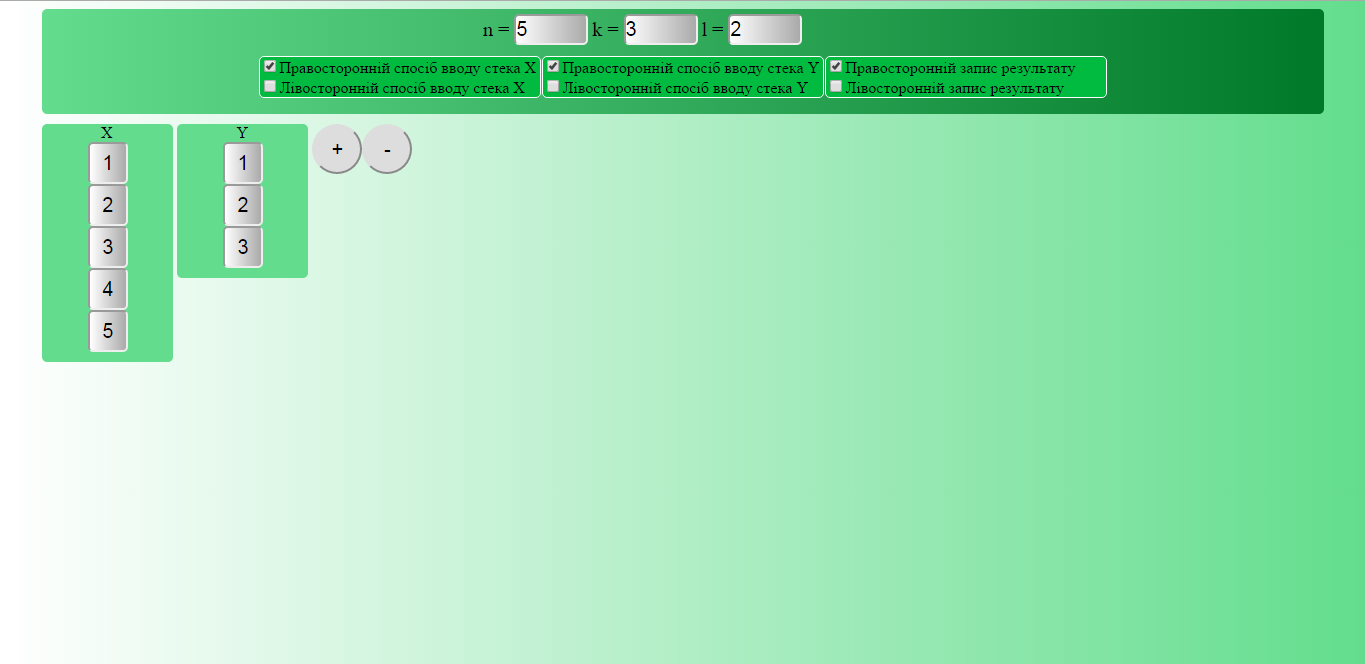


Рис. 7. Введення початкових значень стеків X, Y

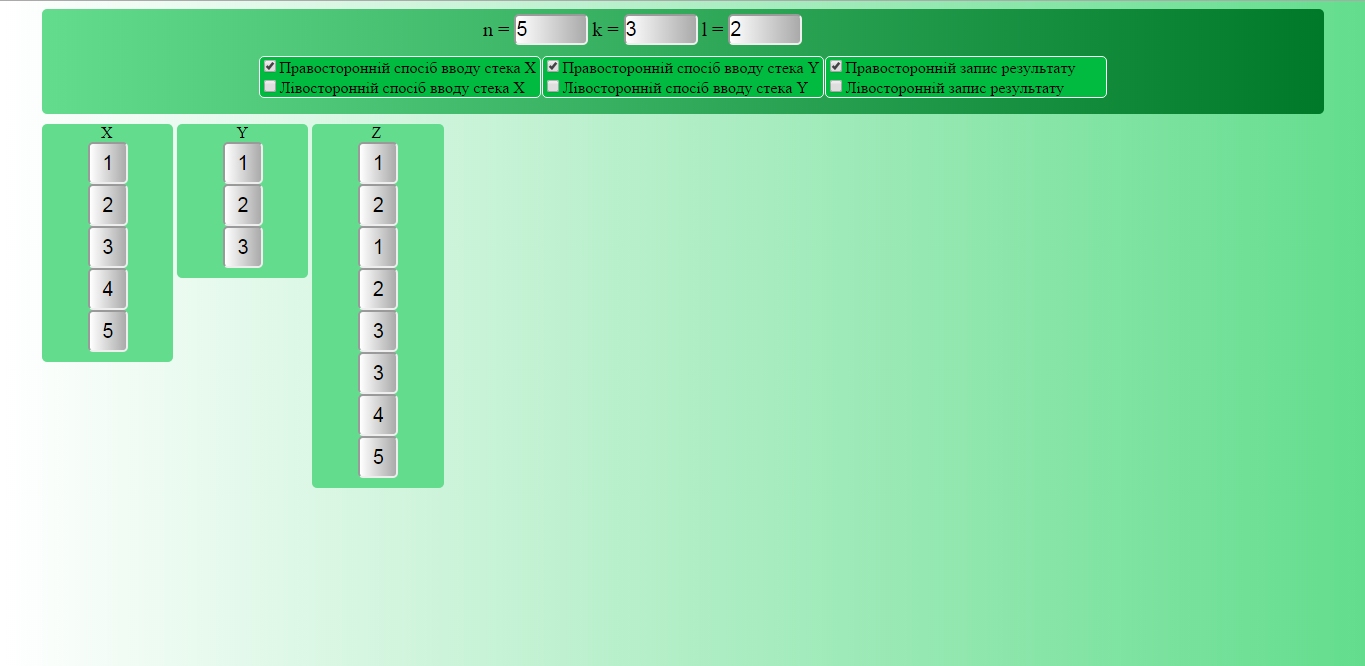


Рис. 8. Результат виконання дії додавання при правосторонньому