# Основи програмування та алгоритмічні мови.

*Отыщи всему начало, и ты многое поймешь.*
*Козьма Прутков*

# Про курс

- Мета;
- Що будемо робити;
- Як будемо працювати;
- Контроль;

# Література

Розробка алгоритмів, програмування, система програмування.


- Прата С. Язык программирования C++. Лекции и упражнения. М.: ООО «И.Д. Вильямс», 2007.
- Дейтел Х., Дейтел П. Как программировать на C++.
- Вирт Н. Алгоритмы и структуры данных. - М.: Мир, 1989.
- Абрамов С.А., Зима Е.В. Начала информатики. - М. : Наука, 1989.
- Прохоренок Н.К. Программирование на C++ в Visual Studio 2010 Express. 2010.

# Знайомство

- П І Б.
- Де, коли, що закінчили?
- В яких позашкільних формах навчання приймали участь? Досягнення?
- Якими мовами програмування володієте? Ваша оцінка.
- Якими задачами з програмування займалися?
- Які розділи математики, задачі привертають Вашу увагу?
- Побажання.

# Алгоритм

Незважаючи на розвиток інформаційних технологій, технологій розробки програм, основним поняттям програмування є ***алгоритм***.

***Алгоритм*** – скінчена послідовність інструкцій виконавцю для розв`язання поставленої задачі.

***Властивості алгоритму***:

- Однозначність;
- Масовість;
- Детермінованість;
- Коректність;
- Скінченність;
- Ефективність.

