МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ  
«ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ

ЛУЦЬКОГО НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ»

Випускова циклова комісія Компʼютерних систем та інформаційних технологій

**Звіт**

**до практичної роботи №1**

“Алгоритми та структури даних”

Виконав: студент **2** курсу, групи **21-КІ**

Спеціальності 123 комп’ютерна інженерія

**Сумарук Михайло**

Перевірив икладач: Володимир Остапюк

Оцінка\_\_\_\_\_\_\_\_\_\_\_\_

Луцьк – 2024

1. **Алгоритм**- набір послідовних дій, інструкцій націлені на розв’язання задачі

2. Алгоритм бінарного пошуку - це ефективний алгоритм пошуку, який використовується для пошуку певного елемента в відсортованому масиві або списку даних.   
 2.1 **Основні принципи**: ділення навпіл масиву або списку,працює лише у впорядкованому списку або масиву,алгоритм може бути реалізований як рекурсивно, так і ітеративно.  
 2.2 **Призизначення**: використовується для пошуку елементу у впорядкованому списку, особливо ефективний при великих обсягах даних.

3. Програмний код:

from random import randint  
  
a = [randint(1, 100) for i in range(15)]  
a.sort()  
print(a)  
print("Введіть число для пошуку: ")  
b = int(input())  
  
c = 1  
c = 1  
d = len(a) - 1  
f = (c + d) // 2  
  
while a[f] != b:  
 if b > a[f]:  
 c = f + 1  
 else:  
 d = f - 1  
 f = (c + d) // 2  
 if c >= d:  
 break  
  
if b == a[f]:  
 print('Кількість ітерацій =', f)  
else:  
 print('Вашого числа немає у списку')

4. Результатом швидкості виконання операцій є кількість ітерацій, тобто кількість необхідних операцій для досягнення завдання.

Якщо брати лінійний тип пошуку то кількість операцій = самому числу.

Бінарний пошук має логарифмічну складність O(log⁡n)O(\log n)O(logn), тому навіть для масиву в 1 мільйон елементів час виконання залишається низьким.

5.

5.1 **Переваги**: швидкість для великих масивів, ефективність для відсортованих даних.

**Недоліки**: вимога до відсортованості масиву, складність для динамічних структур.

5.2 Бінарний пошук є значно ефективнішим, коли масив відсортований і великий. Однак, лінійний пошук залишається універсальним і простим інструментом для будь-яких даних і структур, особливо коли масив невідсортований або невеликий. У великих масивах та списках

5.3 Якщо середній елемент більший за цільовий, ми повинні продовжити пошук праворуч (у менших елементах).

5.4 Якщо середній елемент менший, ми шукаємо ліворуч (у більших елементах).

5.5 Лінійний алгоритму пошуку , однак він ефективний тільки у малих списках.

**Висновок**: у цій практичній роботі я вивчав особливості роботи бінарного пошуку для мови програмування Python. Визначивши його переваги та недоліки, можна сказати що бінарний алгоритм пошуку є дуже ефективним, але лінійний досі лишається самим простим та елементарним. Найбільшою проблемою бінарного пошуку є умова відсортованості елементів, лінйному пошуку це непотрібно, у будь-якому випадку результат буде знайдено, все лише впирається у ліміт часу. Також використання бінарного пошуку у малих об’ємах даних є безглуздим, так як його використання для такої малої робити зробить його недоцільним.