записі стеків X, Y, Z

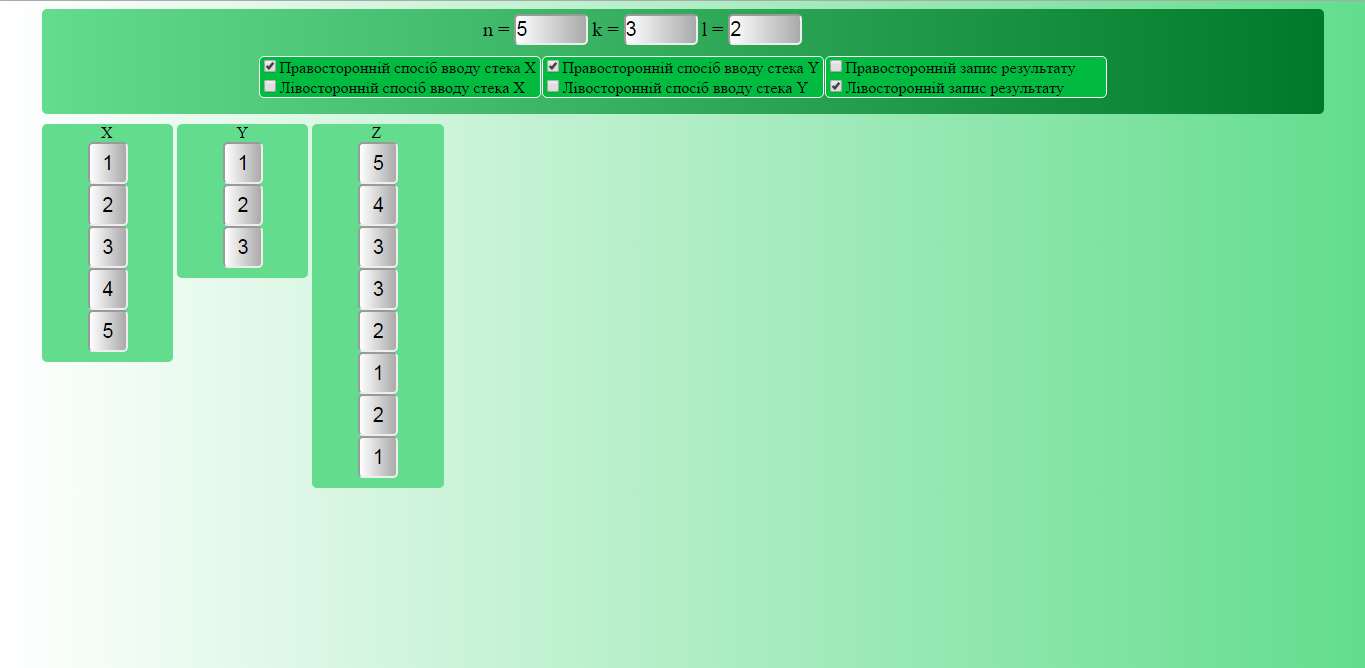


Рис. 9. Результат виконання дії додавання при правосторонньому записі стеків X, Y і лівосторонньому записі результату