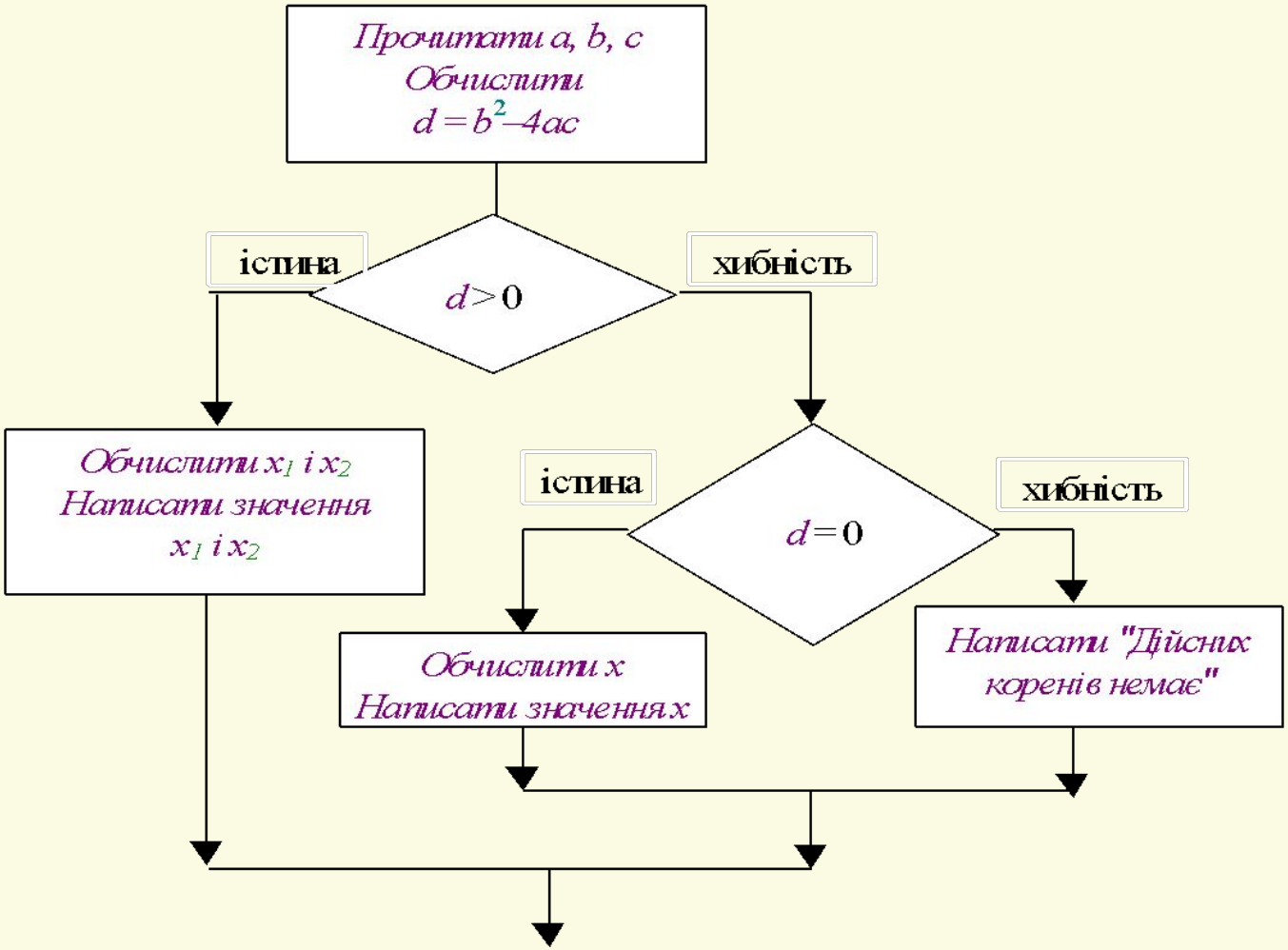
# Приклад

Обчислення дійсних коренів квадратного рівняння $ax^2 + bx + c = 0$, заданого коефіцієнтами a, b, c (при умові, що $a \neq 0$).

# Алгоритм

1. Прочитати коефіцієнти $a$, $b$, $c$.
2. Обчислити дискримінант
   $d = b^2 - 4ac$.
3. Якщо $d > 0$, обчислити
   $x_1=(-b-\sqrt{d})/2a$ , $x_2=(-b+\sqrt{d})/2a$
   та написати ці числа,
   інакше, якщо $d = 0$, обчислити
   $x =-b/2a$ і написати це число,
   інакше написати “дійсних коренів немає”.

# Алгоритм


Прочитати a, b, c
Обчислити
d = b^2 – 4ac
істина
d > 0
хибність
Обчислити x1 і x2
Написати значення
x1 і x2
істина
d = 0
хибність
Обчислити x
Написати значення x
Написати "Дійсних
коренів немає"

# Приклад

*Алгоритм Евкліда* для обчислення найбільшого спільного дільника двох натуральних чисел.

Позначимо найбільший спільний дільник чисел $a$ і $b$ через НСД($a$, $b$), а остачу від ділення $a$ на $b$ — через $a \bmod b$.

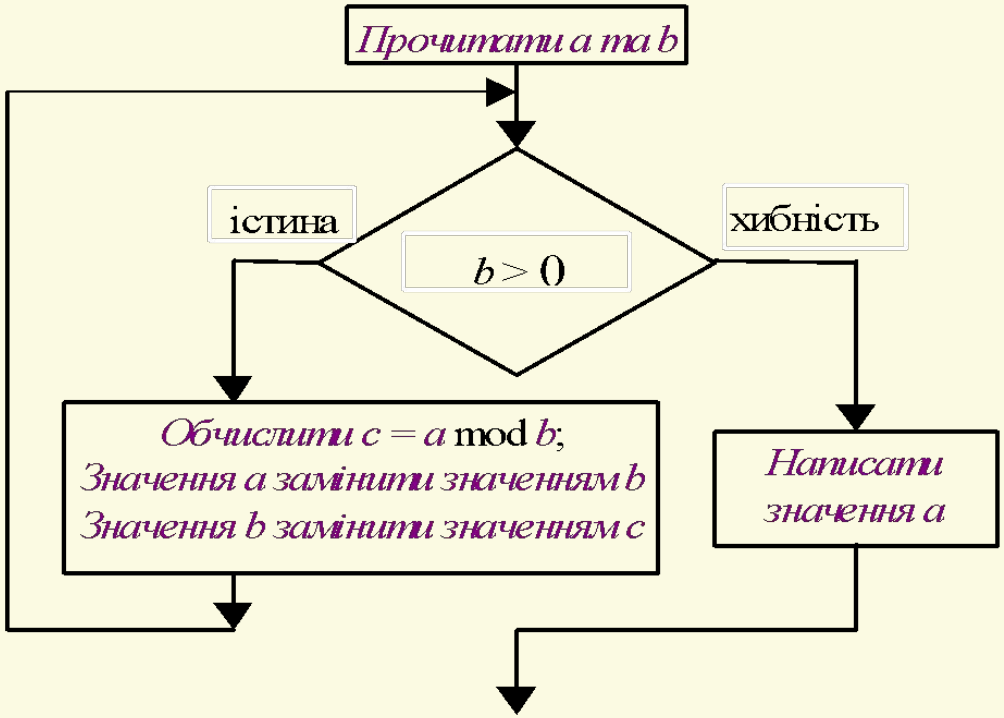
Алгоритм Евкліда оснований на тому, що НСД($a$, $b$) = НСД($b$, $a \bmod b$), якщо $b \neq 0$, і НСД($a$, $b$) = $a$, якщо $b = 0$.

