**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"**

Інститут **ІКНІ**

Кафедра **ПЗ**

**ЗВІТ**

До лабораторної роботи № 10

**З дисципліни:** *“Алгоритми та структури даних”*

**На тему:** *“Бінарний пошук в упорядкованому масиві”*

**Лектор:**

доц. каф. ПЗ

Коротєєва Т.О.

**Виконав:**

ст. гр. ПЗ-22

Солтисюк Д.А.

**Прийняв:**

асист. каф. ПЗ

Франко А.В.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2022 р.

∑= \_\_\_\_\_ .

Львів – 2022

**Тема роботи:** Бінарний пошук в упорядкованому масиві

**Мета роботи:** Навчитися застосовувати алгоритм бінарного пошуку при розв’язуванні задач та перевірити його ефективність на різних масивах даних. Експериментально визначити складність алгоритму.

**ТЕОРЕТИЧНІ ВІДОМОСТІ**

Бінарний, або двійковий пошук – алгоритм пошуку елементу у відсортованому масиві. Це класичний алгоритм, ще відомий як метод дихотомії (ділення навпіл).

Якщо елементи масиву впорядковані, задача пошуку суттєво спрощується. Згадайте, наприклад, як Ви шукаєте слово у словнику. Стандартний метод пошуку в упорядкованому масиві – це метод поділу відрізка навпіл, причому відрізком є відрізок індексів *1..n*. Дійсно, нехай масив *A* впорядкований за зростанням і*m (k < m < l)* – деякий індекс. Нехай *Buffer* =*A[m].* Тоді якщо *Buffer* *> b*, далі елемент необхідно шукати на відрізку *k..m-1*, а якщо *Buffer* *< b* – на відрізку *m+1..l*.

Для того, щоб збалансувати кількість обчислень в тому і іншому випадку, індекс *m* необхідно обирати так, щоб довжина відрізків *k..m*, *m..l* була (приблизно) рівною. Описану стратегію пошуку називають *бінарним пошуком.*

*b* – елемент, місце якого необхідно знайти. Крок бінарного пошуку полягає у порівнянні шуканого елемента з середнім елементом *Buffer* = *A[m]* в діапазоні пошуку *[k..l]*. Алгоритм закінчує роботу при *Buffer* *= b*(тоді *m* – шуканий індекс). Якщо *Buffer* *> b*, пошук продовжується ліворуч від *m*, а якщо *Buffer* *< b* – праворуч від *m*. При *l < k* пошук закінчується, і елемент не знайдено.

**ІНДИВІДУАЛЬНЕ ЗАВДАННЯ**

1. Програма повинна забезпечувати автоматичну генерацію масиву цілих чисел (кількість елементів масиву вказується користувачем) та виведення його на екран;

2. Визначте кількість порівнянь та порівняйте ефективність на декількох масивах різної розмірності заповнивши табл. 1.

3. Представте покрокове виконання алгоритму пошуку.

4. Побудуйте графік залежності кількості порівнянь від кількості елементів масиву у Excel. Побудуйте у тій же системі координат графіки функцій y=n та y=log2(n). Дослідивши графіки, зробіть оцінку кількості С(n) порівнянь алгоритму бінарного пошуку.

5. З переліку завдань виконайте індивідуальне завдання запропоноване викладачем.

**Варіант 22 (7):**

Дано одновимірний масив цілих чисел A[і], де і =1,2,…,n. Знайти елемент що дорівнює різниці максимального та мінімального елементів. Вивести позицію шуканого елемента.

**ВИКОНАННЯ РОБОТИ**

Код програми:

**import** random

**def** gen\_random\_int\_array(n):

**return** [random.randint(1, n) **for** \_ **in** range(0, n)]

**def** gen\_input\_data():

n = int(input("Enter array size: ") **or** 100)

arr = gen\_random\_int\_array(n)

arr.sort()

min\_max\_diff\_el = arr[-1] - arr[0]

**print**(f"Sorted generated array: {arr}")

**print**(f"Min and max element difference: {min\_max\_diff\_el}")

**return** arr, min\_max\_diff\_el

**def** binary\_search(arr, key):

left, right = 0, len(arr) - 1

**while** left <= right:

mid = (left + right) // 2

**print**(f"Left={left}, Middle={mid}, Right={right}")

**if** arr[mid] > key:

right = mid - 1

**elif** arr[mid] < key:

left = mid + 1

**else**:

**return** mid

**if** \_\_name\_\_ == "\_\_main\_\_":

arr, min\_max\_diff\_el = gen\_input\_data()

**print**()

**print**("Searching in array for minmax diff element with binary search...")

**print**()

**print**("Binary search process:")

found\_idx = binary\_search(arr, min\_max\_diff\_el)

**print**()

**print**(

f"Found under id: {found\_idx}"

