МІНІСТЕРСТВО КУЛЬТУРИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КУЛЬТУРИ І МИСТЕЦТВ

ФАКУЛЬТЕТ ЕКОНОМІКИ І ІНФОРММЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Звіт

з дисципліни «Системний аналіз»

на тему «Обробка стеків»

Виконав:

студент 3 курсу

групи КН-23

Михаленков Микола

Київ –2015

ЗМІСТ

[РОЗДІЛ 1 ПОСТАНОВКА ЗАДАЧІ 3](#_Toc431829655)

[РОЗДІЛ 2 ОПИС АЛГОРИТМУ РОЗВ’ЯЗАННЯ ЗАДАЧІ 4](#_Toc431829656)

[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](#_Toc431829657)

[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](#_Toc431829658)

[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](#_Toc431829659)

[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 4](#_Toc431829660)

[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](#_Toc431829661)

[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](#_Toc431829662)

[РОЗДІЛ 3 БЛОК-СХЕМИ ПРОГРАМНОГО ПРОДУКТУ 6](#_Toc431829663)

[РОЗДІЛ 4 ЛІСТИНГ КОДУ 7](#_Toc431829664)

[4.1 Файл index.html 7](#_Toc431829665)

[4.2 Файл style.css 7](#_Toc431829666)

[4.3 Файл script.js 8](#_Toc431829667)

[РОЗДІЛ 5 SCREENSHOTS 12](#_Toc431829668)

# 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 ОПИС АЛГОРИТМУ РОЗВ’ЯЗАННЯ ЗАДАЧІ

## 2.1 Інструменти розв’язання задачі

Задача була розв’язана за допомогою мови програмування JavaScript. Для графічного зображення даних та їх введення використовувалася мова розмітки XHTML, стандарт HTML5 та каскадні таблиці стилів. Переглянути роботу програми можна в будь-якому сучасному веб-браузері (Google Chrome, Mozilla Firefox, Safari, Opera) відкривши файл index.html.

Для полегшення роботи з графікою та приємного оку оформлення роботи використовувалася бібліотека каскадних таблиць стилів Bootstrap 3.

Для полегшення роботи з DOM-елементами та пришвидшення користувацького вводу використовувалася бібліотека Angular.js.

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.

## 2.2 Початковий стан

При завантаженні сторінки у браузері створюються 3 порожніх масиви X, Y,Z.

## 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

# 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>Title</title>

</head>

<body>

<h1>Черга. Додавання і видалення елементів</h1>

<h2>Умова:</h2>

<div id="wrapper">

<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">Apply</button>

<hr>

</div>

</body>

</html>

## 4.2 Файл style.css

#n, #k, #l, button {

width: 240px;

border-radius: 5px;

height: 40px;

font-size: 30px;

margin-right: 10px;

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

}

#result {

letter-spacing: 5px;

width: 80%;

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

}

#wrapper {

background-color: #476DD5;

padding: 10px;

border-radius: 20px;

margin-bottom: 20px;

text-align: center;

}

body {

background: linear-gradient(to left, #fff, #6D89D5);

padding: 10px;

}

span {

font-size: 30px;

}

input {

width: 40px;

height: 40px;

border-radius: 5px;

font-size: 30px;

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

}

#add, #remove {

width: 100px;

height: 100px;

border-radius: 100%;

outline: none;

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

font-size: 70px;

}

h1 {

text-align: center;

}

h2 {

margin-left: 40px;

}

## 4.3 Файл script.js

$(document).ready(function(){

var x = [];

var y = [];

var z = [];

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

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

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

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

if (n === "") {

prompt("Enter n!");

}

else if (k === "") {

prompt("Enter k!");

}

else if (l === "") {

prompt("Enter l!");

}

else {

document.body.appendChild(document.createElement('span')).innerHTML = "x = ";

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

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

element.className = "xelements";

document.body.appendChild(element);

}

document.body.appendChild(document.createElement('br'));

document.body.appendChild(document.createElement('br'));

document.body.appendChild(document.createElement('span')).innerHTML = "y = ";

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

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

element.className = "yelements";

document.body.appendChild(element);