Наприклад, НСД(12, 5) = НСД(5, 2) = НСД(2, 1) = НСД(1, 0) = 1.

# Алгоритм

1. Прочитати натуральні числа *a* і *b*.
2. Поки *b* > 0, виконувати такі дії:
   початок дій
   обчислити *c* = *a* mod *b*;
   значення *a* замінити значенням *b*;
   значення *b* замінити значенням *c*;
   кінець дій.
3. Написати значення *a*.

# Алгоритм


Прочитати a та b
істина
b > 0
хибність
Обчислити c = a mod b;
Значення a замінити значенням b
Значення b замінити значенням c
Написати значення a

# Задачі

- Для дійсних додатних чисел a, b, c з`ясувати, чи існує трикутник з вказаними довжинами сторін. Якщо трикутник існує, відповісти - чи є він гострокутним.
- Перевірити, чи мають три заданих цілих числа однакову парність.
- Обчислити дробову частину середнього геометричного трьох заданих додатних чисел.
- Дано дійсне число a. Знайти найменше n, що $1 + 1/2 + 1/3 + \ldots + 1/n > a$ .

# Комп’ютери та програми

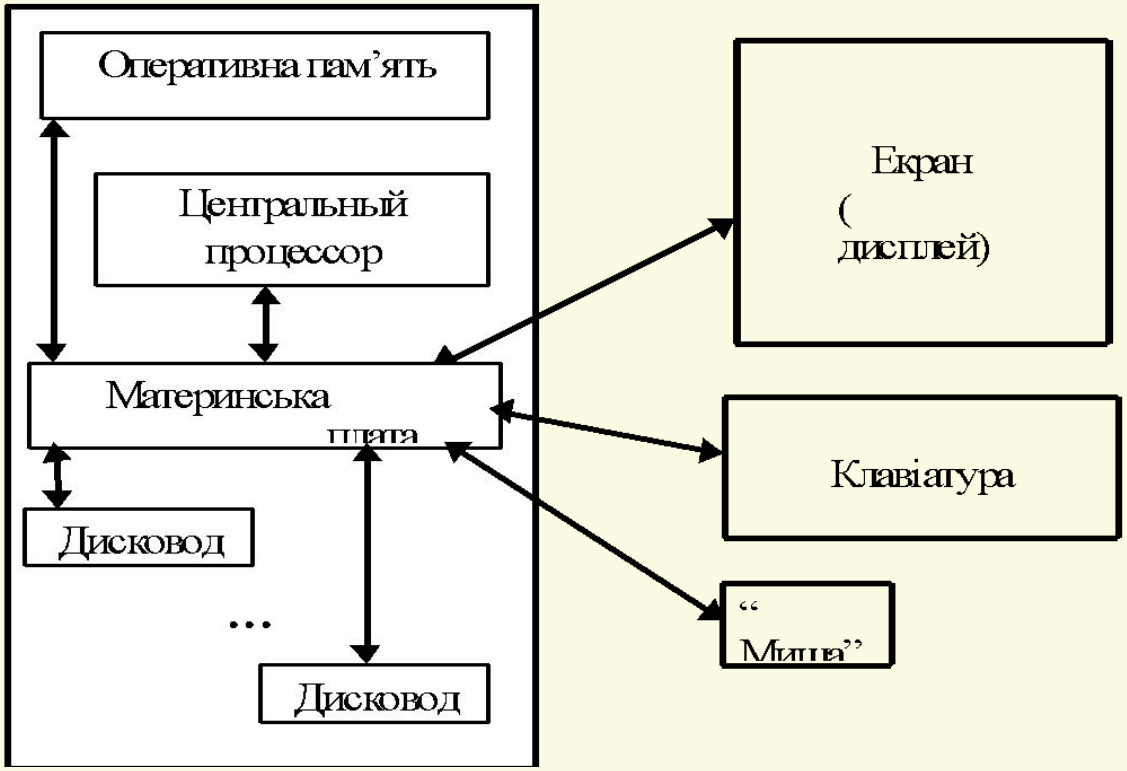
ПК = АЧ + ПЗ

АЧ - виконавець програм. Вимоги:

- обробка (обчислення);
- збереження (пам`ять);
- спілкування.

