Теорія алгоритмів. Вступ до дисципліни. Поняття алгоритму. Основні властивості алгоритмів

Лекція 1

План лекції

- Історія поняття «алгоритм»
- Визначення поняття «алгоритм»
- Властивості алгоритмів
- Форми представлення алгоритмів
- Специфікація алгоритму

Історія поняття «алгоритм»

Теорія алгоритмів як окремий розділ математики, що вивчає загальні властивості алгоритмів, виникла в **30-х роках XX століття**.

Поняття алгоритму відноситься до первинних понять математики і служить концептуальною основою процесів обробки інформації. Загальні властивості алгоритмів вивчає розділ математики, яка називається ***теорією алгоритмів***.

Сучасне формальне визначення алгоритму було дано в **30-50-і роки XX ст.** в роботах Тьюринга, Поста, Чорча, Вінера, Маркова.

Слово «алгоритм» походить від імені великого середньоазіатського вченого **Мухаммеда аль-Хорезмі**, який жив у першій половині IX століття, який вперше дав опис придуманої в Індії позиційної десяткової системи числення.

Визначення поняття «алгоритм»

- ***Визначення 1.*** (А.А. Колмогоров): Алгоритм – це будь-яка система обчислень, виконуваних за суворо визначеними правилами, яка після якого-небудь числа кроків свідомо призводить до вирішення поставленого завдання.
- ***Визначення 2.*** (А.А. Марков). Алгоритм – це точне розпорядження, що визначає обчислювальний процес, що йде від варійованих вихідних даних до шуканого результату.
- ***Визначення 3.*** (Д. Е. Кнут). Алгоритм – це кінцевий набір правил, який визначає послідовність операцій для вирішення конкретної множини завдань і володіє п'ятьма важливими рисами: скінченність, визначеність, введення, вивід, ефективність.
- ***Визначення 4.*** (Філософський словник) Алгоритм – точне розпорядження про виконання в певному порядку деякої системи операцій, що ведуть до розв'язання всіх задач даного типу.

Визначення поняття «алгоритм»

- ***Під алгоритмом*** розуміють скінчену множену точно визначених правил для чисто механічного вирішення завдань певного класу. Така множина правил задає обчислювальний процес, що називається алгоритмічним, який починається з довільних початкових даних (вибраних з деякої фіксованої для даного алгоритму множини початкових даних) і спрямований на отримання повністю визначеного цими початковими даними результату.

Застосовний алгоритм

Множина *вхідних даних*

Множина *вихідних даних*

Область застосування

Алгоритмічно обчислювана функція

Властивості алгоритмів

- **скінченність дій** (фінітність) – алгоритм повинен виконувати кінцеву кількість кроків при вирішенні задачі;
- **скінченність запису** – алгоритм повинен містити кінцеву кількість елементарно здійсненних приписів;
- **масовість** – початкові дані для алгоритму можна вибирати з певної (можливо, нескінченої) множини даних; це означає, що алгоритм призначений не для однієї конкретної задачі, а для класу однотипних завдань;
- **дискретність** – розчленованість процесу виконання алгоритму на окремі кроки;
- **елементарність** – кожен крок алгоритму може бути простим, елементарним, можливість виконання якого людиною або машиною не викличе сумнівів;

Властивості алгоритмів

- **детермінованість** – однозначність процесу виконання алгоритму; це означає, що набір об’єктів, одержуваних у якийсь момент, однозначно визначається набором об'єктів, отриманих в попередні моменти.
- **результативність** (**спрямованість**) – алгоритм має засоби, які дозволяють відбирати із даних, отриманих на певному кроці виконання, результативні дані, після чого алгоритм зупиниться, завершається алгоритму визначеними результатами;
- **правильність** – алгоритм повинен призводити до правильного по відношенню до поставленого завдання рішенню.

