МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ І СПОРТУ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

МЕТОДИЧНИЙ ПОСІБНИК

ДО ВИКОНАННЯ КОНТРОЛЬНИХ ЗАВДАНЬ

У СЕРЕДОВИЩІ PASCAL

2011

МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ І СПОРТУ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

МЕТОДИЧНИЙ ПОСІБНИК

ДО ВИКОНАННЯ КОНТРОЛЬНИХ ЗАВДАНЬ

У СЕРЕДОВИЩІ PASCAL

(навчально-методичний посібник для студентів заочного відділення)

Розглянуто
на засіданні кафедри ОМіП
протокол № ___от _____

Затверджено
навчально-видавничою
Радою ДонНТУ
протокол № ___ від _____

Донецьк 2011

УДК 681.3.06(071)

Методичний посібник до виконання контрольних завдань у середовищі PASCAL (навчально-методичний посібник для студентів заочного відділення) / Уклад.: Л.О.Лазєбна – Донецьк: ДонНТУ, 2011. – 91 с.

Методичний посібник містить варіанти контрольних завдань і теоретичний матеріал до їх виконання з теми «Алгоритмізація і програмування».

Розраховано для студентів усіх спеціальностей заочного відділення, що вивчають дисципліну «Інформатика», «Інформатика і комп'ютерна техніка», «Обчислювальна математика і програмування».

Укладачі: Л.О. Лазєбна, ст.викл.

Відп. за випуск В.М. Павлиш, проф.

З М І С Т

ВСТУП

Однією з найважливіших умов інтенсифікації праці в області управління виробництвом, в науці, економіці, діловодстві і інших сферах діяльності є інформація, яка заснована на широкому застосуванні засобів обчислювальної техніки для вирішення науково-технічних, проектних, економічних, управлінських, інженерних завдань.

Це визначає необхідність для підвищення рівня підготовки фахівців в області проектування комп'ютерів і їх програмного забезпечення, з одного боку, а з іншого боку, для підвищення рівня комп'ютерної письменності користувачів, тобто наукових, інженерно-технічних і адміністративних працівників, що не є фахівцями в області програмного забезпечення.

Для успішного використання комп'ютерів, грамотної постановки завдань на розробку програмних комплексів інженерові будь-якої спеціальності необхідне знання алгоритмізації, процесів підготовки і рішення задач на ЕОМ, етапів і принципів розробки програм.

Курс основ обчислювальної техніки і інформатики, що входить до складу обов'язкової навчальної підготовки інженерів, закладає базу комп'ютерної письменності фахівців.

1. ОСНОВИ АЛГОРИТМІЗАЦІЇ

1.1 Етапи підготовки і рішення задачі

Процедура підготовки і рішення задач з використанням комп'ютера – це достатньо складний і трудомісткий процес, що складається з наступних етапів:

1. постановка задачі (завдання, яке належить вирішувати, формулюється користувачем або надходить у вигляді завдання);
2. математичне формулювання задачі;
3. розробка алгоритму рішення задачі;
4. написання програми на мові програмування;
5. підготовка початкових даних;
6. введення програми і початкових даних у ПК;
7. відлагодження програми;
8. тестування програми;
9. рішення задачі і обробка результатів.

Оскільки умови більшості завдань дані у математичному формулюванні, тому необхідність у виконанні етапів 1 і 2 відпадає і можна відразу приступити до розробки алгоритму рішення задачі.

1.2 Алгоритм і форми його зображення

Під ***алгоритмом*** розуміється опис послідовності арифметичних і логічних дій над числовими значеннями змінних, що приводить до обчислення результату рішення задачі при зміні початкових даних в достатньо широких межах. Таким чином, при розробці алгоритму рішення задачі математичне формулювання перетвориться в процедуру рішення, що є послідовністю арифметичних дій і логічних зв'язків між ними.

При цьому алгоритм володіє наступними властивостями: детермінірованістю, що означає - застосування алгоритму до одних і тих же початкових даних повинне приводити до одного і того ж результату;

масовістю, що дозволяє отримувати результат при різних початкових даних; результативністю, що забезпечує отримання результату через кінцеве число кроків.

Найбільш наочним способом опису алгоритму є його зображення у вигляді блок-схем. При цьому алгоритм зображується послідовністю блоків, що виконують певні функції (див. додаток 1), які з'єднані між собою лініями потоку. Усередині блоків указується інформація, що характеризує виконувані ними функції. Блоки схеми мають крізну нумерацію.

На етапі 4 складається програма на одній з мов програмування. При описі програм необхідно використовувати характерні прийоми програмування і зважати на специфіку конкретної мови.