Визначають логічну структуру ПК з погляду програміста (*архітектура фон Неймана*).

# Структура комп’ютера



# Структура комп’ютера

***Основні частини процесора*** — це арифметико-логічний та керуючий пристрої, а також регістрова пам’ять.

Процесор має *кеш-пам’ять* (декілька рівнів).

*Основна* пам’ять: постійна (ROM), оперативна (RAM).

*Внутрішня* пам’ять: основна, регістрова, кеш-пам’ять.

*Зовнішня* пам’ять. Носії. Контролери. Порти.

# Дані та програми

Файли, формати (певні правила).

Системи числення з основами 10, 2.

*Бім* (*bit*, або *bi*nary digi*t* — двійкова цифра).

*Байт* - 8 послідовних бітів. Він може мати $2^8 = 256$ станів (ціле число від 0 до 255). Ці самі стани байта можуть розглядатися як числа від –128 до 127 (їх кількість теж 256), символи, логічні значення або щось інше.

Оперативна пам’ять - послідовність байтів, в якій кожний байт має свій номер — *адресу*.

# Дані та програми

Кбайт (“кілобайт”, $2^{10}$ = 1 024 байт),

Мбайт (“мегабайт”, $2^{20}$ = 1 048 576 байт)

Гбайт (“гігабайт”, $2^{30}$ = 1 073 741 824 байт).

- *Існує жарт: фізик вважає, що кілобайт це тисяча байтів, а програміст вважає, що кілометр це тисяча двадцять чотири метри.*

Регістри процесора можуть мати від одного до десяти байт. Обсяг кеш-пам’яті вимірюється десятками й сотнями Кбайт, а оперативної — сотнями Мбайт, кількома Гбайтами.

# Дані та програми

*Машинні команди*, як і дані, також записуються в RAM. Вони являють собою вказівки типу: “прочитати число за даною адресою пам’яті в даний регістр”, “додати два числа з даних регістрів і запам’ятати суму в даному регістрі”, …

Дії процесора (“прочитати”, “скласти” і т.п.) задаються в командах *кодами операцій*. Система команд, які можуть виконуватися процесором, називається *машинною мовою*.

Команда : КОП Адр1 Адр2 Адр3

# Приклад

Припустимо, що 8200, 8204 та 8248 — адреси оперативної пам’яті, де записані числа, 001 та 002 — номери регістрів, операція “*прочитати*” задається кодом 08, “*записати*” — 09, “*додати*” — 01, “*виконати команду*” — 22.

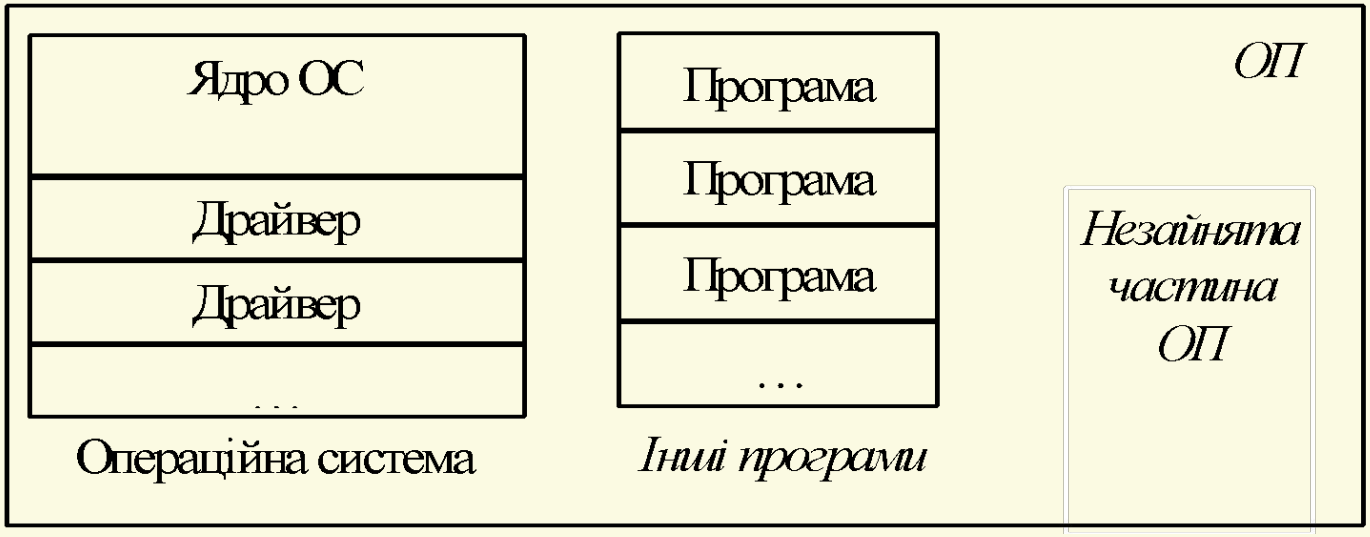
Щоб додати два числа, що записані за адресами 8200 і 8204, запам’ятати суму за адресою 8248 і після цього виконати команду, записану за адресою 15 376, процесор має виконати таку послідовність команд.

