МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ   
ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕЛЕКТРИЧНОЇ ІНЖЕНЕРІЇ   
ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА АВТОМАТИЗАЦІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

НАВЧАЛЬНА ДИСЦИПЛІНА  
 «Алгоритми та методи обчислень»

ЗВІТ

З ПРАКТИЧНИХ РОБІТ

Виконав:

студент групи КН-23-1

Ярковий Т.С.

Кременчук 2023

**Практична робота № 3**

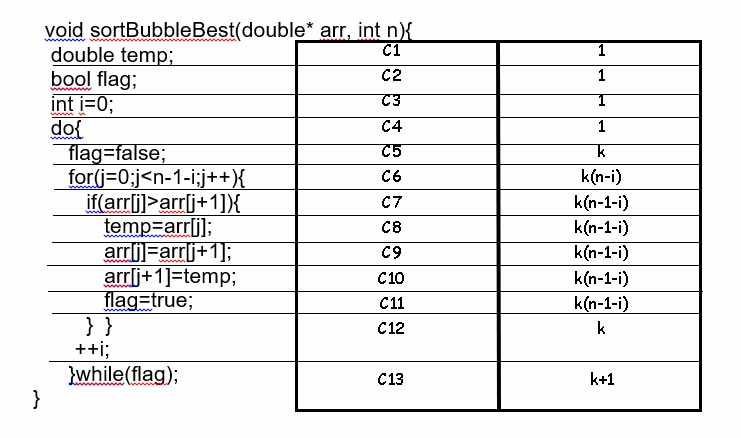
**Тема. Алгоритми сортування та їх складність. Порівняння алгоритмів сортування**

**Мета:** опанувати основні алгоритми сортування та навчитись методам аналізу їх асимптотичної складності.

**Завдання**

1. . Вивчити самостійно і записати (будь-яким способом) алгоритм бульбашкового сортування. Оцінити асимптотику алгоритму сортування методом бульбашки в найгіршому і в найкращому випадку. Порівняти за цими показниками бульбашковий алгоритм з алгоритмом сортування вставлянням. Чому на практиці бульбашковий алгоритм виявляється менш ефективним у порівнянні з сортуванням методом зливанням?

Сортування бульбашкою на C++:



k – кількість проходів. У найкращому випадку, якщо масив вже відсортований k = 1. У найгіршому k=n.

i – число, яке буде мінятися під час циклу.

Накращий випадок: T(n)=C1+C2+C3+C4+C5+C6(n-i) +C7(n-1-i) +C8(n-1-i) +C9(n-1-i) +C10(n-1-i) +C11(n-1-i)+C12+2C13=

(C1+C2+C3+C4+C5+C12+2C13)+(n-1-i)(C6+C7+C8+C9+C10+C11)

Для О нотації першими та останніми дужками, числами біля n можна знехтувати і складність буде O(n).

Найгірший випадок: T(n)=C1+C2+C3+C4+nC5+C6n(n-i) +C7n(n-1-i) +C8n(n-1-i) +C9n(n-1-i) +C10n(n-1-i) +C11n(n-1-i)+nC12+(n+1)C13=

Найбільша степінь n2 іншими менш значимими значеннями можна знехтувати

= (n2-ni)( C6+C7 +C8 +C9 +C10 +C11)= (n2-ni)=n2. Складність буде O(n2).

Сортування вставлянням Θ(𝑛2 ) Θ(𝑛) Θ(𝑛2 ).

Асимптотична складність у бульбашки та вставляння однакова.

Сортування зливанням Θ(𝑛𝑙𝑜𝑔 𝑛) Θ(𝑛𝑙𝑜𝑔 𝑛) Θ(𝑛𝑙𝑜𝑔 𝑛)

Бульбашка менш ефективна, бо її асимптотична складність n2 та n, в той час як у зливання nlogn, а логарифм зростає повільніше.

**2**. Оцінити асимптотичну складність алгоритму сортування зливанням, скориставшись основною теоремою рекурсії.

𝑇(𝑛) = 𝑎𝑇([ 𝑛𝑏 ]) + 𝑂(𝑛𝑑 )

Складність рекурсивної функції O(n), отже d=1. У алгоритмі зливання йде поділ послідовності на 2 підпослідовності і вони обидві розглядаються, отже a=b=2.

d=1, і logba=log22=1, тому logba=d. Отже Fn=O(ndlogn).

Часова складність: T(n)= Θ(nlogn).

**3**. Вивчити і записати (будь-яким способом) самостійно алгоритм швидкого сортування. Оцінити асимптотичну складність алгоритму швидкого сортування, скориставшись основною теоремою рекурсії.

У алгоритмі йде поділ на 2 підзадачі і обидві розглядаються тому a=b=2.

int Quick\_sort (int b[], int B, int E)

{ | T | n |

long i = B, j = E; | C1 | 1 |

int p = b[(B+E)/2]; | C2 | 1 |

do{ | C3 | 1 |

while (b[i] < p)i++; | C4 | k(n/2) |

while (b[j] > p)j--; | C5 | k(n/2) |

if (i<=j) | C6 | k |

{ |

swap(b[i], b[j]); | C7 | k |

i++; | C8 | k |

j--; | C9 | k |

} |

}while (i<=j); | C10 | k+1 |

if (B<j)return Quick\_sort(b,B,j); | C11 | 1 |

if (i<E) return Quick\_sort(b,i,E); | C12 | 1 |

}

У найкращому випадку складність рекурсивної функції, при k=1 (якщо масив вже відсортований), то після відкидань менш значущих частин отримаємо

T(n)=C4(n/2)+C5(n/2)=n/2

Ω(n)=n, Отже d=1=logba=log22.

Тому за теоремою рекурсії у найкращому випадку складність буде O(n)=nlogn.

У найгіршому випадку, при k=n/2, отримаємо складність

T(n)=C4(n2/4)+C5(n2/4)=n2/4

O(n)=n2, d=2, log22=1; d>logba.

Тоді складність = O(nd), отже у найгіршому випадку складність буде O(n)=n2.