



Тема 1. Основи алгоритмізації та програмування

Будова комп'ютерних систем

Перші комп'ютери (ЕОМ) були переважно аналоговими, тобто, опрацьовували неперервні сигнали. Вони могли виконувати лише *обмежений набір завдань*. Перейти до виконання інших завдань можна було лише *змінивши будову таких комп'ютерів*.

Джон фон Нейман запропонував— комп'ютери мають бути *універсальними* щоб *виконувати різні завдання*.

Принципи Джона фон Неймана

1) використання двійкової системи числення для кодування інформації;

Один двійковий розряд може містити значення 0 або 1 — *біт*

В одному біті можна записати $2^1 = 2$ значення (0 та 1).

В двох бітах можна записати $2^2 = 4$ значення (00, 01, 10, 11).

В трьох бітах можна записати $2^3 = 8$ значень (000, 001, 010, 011, 100, 101, 110, 111).

В вісьмох бітах можна записати $2^8 = 256$ значень.

Сукупність 8 бітів — *байт*

1 кб = 1 кілобайт = 1024 байти = 2^{10} байт

1 Мб = 1 мегабайт = 1024 кілобайти = 2^{20} байт

1 Гб = 1 гігабайт = 1024 мегабайти = 2^{30} байт

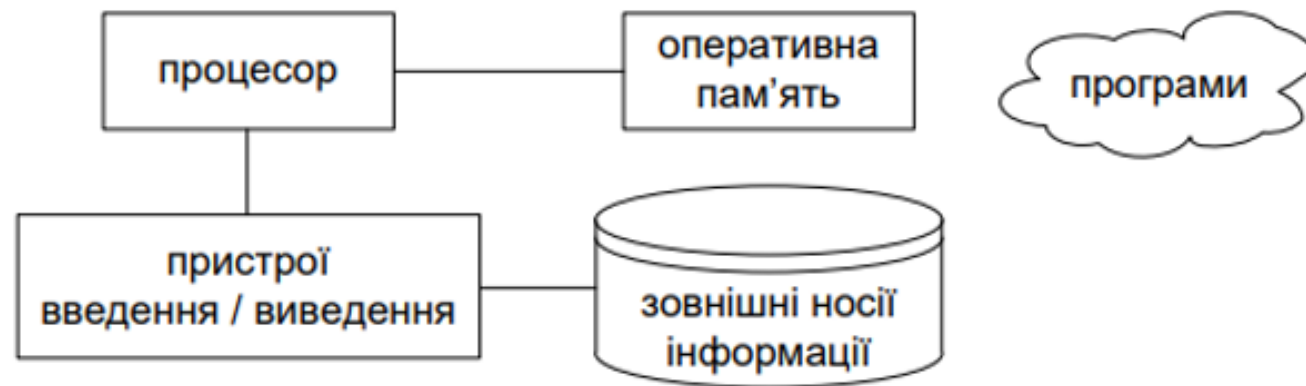
1 Тб = 1 терабайт = 1024 гігабайти = 2^{40} байт

1 Пб = 1 петабайт = 1024 терабайти = 2^{50} байт

2) програмне керування роботою комп'ютера;

Комп'ютер – сукупність технічних та програмних засобів для автоматизованого опрацювання дискретних даних відповідно до заданого алгоритму.

Має *фізичну* та *програмну* складові



Комп'ютер опрацьовує дискретні дані

3) зберігання програм у пам'яті комп'ютера;

Модель пам'яті

- *Принстонська* (університету м. Принстон): пам'ять однорідна – команди та дані зберігаються в одних і тих самих областях пам'яті. Це повільніше, але дешевше.

- *Гарвардська* (університету м. Гарвард): команди і дані використовують конструктивно різні види пам'яті. Це швидше, але дорожче

4) адресація пам'яті

Кожний байт має свою адресу. Найменша комірка пам'яті, яка може мати свою адресу – байт.