# Приклад

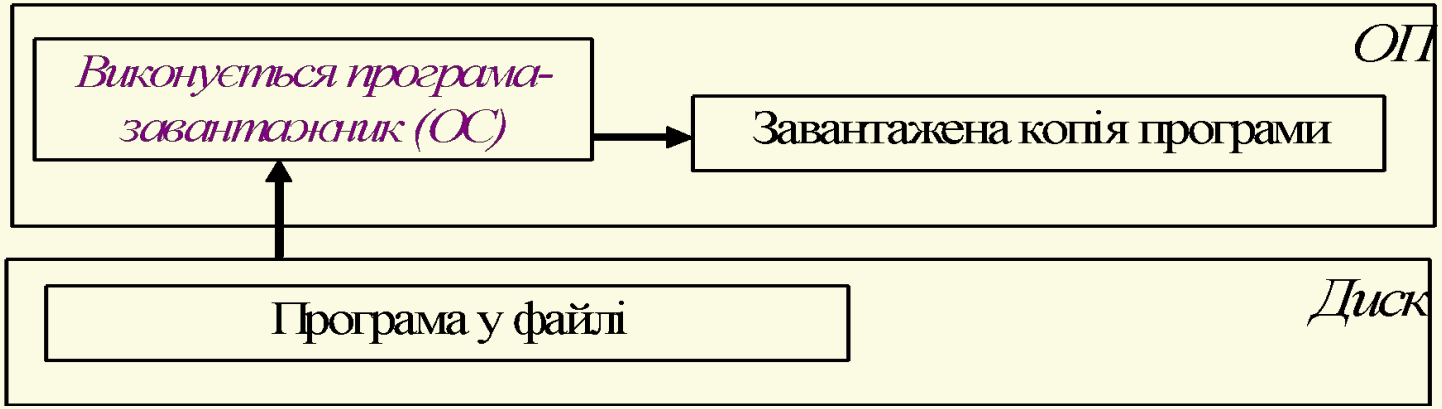
08 8200 001
    08 8204 002
    01 001 002 001
    09 001 8248
    22 15376

***Висновки*** - програми послідовності інструкцій з обробки даних. Програми записуються за певною системою, мовою зрозумілою комп’ ютеру.

# Програми у пам`яті



# Завантаження програм



# Засоби створення програм

***ІСТОРІЯ:***

- Пульт-комутатор;
- Машинні команди;
- Асемблери;
- Машино-незалежні мови високого рівня (FORTRAN - FORmula TRANslation );
  (~ 1 000 рядків-операторів)
- Структурне програмування (Pascal, C, ...);
  (~ 10 000 рядків-операторів)
  *універсальні мови програмування*
- ООП.

# Приклад

```
RD AX, A  ;прочитати число за адресою A
  до регістра AX
  RD BX, B  ;прочитати число за адресою
  B до регістра BX
  ADD AX, BX  ;додати числа з регістрів
  AX і BX, суму записати в AX
  WR C, AX  ;записати число з регістра
  AX за адресою C
  GOTO NEXT  ;перейти до виконання
  команди за адресою NEXT
```

# Приклад

```
Z=X+Y
  GO TO 315
```

# Етапи роз`язання задачі

- Постановка задачі;
- Специфікація задачі (формалізація, ТЗ);
- Математичні моделі та методи;
- Проектування програми;
- Кодування;
- Налагодження;
- Тестування;
- Документування;
- Впровадження.

*Ітеративний характер процесу.*

# Інструменти

- Технології проектування;
- Системи програмування.

