*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*

*Гур’єв Д.П*

*Кременчук 2023*

*Практична робота № 1*

*Тема. Асимптотична складність алгоритмів. 𝐎-нотація*

*Мета: набути практичних навичок у розв’язанні задач на оцінку асимптотичної складності алгоритмів у 𝑂.*

*№8* Довести, що 𝑓(𝑛) = 2𝑛^2 + 10𝑛 + 3 = 𝑂(𝑛^2 ).

**Запис функцій:**

f*(n)=2n^2+10n+3*

*g(n)=n^2*

**Знаходження границі:**

Розглянемо, як зростає кожна частина функцій при n→∞: n^2≤2*n*2+10*n*+3

**Вибір константи** *C***:**

Виберемо *C*=15, оскільки 2*n^*2+10*n*+3≤15*n*2 при *n*≥1.

**Вибір початкового значення n0** **:**

Оберемо *n*0 =1, оскільки ми вже довели нерівність для *n*≥1.

**Доведення:**

Тепер ми можемо перевірити, що для всіх *n*≥*n*0 виконується *f*(*n*)≤*C*⋅*g*(*n*): 2*n^*2+10*n*+3≤15*n*2 для *n*≥1.

Отже, ми обрали константи *C*=15 та *n*0 =1, для яких виконується *f*(*n*)≤*C*⋅*g*(*n*), що доводить, що *f*(*n*)=*O*(*g*(*n*)).

№13 Задано функції 𝑓(𝑛) = 100𝑛^3 + 8 та 𝑔(𝑛) = 𝑛^3 . Доведіть, що 𝑓(𝑛) = 𝑂(𝑔(𝑛)).

**Запис функцій:**

*f*(*n*)=100*n*3+8

*3g*(*n*)=*n*3

**Знаходження границі:** Розглянемо, як зростає *f*(*n*) порівняно з *g*(*n*) при *n*→∞: *100n^3+8≤100n^3+8⋅n^3=108n^3*

**Вибір константи** *C***:** Оберемо *C*=108, оскільки *100n^3+8≤108n^3*100*n*3+8≤108*n*3 для всіх *n*≥1.

**Вибір початкового значення** *n*0 **:** Виберемо *n*0 =1, оскільки ми вже довели нерівність для *n*≥1.

**Доведення:** Тепер ми можемо перевірити, що для всіх *n*≥*n*0 виконується *f*(*n*)≤*C*⋅*g*(*n*): 100*n^*3+8≤108*n^*3 для *n*≥1.

Отже, ми обрали константи *C*=108 та *n*0 =1, для яких виконується *f*(*n*)≤*C*⋅*g*(*n*), що доводить, що *f*(*n*)=*O*(*g*(*n*)).

Контрольні Запитання

1.Що таке асимптотична складність алгоритму?

2. Яким чином визначається 𝑂-нотація і яка її сутність?

3. Які основні правила використання 𝑂-нотації при аналізі алгоритмів?

4. Що означають вирази 𝑂(1), 𝑂(𝑛), 𝑂(𝑛 2 ) в контексті асимптотичної складності?

5. Яким чином визначити асимптотичну складність алгоритму за його кодом або математичним виразом?

1.Асимптотична складність алгоритму - це спосіб визначення того, як швидко зростає час виконання або використання ресурсів (наприклад, пам'яті) алгоритму зі збільшенням розміру вхідних даних. Це дозволяє оцінити ефективність алгоритму при роботі з великими обсягами даних.

2.𝑂-нотація - це математичний спосіб вираження асимптотичної складності алгоритму. Сутність полягає в тому, що алгоритм оцінюється за його "найгіршим" часом або використанням ресурсів для найгіршого можливого випадку.

3.Основні правила використання 𝑂-нотації при аналізі алгоритмів:

Враховуйте тільки найбільш значущий член виразу, що описує складність.

Ігноруйте коефіцієнти перед членами виразу.

Ігноруйте нижчеоптимальні члени виразу (наприклад, у виразі 𝑂(𝑛^2 + 𝑛), ігноруються менш впливові члени, такі як 𝑛).

4.У контексті асимптотичної складності:

𝑂(1) - означає постійну складність, коли час або ресурси не змінюються з розміром вхідних даних.

𝑂(𝑛) - означає лінійну складність, коли час або ресурси зростають лінійно з розміром вхідних даних.

𝑂(𝑛^2) - означає квадратичну складність, коли час або ресурси зростають квадратично з розміром вхідних даних.

5.Асимптотичну складність можна визначити за кодом алгоритму, розглядаючи, як змінюється кількість операцій або розмір пам'яті при збільшенні розміру вхідних даних. Також можна використовувати математичні методи, такі як аналіз алгоритму за допомогою рекурентних виразів або асимптотичного аналізу математичних виразів, що описують роботу алгоритму.