**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**

**імені ІГОРЯ СІКОРСЬКОГО”**

**ЗВІТ**

**з лабораторної роботи №4**

**з навчальної дисципліни “****Рекурсивні алгоритми”**

**Тема:**

**МЕТОДИ ПОШУКУ У МАСИВАХ**

**Виконав студент групи ТР–12**

Каркушевський Владислав

Лабораторну роботу захищено

з оцінкою \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Перевірив доцент кафедри**

\_\_\_\_\_\_\_\_\_\_\_ Андрій ОНИСЬКО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 року

**Київ 2022**

**Мета:** Метою лабораторної роботи є отримання практичних навичок в обробці масивів, у пошуку елементів масивів різними методами. Дослідження і вивчення методів пошуку ключових елементів у масивах. Здійснення порівняння та аналізу ефективності використовуваних методів пошуку

**Теоретична частина:**

Пошук – знаходження будь-якої конкретної інформації у великому обсязі раніше зібраних даних.

Дані діляться на записи, і кожний запис має хоча б один ключ. Ключ використовується для того, щоб відрізнити один запис від іншого.

Метою пошуку є знаходження всіх записів, що підходять до заданого ключа пошук

Детальніше розглянемо методи які будуть використовуватися у лабораторній

**Метод пошуку з бар’єром**

Ідея алгоритму:

– у вихідний масив потрібно тимчасово включити шукане значення.

– для одержання результату пошуку потрібно перевірити, чи дорівнює шукане значення тому елементу масиву, на якому відбулось завершення роботи алгоритму.

– навіть, якщо елемент у початковому масиві був відсутній, і зупинка була здійснена на включеному в масив зразкові, перевірка результату буде проведена для вихідного елементу масиву, який на той час замінить зразок пошуку.

– після завершення пошуку потрібно повернути в кінець масиву початкове значення .

Роль **бар’єрного елементу** виконує включений в масив зразок пошуку.

Додаткові операції по установці і зняттю бар'єра окупаються спрощенням циклу, у якому витрачається основний час при пошуку. Особливо це позначиться при великих розмірах масиву.

Для практичної реалізації алгоритму потрібно застосувати цикл з лічильником, а пошук проводити до кінця масиву, це дасть можливість знайти всі відповідні елементи. У такому випадку кількість перевірок дорівнює – N.

**Пошук послідовності елементів в масив**

Одне з найпростіших завдань пошуку інформації – пошук точно заданого підрядка у рядку. Проте, це завдання надзвичайно важливе – воно застосовується в текстових редакторах, СУБД, пошукових машинах тощо. Пошук рядка формально визначається в такий спосіб. Нехай заданий масив Т з N елементами і масив W з M елементами, причому 0 < M ≤ N. Пошук рядка виявляє перше входження W у Т, результатом будемо вважати індекс і, що вказує на перший з початку рядка (з початку масиву Т) збіг зі зразком (словом

**Алгоритм прямого (послідовного) пошуку**

**Ідея алгоритму:**

1) i = 1

2) порівняти i-й символ масиву T з першим символом масиву W

3) збіг → порівняти другий символ і так далі

4) розбіжність → i = i + 1 і перехід до пункту 2.

**Умова закінчення алгоритму:**

1) підряд М порівнянь вдалі

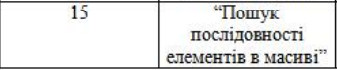
2) i + M > N, тобто слово не знайдене.

**Завдання**

**Варіант 15**

1.Знайти заданий елемент у невпорядкованому масиві (не менше 10х10) за допомогою методу пошуку з бар’єром.