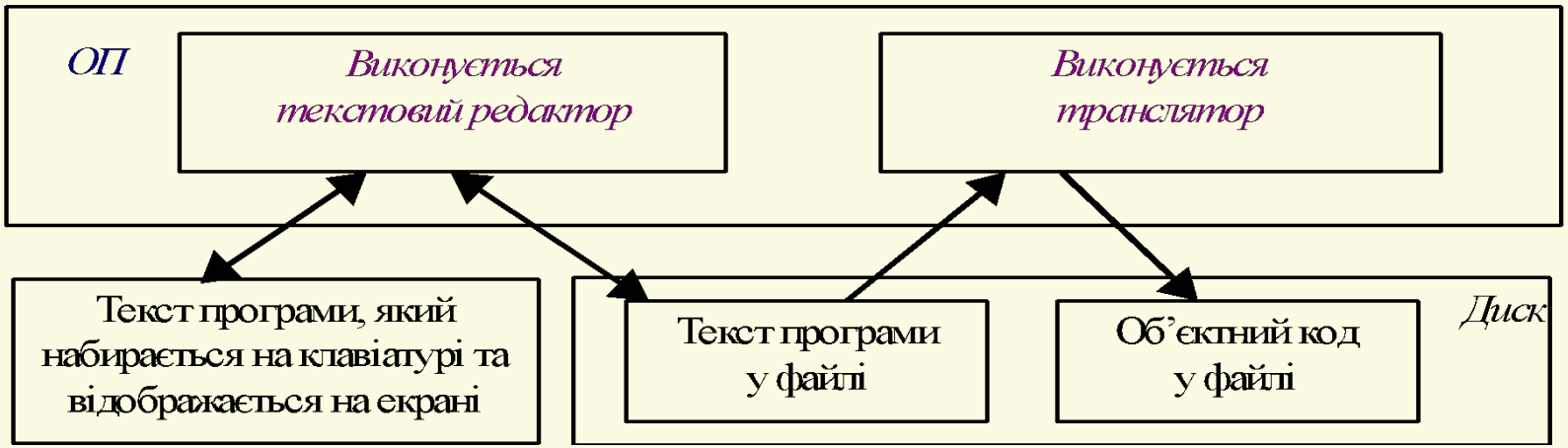
# Переклад та виконання програм

Необхідність перетворення (перекладу) програм з форми “мов високого рівня” у “машинні команди”.

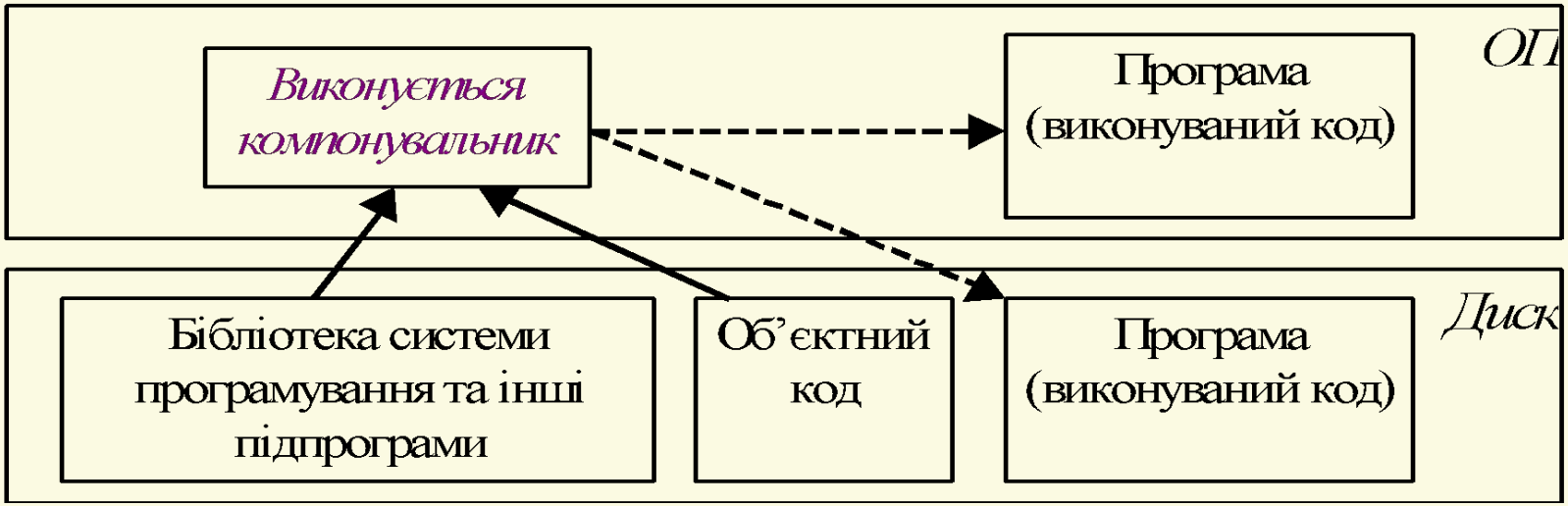
Програми - транслятори:

- Компілятори;
- Інтерпретатори;
- Поєднання підходів.