}

document.body.appendChild(document.createElement('br'));

document.body.appendChild(document.createElement('br'));

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

}

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

element.innerHTML = '+', element.id = "add";

document.body.appendChild(element);

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

element.innerHTML = '-', element.id = "remove";

document.body.appendChild(element);

});

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

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

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

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

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

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

return input.value;

}).join( '|' );

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

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

return input.value;

}).join( '|' );

x = xelements.split("|");

y = yelements.split("|");

//

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

z[i] = x[i];

}

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

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

}

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

var index = k+l+i;

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

}

//

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

document.body.appendChild(element);

$('#add').remove();

$('#remove').remove();

document.body.appendChild(document.createElement('h2')).innerHTML = "Результат:";

document.body.appendChild(document.createElement('span')).innerHTML = " z = ";

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

element.id = "result";

document.body.appendChild(element);

var result = z.join('|');

$('#result').val('|'+result+'|');

});

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

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

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

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

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

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

return input.value;

}).join( '|' );

x = xelements.split("|");

//

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

z[i] = x[i];

}

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

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

}

//

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

document.body.appendChild(element);

$('#add').remove();

$('#remove').remove();

document.body.appendChild(document.createElement('h2')).innerHTML = "Результат:";

document.body.appendChild(document.createElement('span')).innerHTML = " z = ";

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

element.id = "result";

document.body.appendChild(element);

var result = z.join('|');

$('#result').val('|'+result+'|');

});

});

# РОЗДІЛ 5 SCREENSHOTS

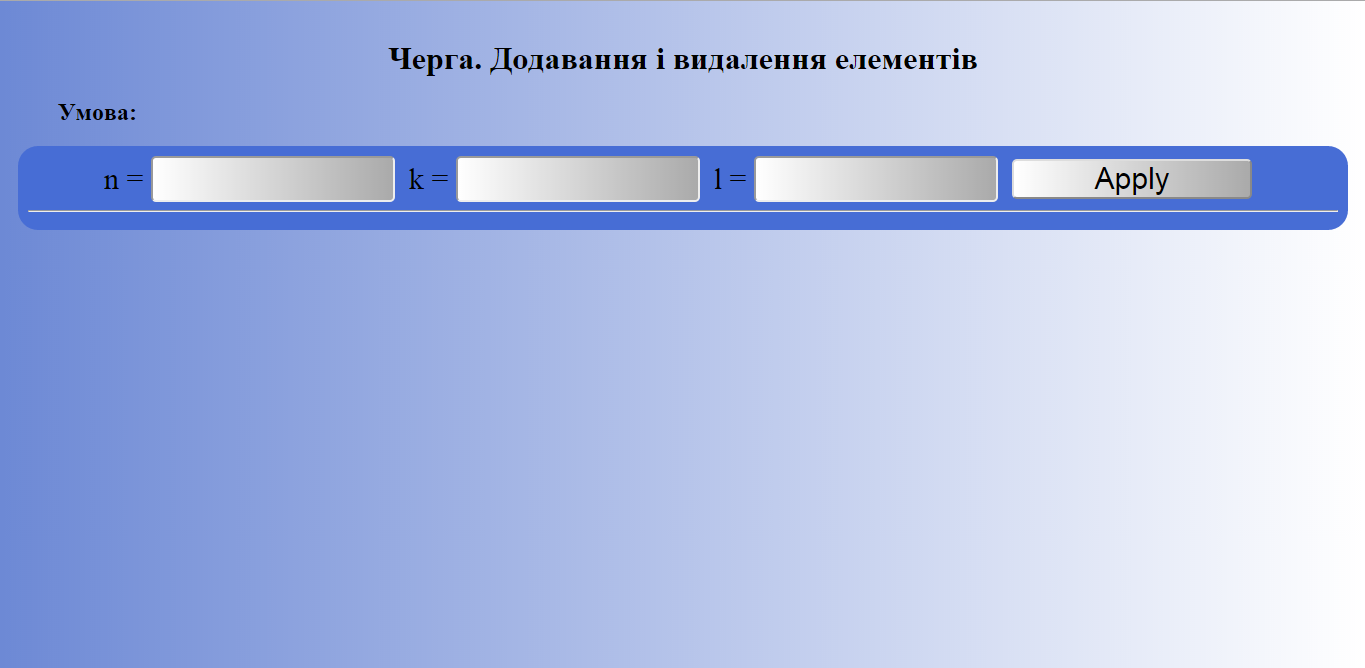


Рис. 1. Початкове вікно програми.