Специфікація алгоритму

- **Постановка задачі**
  - Призначення алгоритму
  - Умови тощо
- **Дані**
  - Початкові
  - Проміжні
  - Вихідні
- **Вхідна / вихідна форма**
- **Обмеження**
- **Характеристики алгоритму.**
- **Метод**
- **Тестові приклади**

- **Початкові дані:**
  - Перелік початкових даних;
  - Типи та формати даних;
  - Обмеження;
  - Спосіб введення.
- **Вихідні дані:**
  - Перелік вихідних даних;
  - Типи та формати даних;
  - Обмеження;

Таблиця даних / тестування

Таблиця даних

Клас	Ім'я	Зміст	Тип	Структура	Діапазон	Формат

- **Клас даних**
  - Початкові
  - Проміжні
  - Вихідні
- **Структура**
  - Проста змінна
  - Масив
  - Запис
- **Тип визначає**
  - Значення
  - Розмір
  - Допустимі операції

Таблиця тестування

№ тесту	Призначення тесту	Вхідні дані	Очікуваний результат	Фактичний результат	Висновок

Форми представлення алгоритмів

- **Словесний** (запис природною мовою)

  опис послідовних етапів обробки даних природною мовою

  - строго не формалізується;
  - страждає багатослівністю записів;
  - допускає неоднозначність тлумачення окремих приписів.

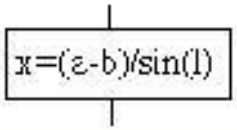
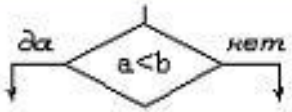
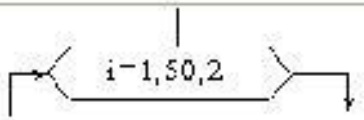
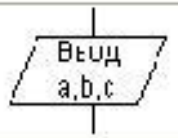
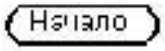
- **Графічний** (зображення з графічних символів, блок-схема)

  алгоритм зображується у вигляді послідовності пов'язаних між собою функціональних блоків, кожен з яких відповідає виконанню однієї або декількох дій.

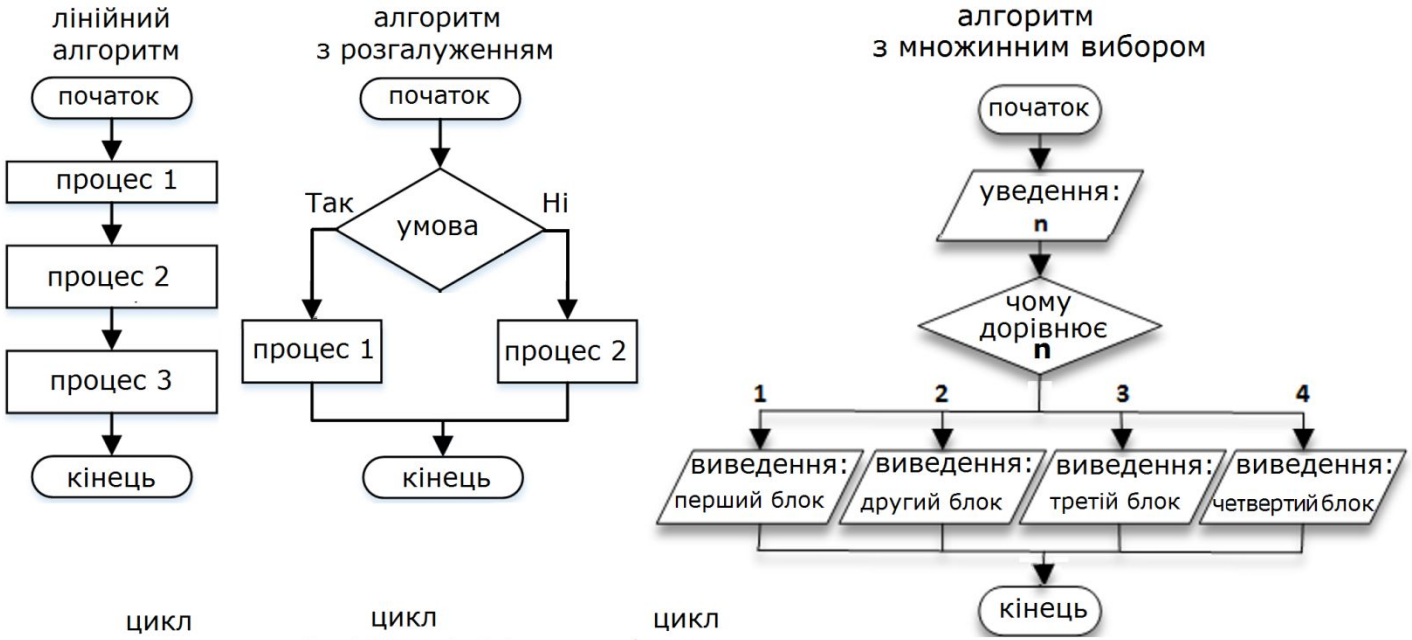
- **Псевдокод** (напівформалізований опис алгоритмів умовною алгоритмічною мовою, що включає елементи мови програмування, фрази природної мови, загальноприйняті математичні позначення)