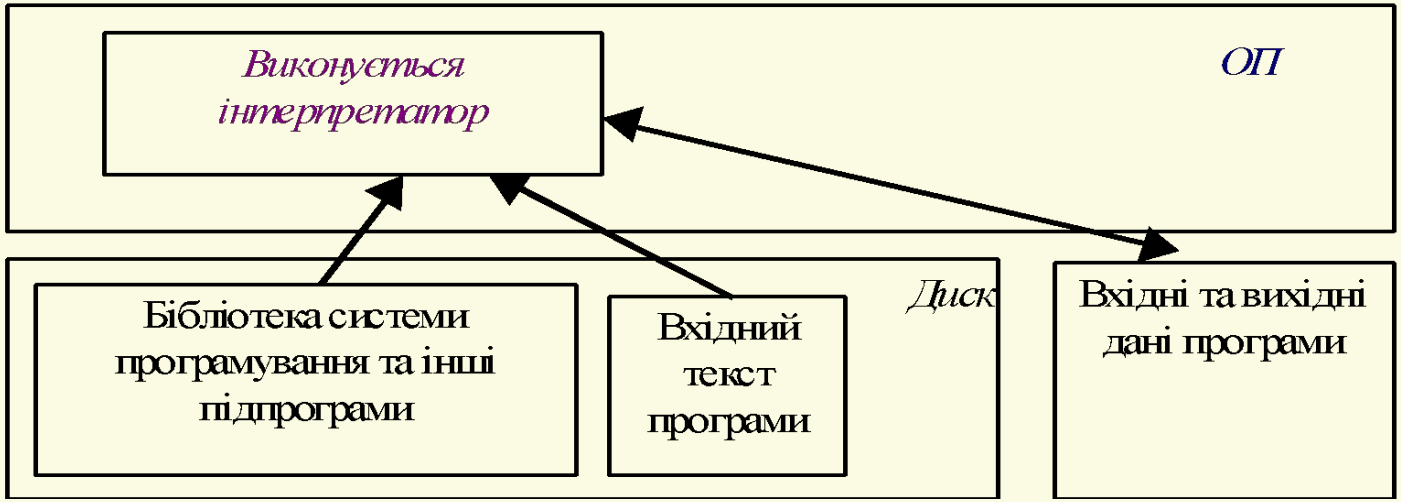
# Компілятор


ОП
Виконується текстовий редактор
Виконується транслятор
Текст програми, який набирається на клавіатурі та відображається на екрані
Текст програми у файлі
Об'єктний код у файлі
Диск

# Створення машинної програми



# Інтерпретатор



# Поєднання підходів

# Системи числення

Системи запису чисел:

- непозиційні (римська - I, V, X, L, C, M, ...);
- позиційні (10, 2, 8, 16, 3, ...), (наприклад -1357).

Запис $x_n x_{n-1} \ldots x_1 x_0, x_{-1} x_{-2} \ldots x_{-m}$ в системі з основою $P$ -

$$x_n P^n + x_{n-1} P^{n-1} + \ldots + x_1 P + x_0 + x_{-1} P^{-1} + x_{-2} P^{-2} + \ldots + x_{-m} P^{-m}$$

# Приклади

$512_{10} = 5\times10^2 + 1\times10^1 + 2\times10^0;$

$(512{,}346)_{10} = 5\times10^2 + 1\times10^1 + 2\times10^0 + 3\times10^{-1} + 4\times10^{-2} + 6\times10^{-3};$

$(10011)_2 = 1\times2^4 + 0\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0;$

$(10{,}011)_2 = 1\times2^1 + 0\times2^0 + 0\times2^{-1} + 1\times2^{-2} + 1\times2^{-3}$

$(1BC)_{16} = 1\times16^2 + 11\times16 + 12$

$(1B{,}C)_{16} = 1\times16^1 + 11\times16^0 + 12\times16^{-1}$

$(257)_8 = 2\times8^2 + 5\times8 + 7$

$(12{,}2)_3 = 1\times3^1 + 2\times3^0 + 2\times3^{-1} = 5{,}(6)_{10}$

# Запис числа у системі з основою P

???