1 кб = 1 кілобайт = 1024 байти = 2^{10} байт

1 Мб = 1 мегабайт = 1024 кілобайти = 2^{20} байт

1 Гб = 1 гігабайт = 1024 мегабайти = 2^{30} байт

1 Тб = 1 терабайт = 1024 гігабайти = 2^{40} байт

1 Пб = 1 петабайт = 1024 терабайти = 2^{50} байт

архітектура комп'ютера – інтерфейс між його апаратним та програмним забезпеченням

Залежно від того, який тип пам'яті адресується, розрізняють наступні типи архітектур комп'ютера: стекова, акумуляторна, на основі регістрів загального користування.

Архітектура комп'ютера має визначальний вплив на його споживчі характеристики: коло вирішуваних задач, продуктивність, ємність основної пам'яті, ємність зовнішньої пам'яті, вартість, організація технічного обслуговування, надійність тощо.



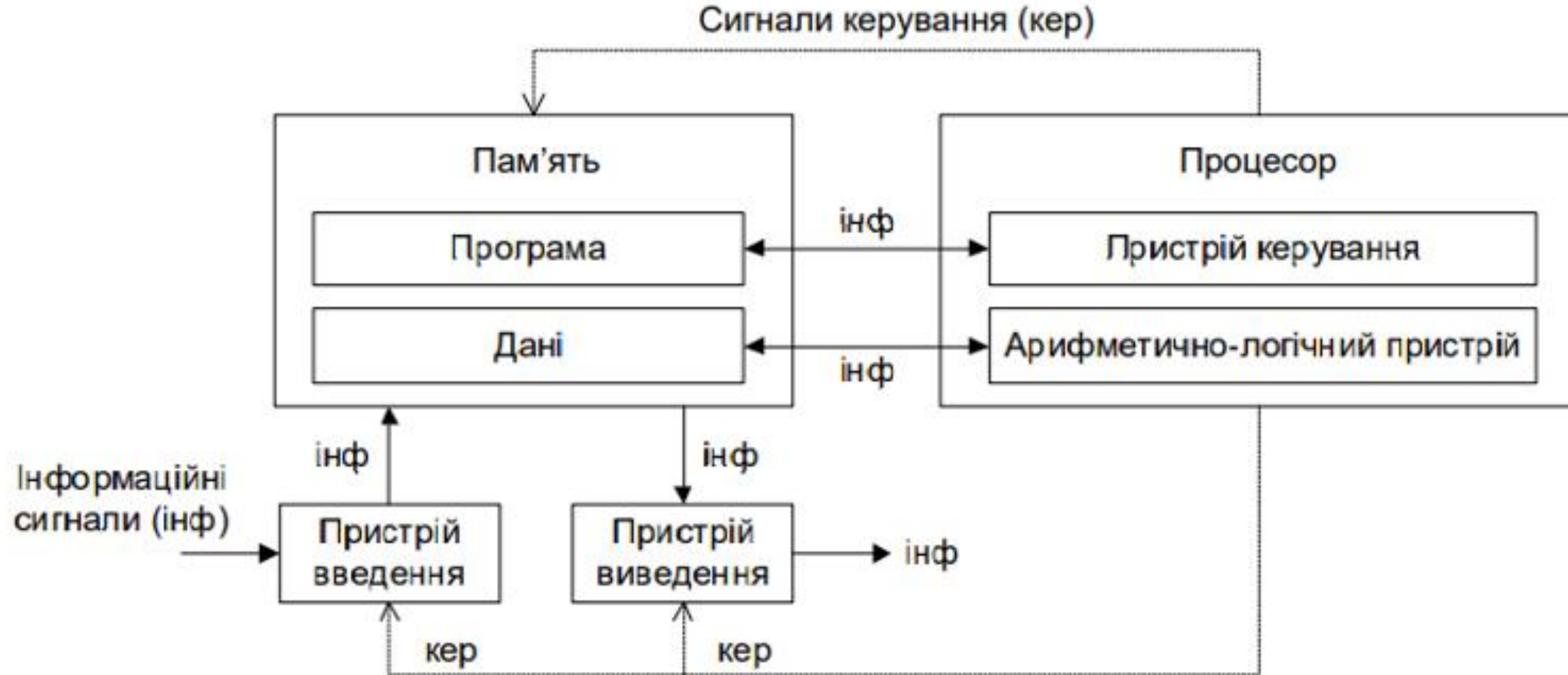
Рівні процесу опрацювання даних в сучасному комп'ютері