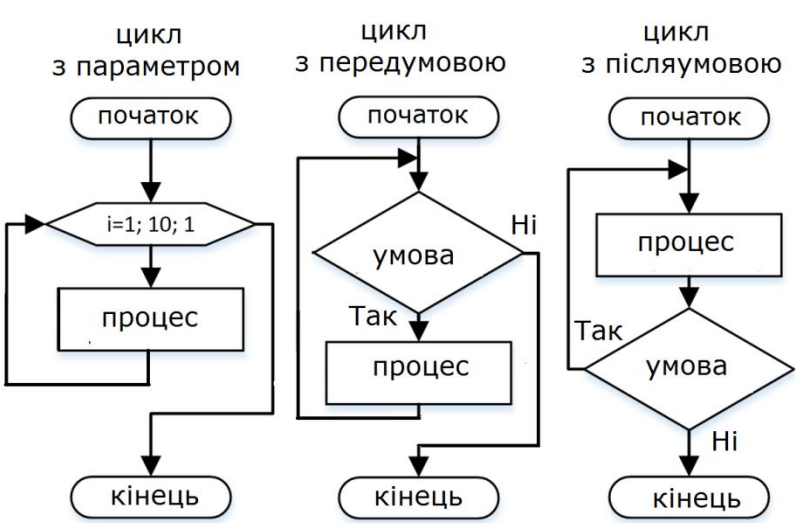
- **Код мовою на програмування**

Блок-схема

Назва символу	Позначення	Пояснення
Процес		Обчислювальна дія або послідовність дій
Умова		Перевірка умов
Модифікація		Початок циклу
Зумовлений процес		Обчислення в підпрограмі
Уведення / вивід		Уведення – вивід даних
Початок / кінець		Початок-кінець алгоритму

Блок-схема


лінійний алгоритм
початок
процес 1
процес 2
процес 3
кінець
алгоритм з розгалуженням
початок
умова
Так
Ні
процес 1
процес 2
кінець
алгоритм з множинним вибором
початок
уведення: n
чому дорівнює n
1
2
3
4
виведення: перший блок
виведення: другий блок
виведення: третій блок
виведення: четвертий блок
кінець


цикл з параметром
початок
i=1; 10; 1
процес
кінець
цикл з передумовою
початок
умова
Ні
Так
процес
кінець
цикл з післяумовою
початок
процес
Так
умова
Ні
кінець

Псевдокод

Псевдокод – компактна, напівформальна мова опису алгоритмів, що використовує **ключові слова** імперативних мов програмування, але опускає несуттєві для розуміння алгоритму подробиці і специфічний синтаксис.

Призначений для подання алгоритму людині, а не для комп'ютерної трансляції та подальшого виконання програми.