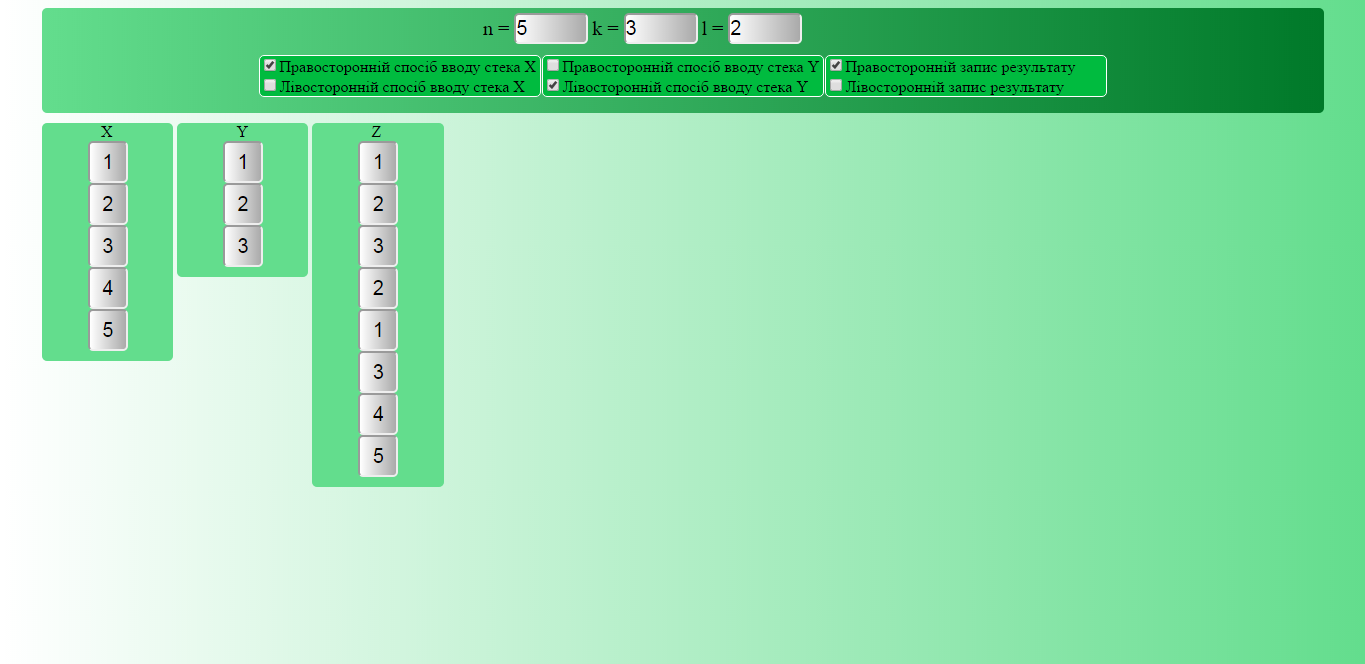


Рис. 10. Результат виконання дії додавання при правосторонньому записі стека Х, Z і лівосторонньому записі стека Y