**if** found\_idx

**else** "Minmax diff element was not found in the provided array"

)

**ПРОТОКОЛ РОБОТИ**

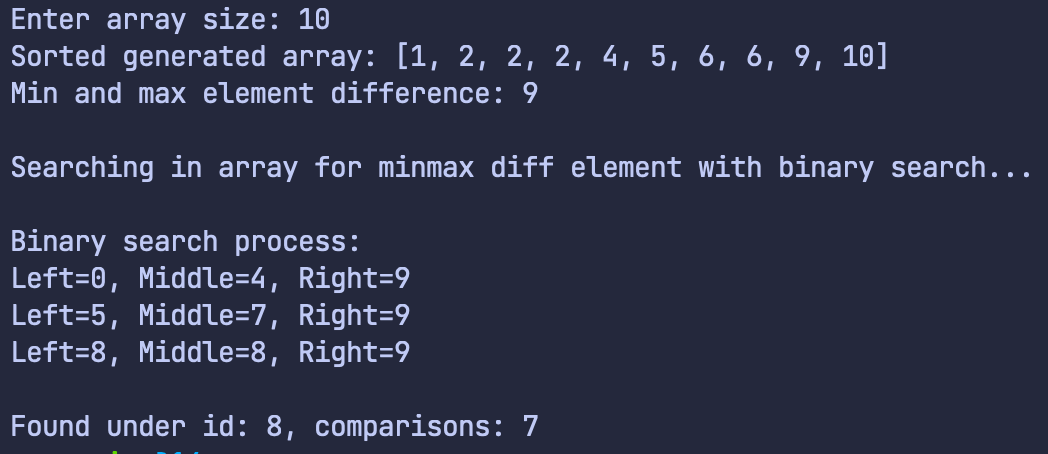
****

Рис. 1. Загальний вигляд роботи програми

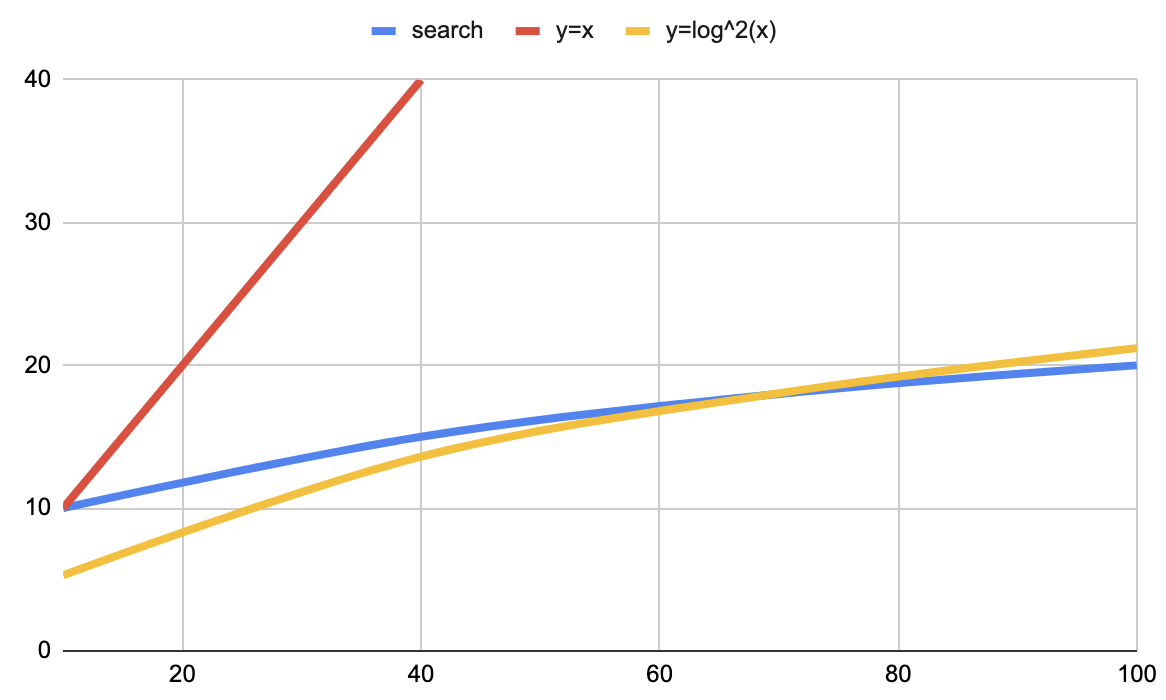


Рис. 2. Графік для нагального огляду роботи програми

**ВИСНОВКИ**

Під час виконання лабораторної роботи я навчився застосовувати алгоритм бінарного пошуку при розв’язуванні задач та перевірити його ефективність на різних масивах даних. Експериментально визначив складність алгоритму.