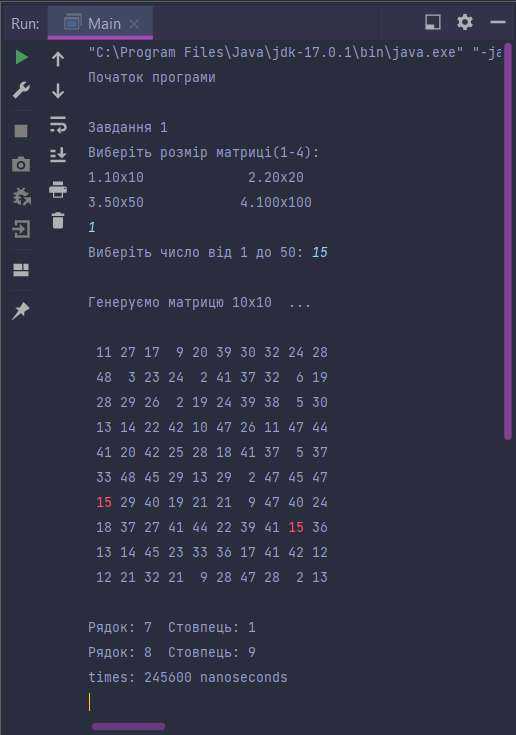
2. Знайти заданий елемент у впорядкованому масиві (не менше 10х10) згідно варіантів за таким принципом

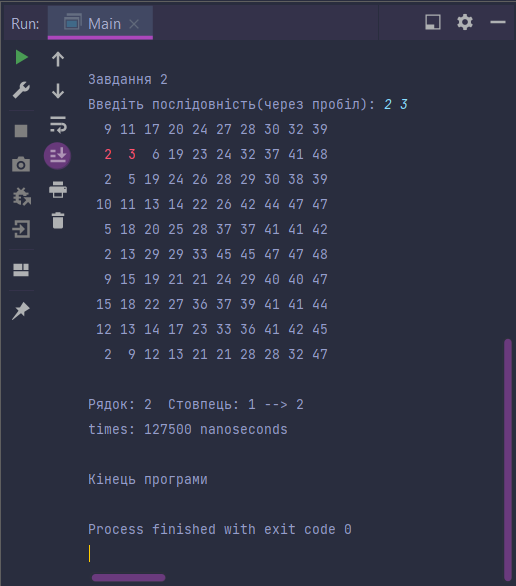


**Код програми:**

*import* java.util.Scanner;  
  
*public class* Main {  
 *public static void* main(String[] args) {  
 System.***out***.println("Початок програми\n");  
 System.***out***.println("Завдання 1");  
 System.***out***.println("Виберіть розмір матриці(1-4):\n1.10x10 2.20x20\n3.50x50 4.100x100");  
  
 Scanner sc = *new* Scanner(System.***in***);*//створюємо сканер  
  
 int* size = sc.nextInt();*//за допомогою сканера вибираємо розмір матриці* System.***out***.print("Виберіть число від 1 до 50: ");  
 *int* key = sc.nextInt();  
  
 size = *info*(size);  
 *int*[][] matrixStart = *makeMatrix*(size);*//викликаємо метод для створення матриці  
 print1*(matrixStart, key);  
 *task1*(matrixStart,key);  
  
 System.***out***.println("\nЗавдання 2");  
 System.***out***.print("Введіть послідовність(через пробіл): ");  
 Scanner scan = *new* Scanner(System.***in***);  
 String str = scan.nextLine();  
 *insertionSort*(matrixStart);  
 *print2*(matrixStart,str);  
 *task2*(matrixStart,str);  
 System.***out***.println("\nКінець програми");  
  
 }  
  
 *public static int* barrier(*int* key, *int*[] num) {  
  
 *int* finaly = num[num.length - 1];*//зберігаємо останній елемент масиву у змінній finaly* num[num.length - 1] = key;*//в останній елемент масиву розміщуємо шукане значення  
 int* j;  
 *for* (j = 0; num[j] != key; j++);*//за допомогою циклу шукаємо значення у масиві* num[num.length - 1] = finaly;  
 *if* (j != num.length - 1 || key == finaly) {*//якщо значення знайдено повертаємо його індекс  
 return* j;  
 }  
  
 *return* -1;*//якщо ні повертаємо -1* }  
  
*public static void* task1(*int* [][] matrix,*int* key){*//створюємо метод для першого завдання  
 long* start = System.*nanoTime*();*//розпочинаємо вимір часу  
 boolean* var = *false*;  
 *for* (*int* i = 0; i < matrix.length; i++) {  
  