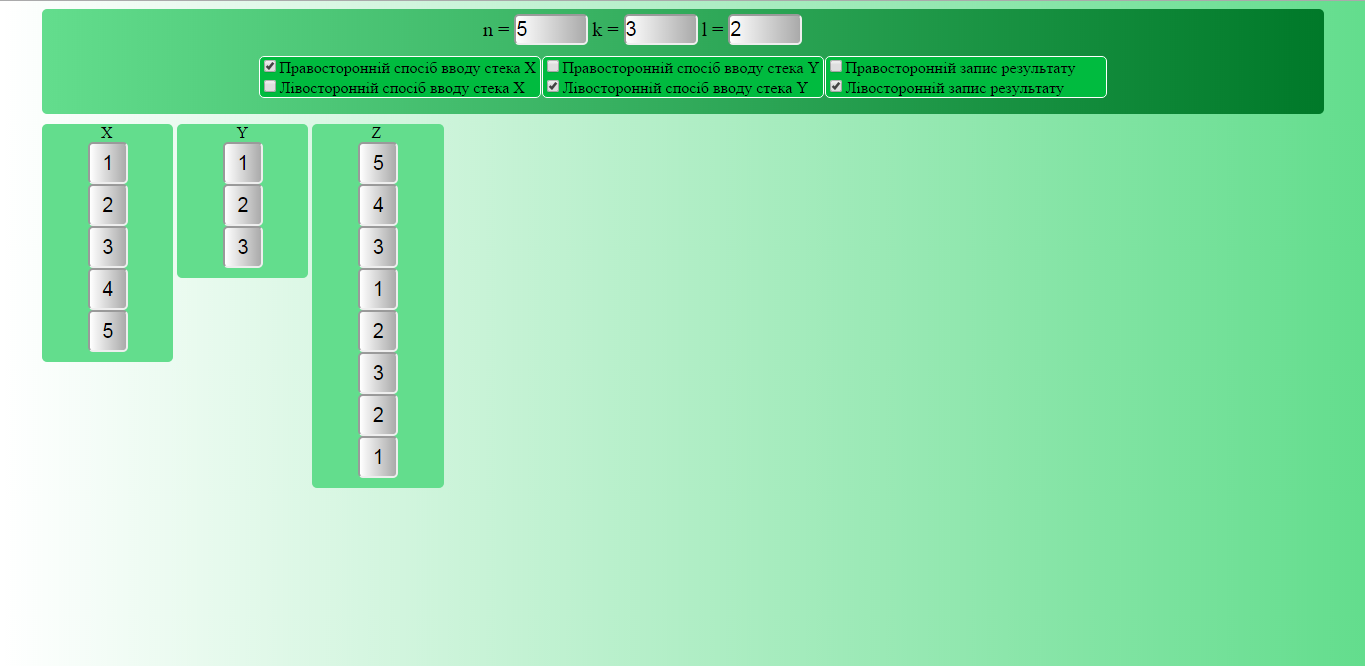


Рис. 11. Результат виконання дії додавання при правосторонньому записі стека Х і лівосторонньому записі стеків Y, Z

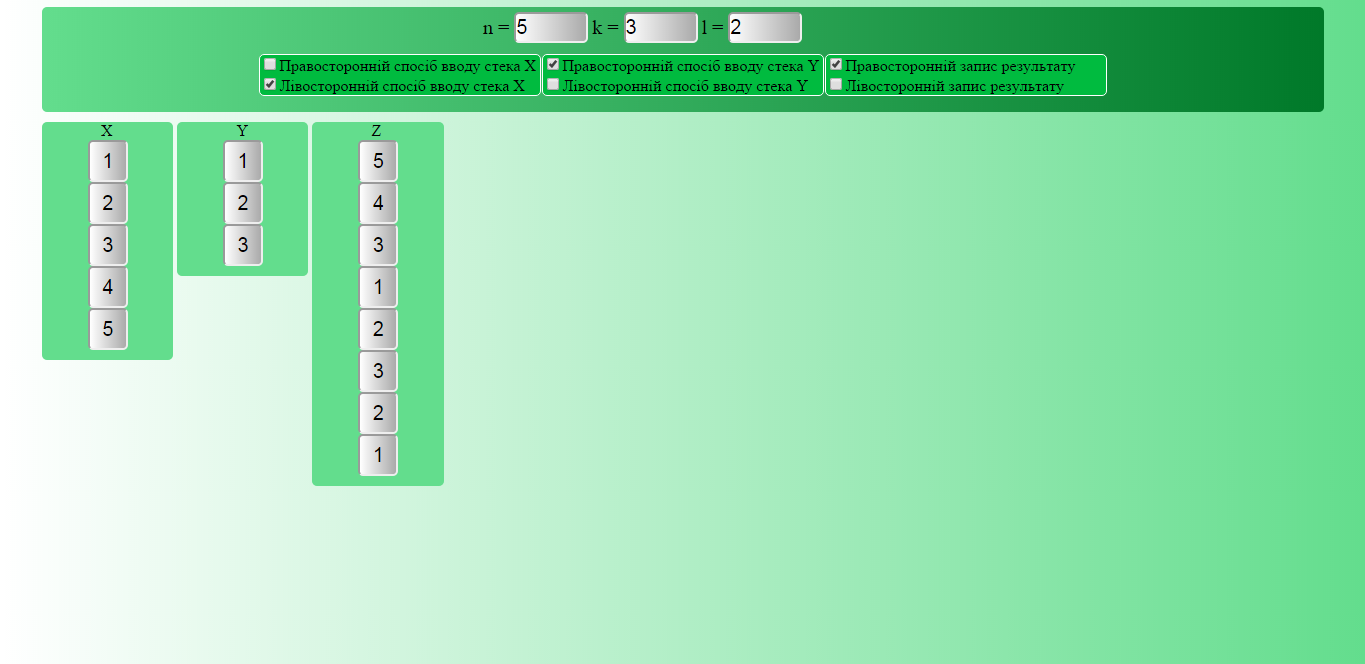


Рис. 12. Результат виконання дії додавання при лівосторонньому записі стека Х і правосторонньому записі стеків Y, Z

