“Метод швидкого сортування”

Завдання

Реалізувати наступні три модифікації алгоритму швидкого сортування (Quick Sort) та порівняти їх швидкодію. Швидкість алгоритмів порівнюється на основі підрахунку кількості порівнянь елементів масиву під час роботи алгоритмів.

Алгоритм 1. Звичайний алгоритм швидкого сортування

В якості опорного елементу масиву під час кожного розбиття використовується останній елемент з поточного підмасиву (див. лекцію).

```
Partition(A, p, r)
1  x ← A[r]
2  i ← p - 1
3  for j ← p to r-1
4     do if A[j] ≤ x
5           then i ← i + 1
6                Обміняти A[i] ↔ A[j]
7  Обміняти A[i+1] ↔ A[r]
8  return i + 1
```

В цьому алгоритмі є тільки одне місце, де відбувається порівняння елементів масиву — рядок 4 наведеного вище псевдокоду. Зверніть увагу, що при додаванні в код лічильника порівнянь, місце, де повинно відбуватись його збільшення (інкрементація), повинно розташовуватись перед умовою if, а не в середині тіла умовного оператора (тобто після then).

Алгоритм 2. Швидке сортування з 3-медіаною в якості опорного елемента

Ця модифікація швидкого сортування працює наступним чином. Перед початком кожного розбиття для поточного підмасиву (A[p..r]) обирається три елементи: перший елемент підмасиву, останній елемент підмасиву, та елемент за індексом (p+r)/2 (той що знаходиться посередині підмасиву). Серед цих обраних елементів в якості опорного для подальшого розбиття обирається медіана — середній з трьох обраних.

При підрахунку кількості порівнянь даного алгоритму необхідно враховувати наступне.

• Порівняння не підраховуються під час визначення медіани.

• Процедура розбиття викликається тільки для підмасивів розміром більше 3. Для підмасивів з розміром менше або рівним 3, відбувається сортування без процедури розбиття. Але в цьому випадку необхідно всеодно враховувати порівняння елементів і вести їм облік.

На додаткові 2 бали:

Алгоритм 3. Швидке сортування з трьома опорними елементами

В цій модифікації замість одного опорного елементу обирається три. Позначимо ці опорні елементи q_1, q_2, q_3 (необхідно, щоб виконувалось: $q_1 < q_2 < q_3$). Перед основною частиною процедури розбиття ці опорні елементи обираються серед наступних елементів підмасиву A[p..r]: A[p], A[p+1] та A[r]. По завершенню розбиття всі елементи підмасиву $A[p..q_1-1]$ будуть менші за q_1, всі елементи $A[q_1+1..q_2-1]$ — менші за q_2, всі елементи $A[q_2+1..q_3-1]$ — менші за q_3,та всі елементи $A[q_3+1..r]$ — більші за q_3. І алгоритм рекурсивно продовжує свою роботу для вказаних чотирьох частин масиву: $A[p..q_1-1]$, $A[q_1+1..q_2-1]$, $A[q_2+1..q_3-1]$, $A[q_3+1..r]$.

Детальна робота цього алгоритму та його псевдокод наведений в статті Multi-Pivot Quicksort: Theory and Experiments. S Kushagra, A L.pez-Ortiz, A Qiao, JI Munro - ALENEX, 2014 – SIAM (текст статті додається до завдання).

Аналогічно до алгоритму 2, при підрахунку кількості порівнянь даного алгоритму необхідно враховувати наступне.

• Порівняння не підраховуються під час визначення впорядкування трьох опорних елементів.

• Процедура розбиття викликається тільки для підмасивів розміром більше 3. Для підмасивів з розміром менше або рівним 3, відбувається сортування без процедури розбиття. Але в цьому випадку необхідно все-одно враховувати порівняння елементів і вести їм облік.
Запропоновані модифікації алгоритму швидкого сортування дозволяють значно зменшити кількість порівнянь: алгоритм №2 до 10-15% і алгоритм №3 до 20-25% порівняно з рандомізованим алгоритмом швидкого сортування.

Рекомендації до підрахунку порівнянь

При підрахунку порівнянь елементів вхідного масиву слід бути уважним до місць розміщення_операцій збільшення лічильника порівнянь:
• Коли порівняння відбувається в операторі if, то операцію збільшення лічильника слід розміщувати перед оператором, а не всередині тіла then, щоб також враховувати негативні результати порівняння.
• Коли порівняння відбувається в умові операторів while чи for, то операцію збільшення лічильника слід розміщувати всередині циклу, а також додавати одну операцію збільшення лічильника відразу після закінчення циклу щоб враховувати останню невдалу перевірку (завдяки якій виконання циклу закінчується; втім тут слід бути обережним із завчасним перериванням роботи циклу за допомогою команди break).
• Слід пам’ятати, що необхідно вести облік тільки порівнянням елементів вхідного масиву.
Тому коли відбувається порівняння індексів чи будь-яких інших допоміжних змінних, це не слід враховувати.
• Уважно слідкуйте за тим, які опорні елементи обираються (див. опис алгоритмів вище), бо вибір опорного елементу на пряму впливає на кількість порівнянь для кожного конкретного масиву.

Формат вхідних/вихідних даних

Розроблена програма повинна зчитувати вхідні дані з файлу заданого формату та записувати дані у файл заданого формату.
Вхідний файл представляє собою текстовий файл із N+1 рядків, де N — це розмірність вхідного масиву A. Першим записом є число — кількість елементів в масиві; наступні N записів містять елементи вхідного масиву.
Вихідний файл представляє текстовий файл з одним рядком. Вміст цього рядка наступний: X Y Z, де X — це кількість порівнянь під час роботи алгоритму №1 над заданим вхідним масивом, Y— кількість порівнянь алгоритму №2, та Z — кількість порівнянь алгоритму №3.
До документу завдання також додаються приклади вхідних і вихідних файлів різної розмірності.

Нижче наведені приклади вхідного та вихідного файлу для N = 10.

Вхідний файл	Вихідний файл
10	
7	
5	
4	
8	48 29 20
9	
6	
3	
10	
1	
2	

Вимоги до програмного забезпечення

• Програма повинна розміщуватись в окремому висхідному файлі, без використання додаткових нестандартних зовнішніх модулів.
• Не дозволяється використовувати будь-які нестандартні бібліотеки та розширення. Програма не повинна залежати від операційної системи.
• Не реалізуйте жодного інтерфейсу користувача (окрім командного рядку). Програма не повинна запитувати через пристрій вводу в користувача жодної додаткової інформації. Вашу програму будуть використовувати виключно у вигляді “чорного ящику”.
• Назва висхідного файлу вашої програми повинна задовольняти наступному формату: НомерГрупи_ПрізвищеСтудента_НомерЗавдання.Розширення, де НомерГрупи — це один з рядків is91, is92, is93; ПрізвищеСтудента — прізвище студента записане латинськими літерами; НомерЗавдання — двозначний номер завдання (01, 02, ...); Розширення — розширення файлу, відповідно до мови програмування (.py). Приклад назви висхідного файлу: is91_ivanenko_04.py.
• Розроблена програма повинна зчитувати з командного рядку назву вхідного файлу та записувати результат у вихідний файл. При запуску першим і єдиним аргументом командного рядку повинна бути назва вхідного файлу (наприклад, input_10.txt). Назва вихідного файлу повинна складатись із назви файлу самої програми разом із суфіксом “_output” і мати розширення .txt. Приклад назви вихідного файлу: is91_ivanenko_04_output.txt.
Увага! Дозволяється не реалізовувати алгоритм №3. В такому випадку не нараховуються додаткові 2 бали.