 *int* j = *barrier*(key,matrix[i]);  
 *if*( j != -1) {  
 System.***out***.printf("Рядок: %d Стовпець: %d\n", i + 1, j + 1);  
 var = *true*;  
 }  
 }  
 *if*(!var){  
 System.***out***.print("\u001B[31m");  
 System.***out***.println("Елементів не знайдено");  
 System.***out***.print("\u001B[0m");  
 }  
 *long* finish = System.*nanoTime*();*//закінчуємо вимір часу* System.***out***.printf( "times: %d nanoseconds\n", finish-start);*//виведення часу перестановки матриці*}  
  
 *public static void* insertionSort(*int*[][] matrix) {*//створюємо метод для сортування вставками  
  
 int* [] nums ;  
 *for* (*int*[] ints : matrix) {  
 nums = ints;  
 *int* j;  
 *for* (*int* i = 1; i < nums.length; i++) {  
 *int* x = nums[i];  
 j = i;  
 *while* (j > 0 && nums[j - 1] > x) {  
 *int* temp = nums[j];  
 nums[j] = nums[j - 1];  
 nums[j - 1] = temp;  
 j--;  
 }  
 nums[j] = x;  
 }  
  
 }  
  
  
 }  
  
 *public static int* sequence(String str, *int* [] arr ){*//створюємо метод пошуку послідовності елементів в масиві* String[] strMassive = str.split(" ");*//через сканер передаємо рядок, і розміщуємо значення задані через пробіл в масив рядків  
 int* [] numMassive = *new int*[strMassive.length];*//створюємо масив чисел з довжиною масиву рядків  
 for*(*int* i = 0; i < strMassive.length; i++){  
 numMassive[i] = Integer.*parseInt*(strMassive[i]);*//переміщуємо з масиву рядків в масив чисел наші дані* }  
  
 *//проводимо операції для знаходження послідовності елементів в масиві  
 for* (*int* i = 0; i < arr.length - (numMassive.length - 1) ; i++) {  
 *if* (arr[i] == numMassive[0]) {  
 *boolean* bool = *true*;  
 *for* (*int* j = i + 1, k = 1; j < arr.length && k < numMassive.length; j++, k++) {  
 *if* (arr[j] != numMassive[k]) {  
 bool = *false*;  
 *break*;  
 }  
 }  
 *if* (bool) *return* i;*//якщо значення знайдено повертаємо індекс першого елемента* }  
 }  
 *return* -1;*//якщо ні повертаємо -1* }  
  
  
*public static void* task2(*int* [][] matrix,String str){*//створюємо метод для другого завдання  
 long* start = System.*nanoTime*();*//розпочинаємо вимір часу* String[] strMassive = str.split(" ");  
  
 *boolean* var = *false*;  
 *for* (*int* i = 0; i < matrix.length; i++) {  
  
 *int* j = *sequence*(str,matrix[i]);  
 *if*( j != -1) {  
 System.***out***.printf("Рядок: %d Стовпець: %d --> %d\n", i + 1, j + 1,j+strMassive.length);  
 var = *true*;  
 }  
 }  
 *if*(!var){  
 System.***out***.print("\u001B[31m");  
 System.***out***.println("Елементів не знайдено");  
 System.***out***.print("\u001B[0m");  
 }  
 *long* finish = System.*nanoTime*();*//закінчуємо вимір часу* System.***out***.printf( "times: %d nanoseconds\n", finish-start);*//виведення часу перестановки матриці*}  
  
 *public static int*[][] makeMatrix( *int* size ){*//створюємо метод для створення матриці  
  
 int*[][] a = *new int* [size][size];  
  
 *for*(*int* i = 0 ; i< a.length; i++){  
 *for*(*int* j = 0 ; j< a[0].length;j++){  
 a[i][j] = (*int*) (Math.*random*()\*49+1);*//заповнюємо матрицю числами в діапазоні [1;50]* }  
 }  
 *return* a;  
 }  
  
  
  