Архітектура фон Неймана

Архітектура фон Неймана – архітектура електронних обчислювальних машин (ЕОМ), головною характеристикою якої є спільне зберігання даних та машинних команд в комірках однієї й тієї ж пам'яті, що робить неможливим розрізнити дані та команди за способом представлення або кодування.

Обчислювальна машина є машиною з архітектурою фон Неймана, якщо:

- 1. Програма та дані зберігаються в одній загальній пам'яті.*
- 2. Кожна комірка пам'яті машини ідентифікується унікальним номером, який називається адресою.*
- 3. Різні слова інформації (команди та дані) розрізняються за способом використання, але не за способом кодування чи представленням в пам'яті.*
- 4. Кожна програма виконується послідовно, починаючи з першої команди, якщо немає спеціальних вказівок.*



Архітектура комп'ютера згідно принципів Дж. фон Неймана

Машинні та високорівневі мови програмування

Послідовність команд, що описують вирішення певної задачі, називається
програмою.

Операнди – це величини (змінні, константи), значення яких використовуються в операціях опрацювання даних.

Мова програмування – це формалізована мова, призначена для опису алгоритмів розв'язування задач на обчислювальних машинах.

Інтерпретатор зчитує програму по одному рядку і одразу ж виконує машинні інструкції, що містяться у цьому рядку

компілятор зчитує програму повністю і лише після цього перетворює її в об'єктний модуль, який безпосередньо може виконати комп'ютер.

Машинні та високорівневі мови програмування

Основними парадигмами програмування є *декларативне та імперативне*.

Декларативне програмування описує, який результат необхідно отримати, замість опису послідовності дій для отримання цього результату.

функційне та логічне програмування.

LISP, JavaScript, Python, Scala', Haskell.

Імперативне програмування описує процес отримання результатів як послідовність інструкцій зміни стану програми.

**структурне, процедурне, модульне (об'єктне),
об'єктно-орієнтоване програмування**

Системи числення та бітові (розрядні) операції



Система числення – це спосіб подання довільного числа за допомогою алфавіту символів, які називають цифрами.

Розрізняють *позиційні* та *непозиційні* системи числення.

Якщо в послідовності цифр, які зображають число, враховується позиція цифри, то систему числення називають **позиційною**.

Наприклад, у числі 5839 остання цифра 9, що перебуває на нульовій позиції, відповідає кількості одиниць, 3 на першій позиції вказує на кількість десятків, 8 на другій позиції – це кількість сотень і цифра 5 на третій позиції – це кількість тисяч.

10-ва система числення

Цифри	1	3	7
№№ розрядів	2	1	0
Множники	10^2	10^1	10^0
Значення множників	100	10	1

$$1 \cdot 10^2 + 3 \cdot 10^1 + 7 \cdot 10^0 = 1 \cdot 100 + 3 \cdot 10 + 7 \cdot 1 = 137$$

2-ва система числення

Цифри	0	1	0	1	0	1	1	0
№№ розрядів	7	6	5	4	3	2	1	0
Множники	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Значення множників	128	64	32	16	8	4	2	1

$$\begin{aligned}
 &0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = \\
 &0 \cdot 128 + 1 \cdot 64 + 0 \cdot 32 + 1 \cdot 16 + 0 \cdot 8 + 1 \cdot 4 + 1 \cdot 2 + 0 \cdot 1 = \\
 &64 + 16 + 4 + 2 = 86
 \end{aligned}$$