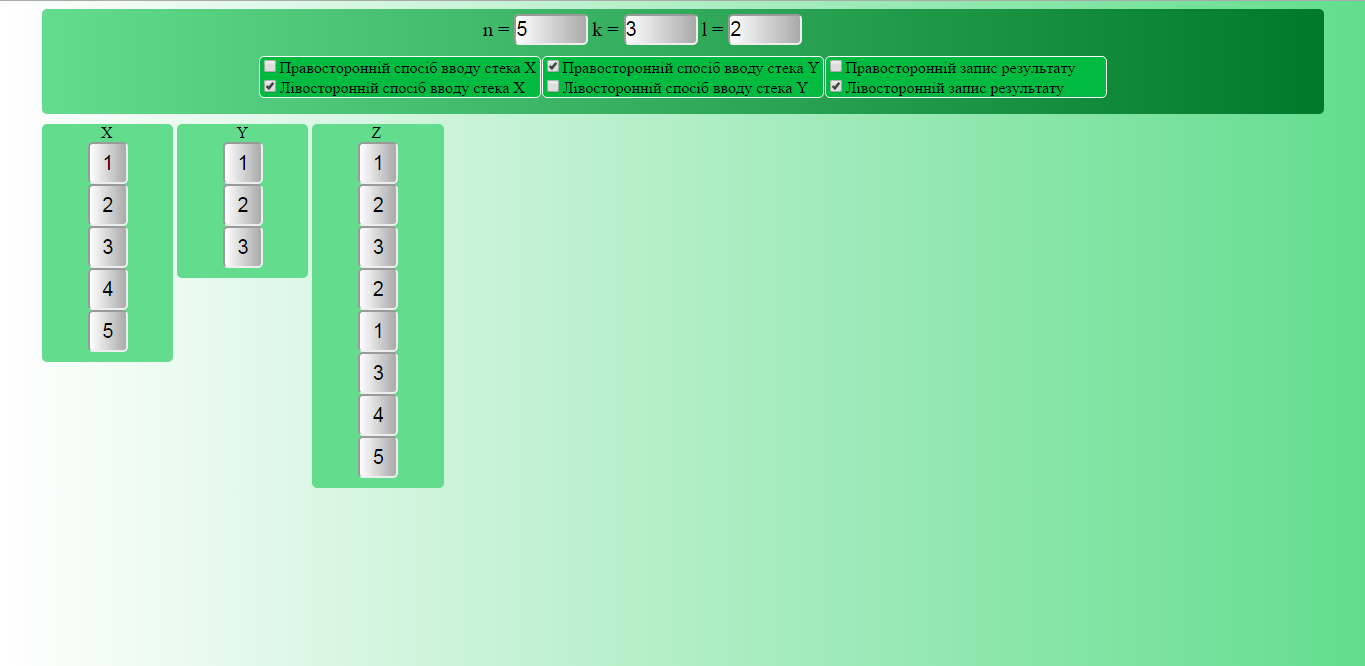


Рис. 13. Результат виконання дії додавання при лівосторонньому записі стеків Х, Z і правосторонньому записі стека Y