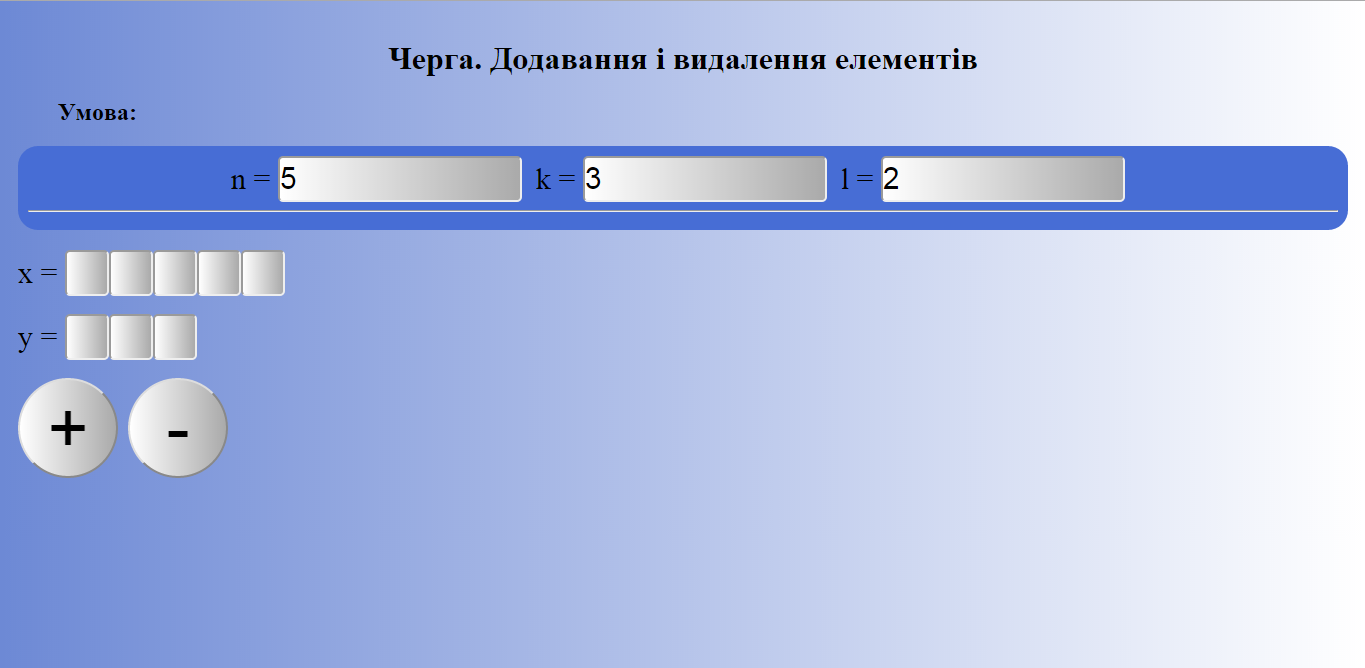


Рис. 2. Вікно програми після вводу умови.

