МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

Кафедра комп'ютерної інженерії та електроніки

ЗВІТ З ПРАКТИЧНИХ РОБІТ

з навчальної дисципліни

«Алгоритми та методи обчислень»

Тема «Асимптотична складність алгоритмів. Інші нотації»

Студент гр. КІ-23-1 ПІБ Кобець О. О.

Кременчук 2024

Практична робота № 2

Тема. Асимптотична складність алгоритмів. Інші нотації

Мета: набути практичних навичок у розв'язанні задач на оцінку асимптотичної складності алгоритмів у Ω, Θ, o, θ, ω-нотаціях.

№8

8. Маємо функції $f(n) = n\ 4 + 2n\ 3 - 5n\ 2 + 8$ та $g(n) = n\ 4$. Показати, що $f(n) = O(g(n))$, використовуючи метод меж.

$f(n) = (n)^4 + 2n^3 - 5n^2 + 8$ та $g(n) = n^4$

f(n)=O(g(n))

$$\frac{lim}{n \to \infty} \frac{f(n)}{g(n)} = \frac{n4+2n3-5n2+8}{n^4}$$

$f(n)^1 = (4n^3+6n^2-10n)^1 = (12n^2+12n-10)^1 = (24n+12)^1 = 24$

$g(n^4)^1 = (4n^3)^1 = 12n^2 = 24$

$$\lim \frac{f(n)}{g(n)} = \frac{24}{24} = 1$$

f(n)=O(g(n))

№13

13. Задано функції $f(n) = n\ 3 - n\ 2 + 2$ і $g(n) = n\ 4$. Показати, що $f(n) = O(g(n))$, використовуючи метод меж.

$f(n) = n\ 3 - n\ 2 + 2$ і $g(n) = n\ 4$

f(n)=O(g(n))

$$\frac{lim}{n \to \infty} \frac{f(n)}{g(n)} = \frac{n3 - n2 + 2}{n^4} = \frac{\infty}{\infty}$$

$f(n)^1 = (3n^2 - 2n)^1 = (6n - 2)^1 = 6$

$g(n^4)^1 = (4n^3)^1 = (12n^2)^1 = 24n$

$$\frac{lim}{n \to \infty} \frac{f(n)}{g(n)} = \frac{6}{24n} = \frac{6}{\infty} = 0$$

f(n)=O(g(n))

Контрольні запитання

1. Що таке асимптотична складність алгоритму?

Асимптотична складність алгоритму — це характеристика ефективності алгоритму в умовах великих розмірів вхідних даних. Вона визначає, як швидко змінюється час виконання або кількість операцій алгоритму в залежності від розміру вхідних даних. Зазвичай асимптотична складність виражається через різні нотації, зокрема **O-нотація**, **Θ-нотація**, **Ω-нотація**.

2. Які інші нотації, крім O-нотації, використовуються для вираження

асимптотичної складності?

Крім O-нотації, для вираження асимптотичної складності також використовуються:

- Θ-нотація (Тета-нотація): описує точну (вхідну та вихідну) складність алгоритму.
- Ω-нотація (Омега-нотація): використовується для позначення найменшої складності алгоритму в найгіршому випадку.
- o-нотація (маленьке o): описує верхню межу складності, яка є строго більшою за певну функцію.
- ω-нотація (маленьке омега): описує нижню межу складності, яка є строго меншою за певну функцію.

3. Як визначити асимптотичну складність алгоритму за допомогою

символів Θ і Ω?

- Θ-нотація визначає точну складність алгоритму: якщо функція складності алгоритму є $\Theta(f(n))$, це означає, що алгоритм має складність, яка обмежена з обох сторін функцією $f(n)$ при великих значеннях n. Тобто, $\Theta(f(n))$ означає, що складність алгоритму зростає точно як $f(n)$.
- Ω-нотація визначає найгірший випадок для алгоритму, описуючи мінімальну складність: якщо функція складності алгоритму є $\Omega(f(n))$, це означає, що складність алгоритму не може бути меншою за $f(n)$ при достатньо великих значеннях n.

4. Яка різниця між O-нотацією, Θ-нотацією і Ω-нотацією?

- O-нотація (Велике O): визначає верхню межу складності алгоритму, тобто максимально можливу складність. Вона говорить, як швидко зростає складність алгоритму у найгіршому випадку.

- Θ-нотація: визначає точну складність алгоритму, тобто описує як верхню, так і нижню межу складності алгоритму.
- Ω-нотація (Омега-нотація): визначає нижню межу складності, тобто мінімальну складність алгоритму, яка гарантується навіть у найкращому випадку.

5. Які основні властивості інших нотацій, таких як o (маленька o), ω

(маленька омега) та *o* (маленька о з верхнім індексом)?

- o-нотація (маленьке o): вказує, що функція росте строго повільніше за певну іншу функцію при великих значеннях n. Тобто, f(n) = o(g(n)), якщо для будь-якого множника функція f(n) зростає менше, ніж g(n) при великих значеннях n.
- ω-нотація (маленьке омега): вказує, що функція росте строго швидше за певну іншу функцію при великих значеннях n. Тобто, f(n) = ω(g(n)), якщо функція f(n) зростає швидше за функцію g(n) при великих значеннях n.
- *o*-нотація з верхнім індексом: не є поширеною в стандартних аналізах алгоритмів і зазвичай не застосовується в класичній теорії складності. Вона може використовуватися в специфічних контекстах або як варіант позначення для певних типів асимптотичних функцій.