Головна мета – забезпечити розуміння алгоритму людиною, зробити опис більш прийнятним, ніж мовою програмування.

https://techrocks.ru/2020/03/27/how-to-write-pseudocode/

Псевдокод

```
begin
  поместить A_1, A_2, ..., A_n в ОЧЕРЕДЬ;
  for j ← k step −1 until 1 do
    begin
      for l ← 0 until m − 1 do сделать Q[l] пустым;
      while список ОЧЕРЕДЬ не пуст do
        begin
          пусть A_i — первый элемент в списке ОЧЕРЕДЬ;
          переместить A_i из списка ОЧЕРЕДЬ в черпак Q[a_ij]
        end;
      for l ← 0 until m − 1 do
          присоединить содержимое Q[l] к концу списка ОЧЕ-
            РЕДЬ
    end
end
```

Псевдокод

```
RecursiveChange(money, coins)
if money = 0:
  return 0
MinNumCoins ← ∞
for i from 1 to |coins|:
  if money ≥ coin_i:
    NumCoins ← RecursiveChange(money − coin_i, coins)
    if NumCoins + 1 < MinNumCoins:
      MinNumCoins ← NumCoins + 1
return MinNumCoins
```

```
алг Сумма квадратов (арг цел n, рез цел S)
дано | n > 0
надо | S = 1*1 + 2*2 + 3*3 + ... + n*n
нач цел i
 ввод n; S:=0
   Нц
        для i от 1 до n
       S:=S+i*i
    кц
     вывод "S = ", S
    кон
```

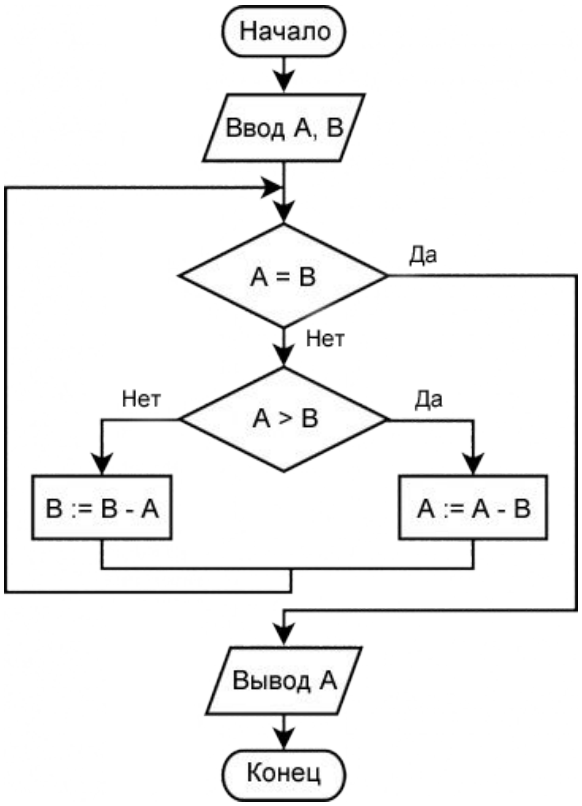
Псевдокод

```
Открыть файл "Продукция"
ЕСЛИ успешно
ТО
    Сбросить счетчик элементов
    ЦИКЛ-ПОКА до конца файла
        Читать очередную запись
        Увеличить счетчик на 1
    ВСЁ-ЦИКЛ
    Закрыть файл "Продукция"
ИНАЧЕ
    Сообщение об ошибке
ВСЁ-ЕСЛИ
```

```
FILE* pf = fopen("product.dat", "rb");
if(pf)
{
   prod_count = 0;
   while(!feof(pf))
   {
      fread(&prod[prod_count],
            sizeof(PROD), 1, pf);
      prod_count++;
   }
   fclose(pf);
}
else
   ShowMessage("Файл не найден");
```

Алгоритм Евкліда

```
функція нсд(a, b)
    якщо a = 0
        поверни b
    поки b ≠ 0
        якщо a > b
            a := a - b
        інакше
            b := b - a
    поверни a
```



Алгоритм Евкліда працює так: на кожному кроці від пари чисел a> b ми переходимо до пари a - b і b, тобто від більшого числа віднімаємо менше.