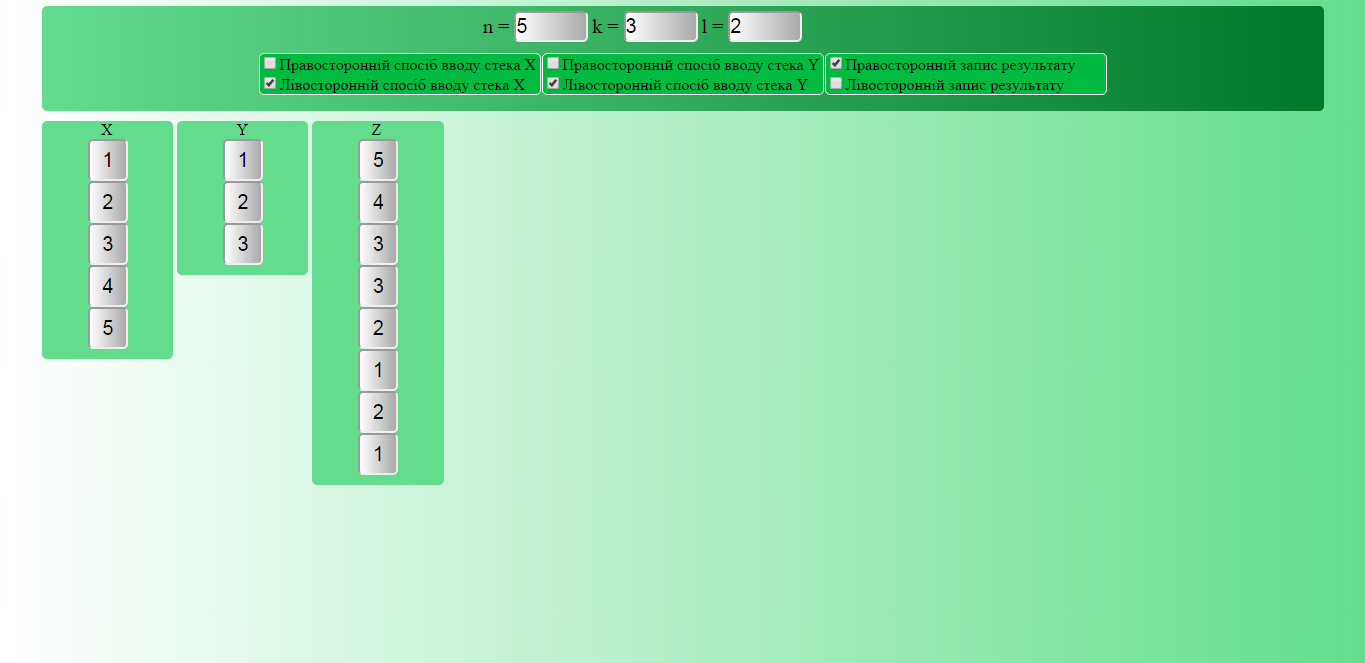


Рис. 14. Результат виконання дії додавання при лівосторонньому записі стеків Х, Y і правосторонньому записі стека Z

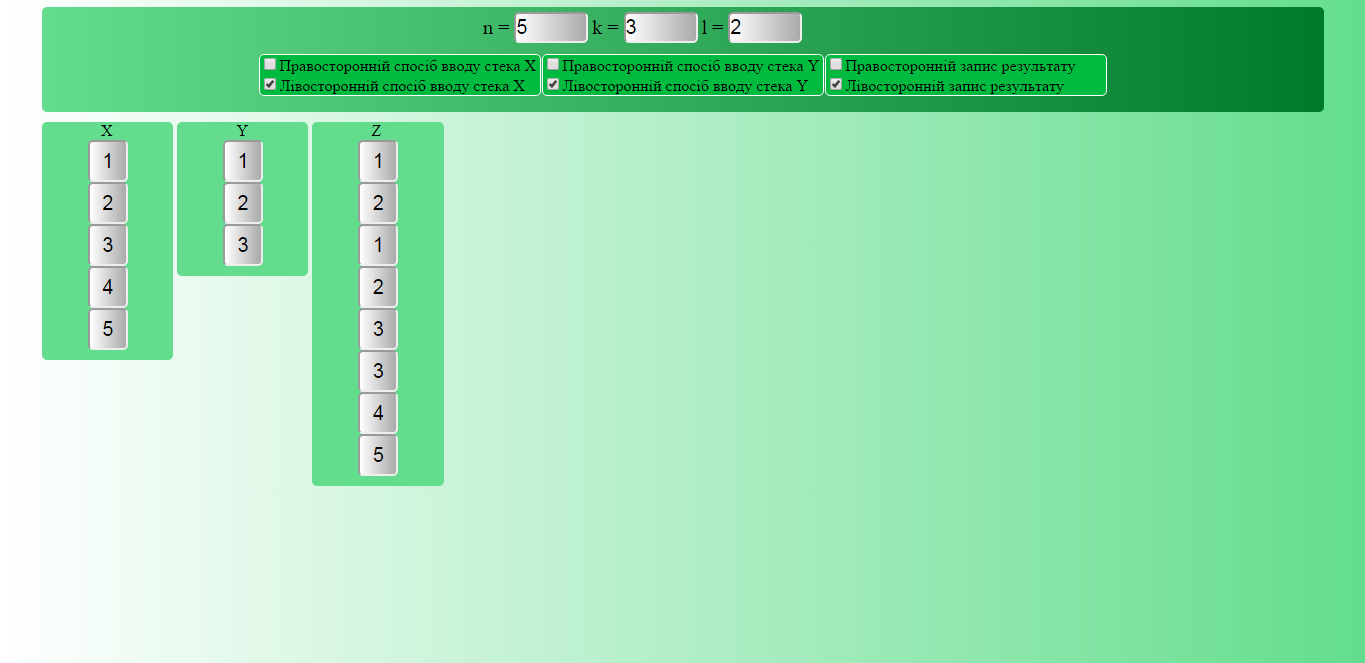


Рис. 15. Результат виконання дії додавання при лівосторонньому записі стеків X, Y, Z



Рис. 16. Результат виконання дії видалення при правосторонньому записі стеків X, Z