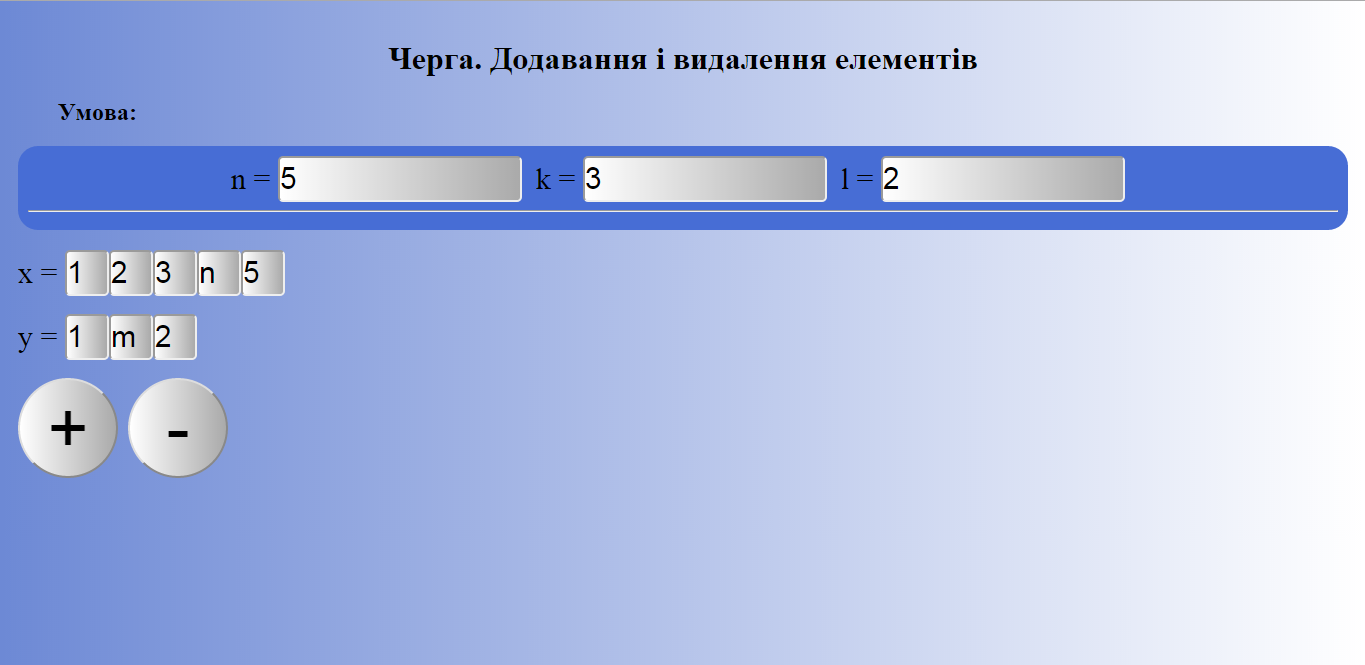


Рис. 3. Введення початкових значень черги.