# Системи з основами 2, 8, 16

- $2 \leftrightarrow 8 \quad 8 = 2^3 = (1000)_2$,

  0 — 000, 1 — 001, 2 — 010, 3 — 011,
  4 — 100, 5 — 101, 6 — 110, 7 — 111.

- $2 \leftrightarrow 16 \quad 16 = 2^4 = (10\,000)_2$,

  0 — 0000, 1 — 0001, 2 — 0010, 3 — 0011,
  4 — 0100, 5 — 0101, 6 — 0110, 7 — 0111,
  8 — 1000, 9 — 1001, *A* — 1010, *B* — 1011,
  *C* — 1100, *D* — 1101, *E* — 1110, *F* — 1111.

# Арифметичні операції в системах числення

$$\begin{array}{r} (1100) \\ +\ 11 \\ \hline 1001 \end{array}$$

$$\begin{array}{r} (1110) \\ +\ 777 \\ \hline 1062 \end{array}$$

$$\begin{array}{r} (1010) \\ +\ 38C \\ \hline D36 \\ \hline 10C2 \end{array}$$

$$\begin{array}{r} \times 1101 \\ 1011 \\ \hline 1101 \\ 11010 \\ 1101000 \\ \hline 10001111 \end{array}$$

$$\begin{array}{r} \times 29 \\ 19 \\ \hline 261 \\ 29 \\ \hline 551 \end{array}$$

$$\begin{array}{r} \times\ 35 \\ 23 \\ \hline 127 \\ 72 \\ \hline 1047 \end{array}$$

$$\begin{array}{r} \times 1D \\ 13 \\ \hline 57 \\ 1D \\ \hline 227 \end{array}$$

# Задачі

- Розглянути приклади переходу між записами чисел у системах числення з різною основою.
- Розглянути виконання арифметичних операцій у системах числення з різною основою.

# Подання цілих чисел

Форми (коди):

- беззнакова

N байтів (1,2,3,4) - від 0 до $2^{8N}-1$.

- Знакова (старший біт - знак числа 0 -”+”, 1- “-”)
  - прямий код (додатні числа, 0000…0 - 0111…1);
  - додатковий (від`ємні числа);

від`ємне число A → прямий код |A| → обернений код R(A) + 1 → додатковий код

# Приклад

Додатковий код числа −144.

Прямий двобайтовий код 0000 0000 1001 0000

Обернений — 1111 1111 0110 1111

Додатковий код — (+1) 1111 1111 0111 0000

# Приклади

| Число | Код | Число | Код |
|---|---|---|---|
| $2^{8N-1}-1$ | 011…11 | $-1$ | 111…11 |
| $2^{8N-1}-2$ | 011…10 | $-2$ | 111…10 |
| … | … | … | … |
| 1 | 000…01 | $-2^{8N-1}-1$ | 100…01 |
| 0 | 000…00 | $-2^{8N-1}$ | 100…00 |

# Принципи подання дійсних чисел

Дійсні числа найчастіше подаються у вигляді <*знак*><*порядок*><*мантиса*> (4, 6, 8, 10 байтів).

Пам`ять - 1 + d + r = 8N біт. Значення - s, e, m.

***Знак*** “+” - 0; “-” - 1.

***Порядок справжній -*** $e - (2^{d-1} - 1)$;

***Мантиса -*** $1 + m \times 2^{-r}$ *(нормалізована форма);*

# Приклад

Число - –12,375 ; N - 2 ; $d = 5,\ r = 10$

Зсув порядку $2^{5-1} - 1 = 2^4 - 1 = 15;$

$-12.375 = (-1100{,}011)_2 = (-1{,}100011)_2 \times 2^3$

$t = 3,\ m_1 = 0{,}100011$

$\Rightarrow \quad s = 1, \quad e = 3 + 15 = 18 = (10010)_2$

$m = 1000110000$

1 10010 1000110000

# Приклад

Число - 0,1. N - 2 ; $d = 5,\ r = 10$

$0{,}1 = (0{,}0001100110011001100\ldots)_2$.

Після нормалізації та округлення -

$(1{,}1001100110) \times 2^{-4}$

$e = -4 + 15 = 11 = (01011)_2$

0 01011 1001100110

!Число 0,1 представляється не точно.

# Висновок

Розглянули лише один із багатьох різних способів подання дійсних чисел. Розташування й довжини полів, а також їх обробка у поданні дійсних чисел можуть відрізнятися.

Дійсні числа, що подаються в комп'ютері, за будь-якого подання утворюють *обмежену підмножину множини раціональних чисел.*