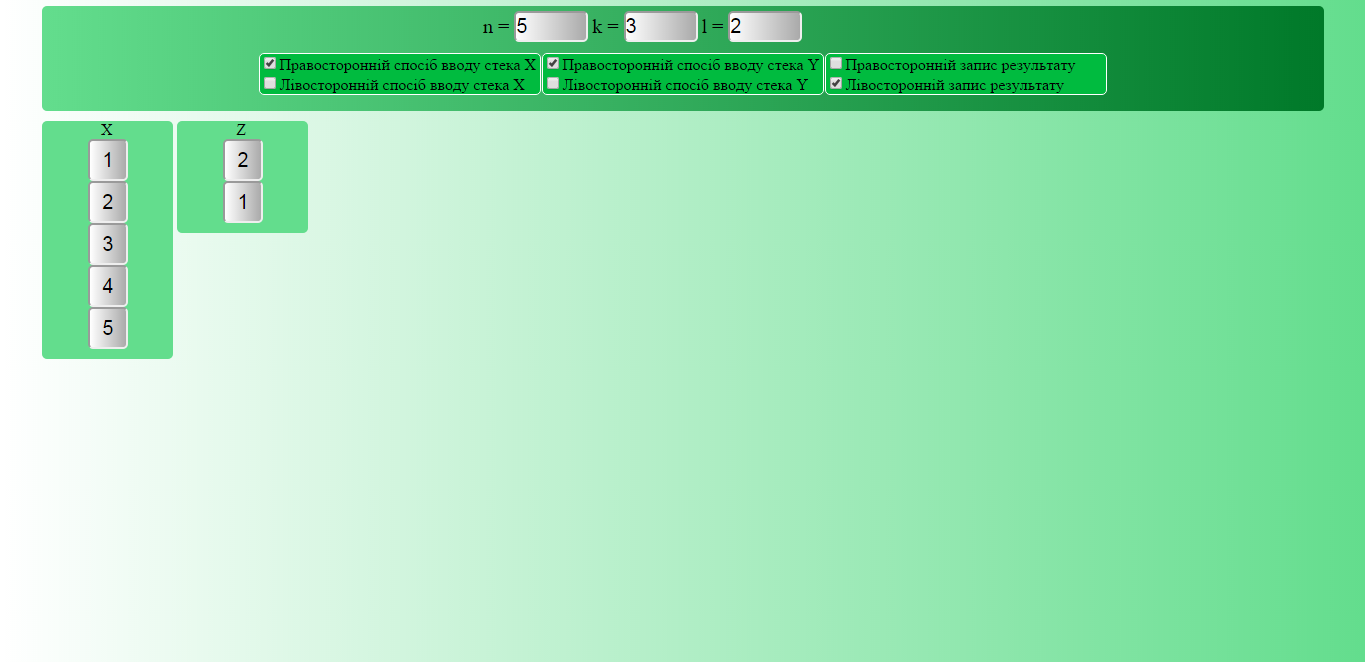


Рис. 17. Результат виконання дії видалення при правосторонньому записі стека Х і лівосторонньому записі результату