 *public static int* info(*int* size ){*//створюємо метод для вибору розміру матриці  
  
 if*(size == 1){  
 size = 10;  
 System.***out***.println("\nГенеруємо матрицю 10x10 ...\n");  
 }*else if*(size == 2 ){  
 size = 20;  
 System.***out***.println("\nГенеруємо матрицю 20x20 ...\n");  
 }*else if*(size == 3 ){  
 size = 50;  
 System.***out***.println("\nГенеруємо матрицю 50x50 ...\n");  
 }*else if*(size == 4 ){  
 size = 100;  
 System.***out***.println("\nГенеруємо матрицю 100x100 ...\n");  
 }*else*{  
 System.***out***.print("\u001B[31m");  
 System.***out***.println("error");  
 System. *exit*(0);  
 }  
  
 *return* size;  
 }  
 *public static void* print1(*int* [][] matrix,*int* var){*//створюємо метод для різнобарвного виведення матриці для першого завдання  
  
  
 for* (*int*[] ints : matrix) {  
 *for* (*int* j = 0; j < matrix[0].length; j++) {  
 *if* (ints[j] == var) {  
 System.***out***.print("\u001B[31m");  
 } *else* {  
 System.***out***.print("\u001B[0m");  
 }  
 System.***out***.printf("%3d", ints[j]);  
 }  
  
 System.***out***.println();  
 }  
  
 System.***out***.println("\u001B[0m");  
 }  
  
 *public static void* print2(*int* [][] matrix,String var){*//створюємо метод для різнобарвного виведення матриці для другого завдання* String[] strMassive = var.split(" ");  
  
 *for* (*int*[] ints : matrix) {  
 *int* k = *sequence*(var, ints);  
  
 *if* (k == -1) {  
 *for* (*int* m = 0; m < matrix.length; m++) {  
 System.***out***.print("\u001B[0m");  
 System.***out***.printf("%3d", ints[m]);  
 }  
 } *else* {  
 *for* (*int* j = 0; j < k; j++) {  
 System.***out***.print("\u001B[0m");  
 System.***out***.printf("%3d", ints[j]);  
 }  
 *for* (*int* p = k; p < k + strMassive.length; p++) {  
 System.***out***.print("\u001B[31m");  
 System.***out***.printf("%3d", ints[p]);  
 }  
 *for* (*int* l = k + strMassive.length; l < matrix.length; l++) {  
 System.***out***.print("\u001B[0m");  
 System.***out***.printf("%3d", ints[l]);  
 }  
  
 }  
 System.***out***.println();  
 }  
  
 System.***out***.println("\u001B[0m");  
 }  
  
}

**Результат виконання програми:**

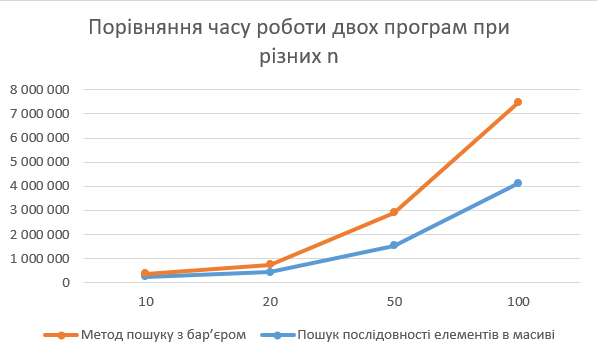
****



**Метод пошуку з бар’єром**

**Пошук послідовності елементів в масиві**

|  |  |
| --- | --- |
| Розмір | Час |
| 10х10 | 245 600 |
| 20х20 | 443 100 |
| 50х50 | 1 528 800 |
| 100х100 | 4 125 400 |
| Розмір | Час |
| 10х10 | 127 500 |
| 20х20 | 301 700 |
| 50х50 | 1 377 600 |
| 100х100 | 3 352 800 |



**Висновок:**

На цій лабораторній роботі було отримано практичні навички в обробці масивів, у пошуку елементів масивів різними методами. Дослідження і вивчення методів пошуку ключових елементів у маси. Здійснення порівняння та аналізу ефективності використовуваних методів пошуку. Побудовано графік порівняння часу роботи двох програм при різних n.