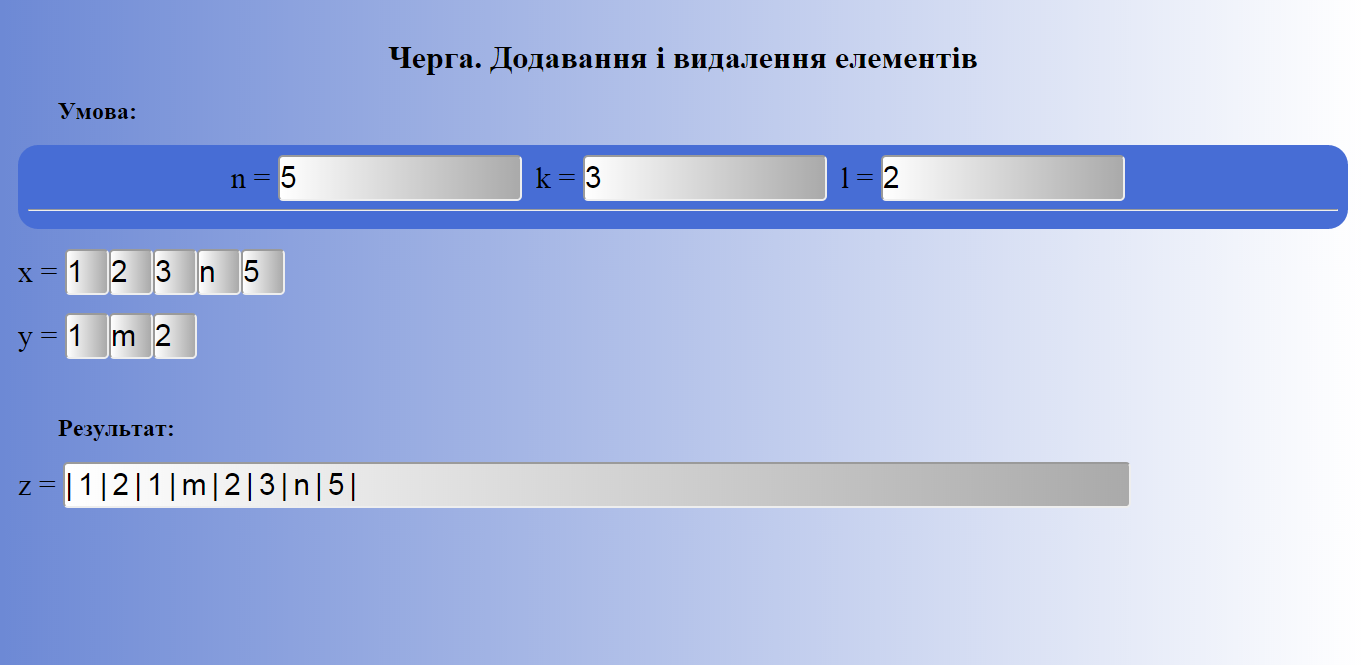


Рис. 4. Результат виконання функції додавання.

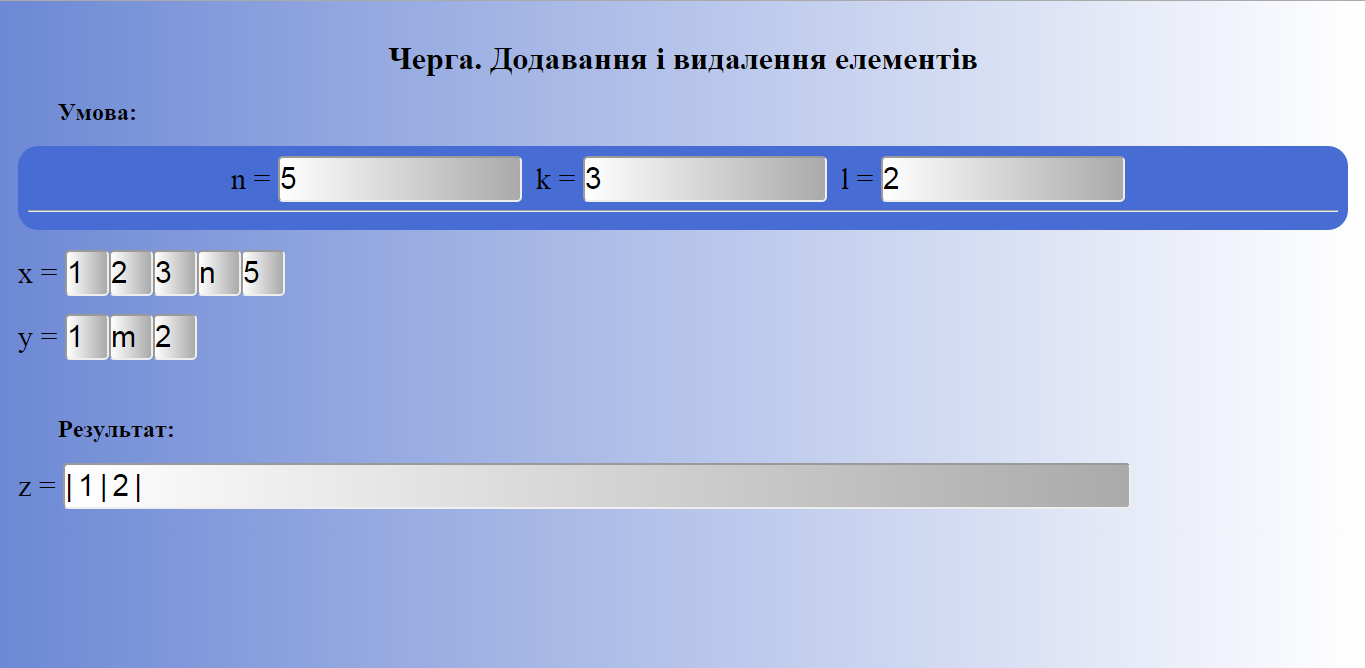


Рис. 5. Результат виконання функції видалення.