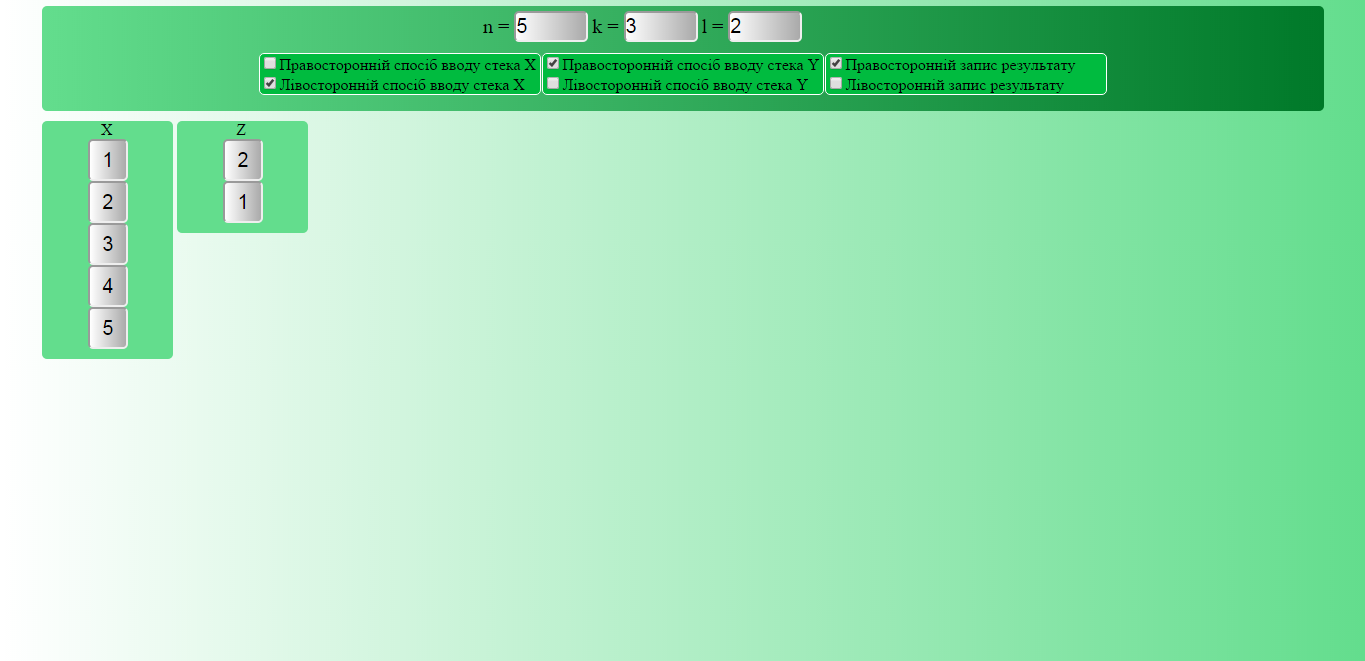


Рис. 18. Результат виконання дії видалення при лівосторонньому записі стека Х і правосторонньому записі результату

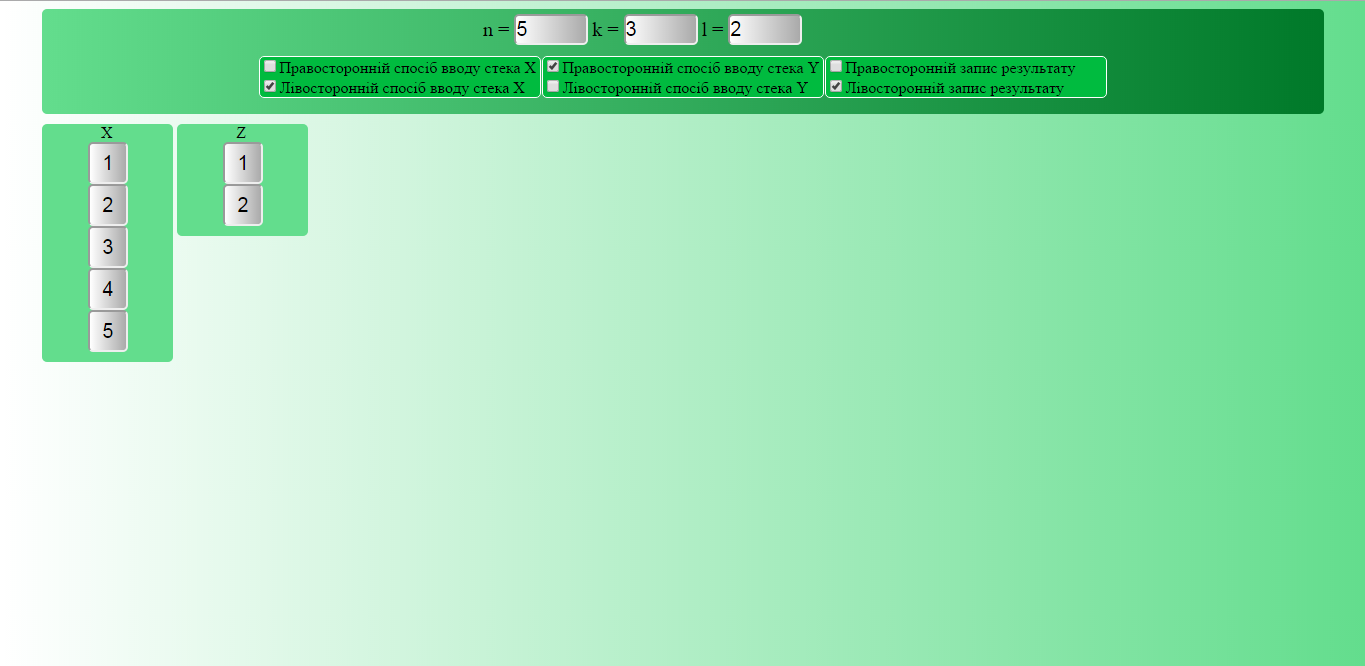


Рис. 19. Результат виконання дії видалення при лівосторонньому записі стеків X, Z