16-ва система числення

0x2C

Цифри	2	C
№№ розрядів	1	0
Множники	16^1	16^0
Значення множників	16	1

$$2 \cdot 16^1 + C \cdot 16^0 = 2 \cdot 16 + 12 \cdot 1 = 32 + 12 = 44$$

16-ва цифра	значення (10-ве зображення)	2-ве зображення	16-ва цифра	значення (10-ве зображення)	2-ве зображення
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	A	10	1010
3	3	0011	B	11	1011
4	4	0100	C	12	1100
5	5	0101	D	13	1101
6	6	0110	E	14	1110
7	7	0111	F	15	1111

Переведення чисел із 10-вої системи числення до 2-вої

№ кроку	Число x	Результат ділення числа x на ціло на 2 $x / 2$	Остача від ділення числа x на ціло на 2 $x \% 2$
0	137	68	1
1	68	34	0
2	34	17	0
3	17	8	1
4	8	4	0
5	4	2	0
6	2	1	0
7	1	0	1



1000 1001

Переведення чисел із 2-вої системи числення до 10-вої

Цифра	1	0	0	1	0	0	0	1
Розряд	7	6	5	4	3	2	1	0
Множник	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Значення множника	128	64	32	16	8	4	2	1

$$1 \cdot 2^7 + 0 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 128 + 16 + 1 = 145$$

Переведення чисел із 10-вої системи числення до 16-вої

Розряд	2	1	0
Цифра	1	3	D

0x13D

№ кроку	Число x	Результат ділення числа x на ціло на 16 x / 16	Остача від ділення числа x на ціло на 16 x % 16
0	317	19	13 → D
1	19	1	3 → 3
2	1	0	1 → 1



0x13D

Алгоритм та способи його подання

Алгоритм – це система правил, яка сформульована мовою, зрозумілою виконавцеві алгоритму, визначає процес переходу від допустимих початкових даних до певного результату і володіє властивостями масовості, скінченності, визначеності, детермінованості.

алгоритм – це скінченна послідовність команд, які слід виконати над вхідними даними, щоб отримати результат.




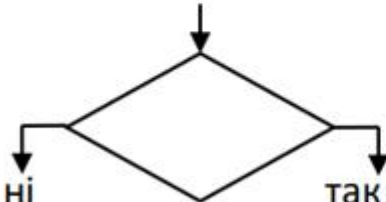

Процес побудови алгоритмів називають **алгоритмізацією**.


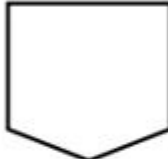
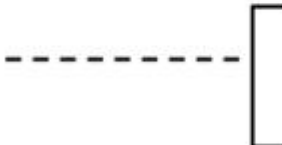
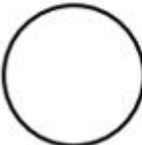
Для подання алгоритма використовують такі *способи*:

- *словесний опис* послідовності дій, який потребує подальшої формалізації;
- *аналітичний опис* у вигляді таблиць, формул, схем, малюнків
- *графічне подання* у вигляді схеми алгоритму (блок-схеми)
- *запис алгоритмічною мовою програмування*;

Схема алгоритму (блок-схема) – це графічне зображення його структури, в якому кожний етап процесу опрацювання даних подається у вигляді певних геометричних фігур (блоків).

Функціональні блоки

	Блок початку/завершення алгоритму
	Блок введення/виведення даних
	Арифметичний блок, модифікування (виконання операції або групи операцій, в результаті яких змінюються значення, форма подання або розміщення даних)
	Умовний блок, розгалуження (вибір напрямку галуження алгоритму залежно від результату аналізу умови)
	Блок виклику підпрограми (функції, процедури)

	Блок циклу з параметром
	Міжсторінковий з'єднувач (вказівка зв'язку між роз'єднаними частинами схем алгоритмів та програм, розташованих на різних аркушах)
	Коментар (зв'язок між елементами схеми і поясненнями)
	З'єднувач (вказівка зв'язку між перерваними лініями потоку в межах однієї сторінки)

Дякую за увагу

Лектор:

кандидат фіз.-мат. наук, доцент

Шаклеїна Ірина

iryna.o.shakleina@lpnu.ua

кафедра ICM, ІКНІ