Етапи алгоритмізації і програмування є найбільш трудомісткими, тому їм приділяється велика увага.

Відлагодження програм полягає у виявленні і виправленні помилок, допущених на всіх етапах підготовки задачі до рішення на ПК.

Синтаксичні помилки виявляються транслятором, який видає повідомлення, указуючи місце і тип помилки. Виявлення семантичних помилок здійснюється на етапі тестування програми, в якому перевіряється правильність виконання програми на спрощеному варіанті початкових даних або за допомогою контрольного розрахунку.

Обробка результатів рішення задачі здійснюється або за допомогою ЕОМ (побудова таблиць, графіків), або уручну. Результати, що виводяться, оформляються у вигляді, зручному для сприйняття людиною.

1.3 Типові структури алгоритмів

1.3.1 Лінійні і розгалужені алгоритми

Простим прикладом алгоритму є алгоритм лінійної структури.

Алгоритм лінійної структури – це алгоритм, в якому блоки виконуються послідовно один за одним, в порядку, заданому схемою. Такий порядок виконання називається ***природним***.

Приклад 1.

Обчислити висоти трикутника із сторонами a, *b, c,* використовуючи формули:

$$h_a = \frac{2}{a}\sqrt{p(p-a)(p-b)(p-c)}$$
$$h_b = \frac{2}{b}\sqrt{p(p-a)(p-b)(p-c)}$$
$$h_c = \frac{2}{c}\sqrt{p(p-a)(p-b)(p-c)}$$
$$\text{де} \quad p = (a+b+c)/2$$

При рішенні даної задачі для виключення повторень слід обчислювати висоти не за приведеними вище формулами безпосередньо, а використовуючи проміжну змінну:

$$t = 2\sqrt{p(p-a)(p-b)(p-c)},$$
$$\text{тоді} \quad h_a = t/a, \;\; h_b = t/b, \;\; h_c = t/c$$

Схема алгоритму рішення наведена на рис.1.

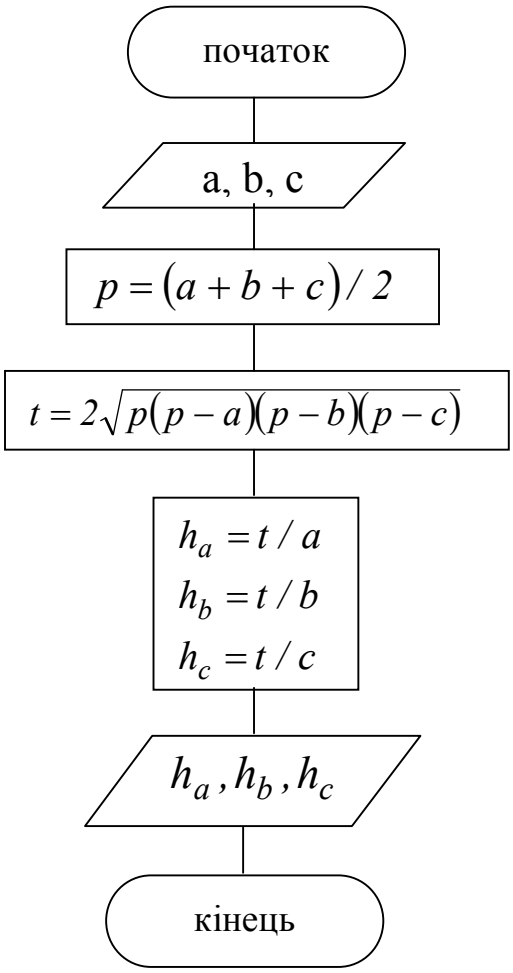


Рис. 1 - Алгоритм лінійної структури.

Проте рішення абсолютної більшості інженерних задач неможливо звести до алгоритмів лінійної структури. Часто залежно від яких-небудь проміжних результатів обчислення здійснюється або по одних, або по інших формулах, тобто залежно від виконання деякої логічної умови обчислювальний процес здійснюється по одній або по іншій гілці.

Алгоритм такого обчислювального процесу називається ***алгоритмом розгалуженої структури***.

Приклад 2.

Обчислити

$$z = \begin{cases} \sin(x), & якщо \quad x \le a \\ \cos(x), & якщо \quad a \prec x \prec b \\ tg(x), & якщо \quad x \ge b \end{cases}$$

З математичного формулювання задачі можна побачити, що обчислювальний процес має три гілки. За допомогою умовного блоку перевіряється виконання тільки однієї умови, по якій вибирається вираз для реалізації однієї гілки. Тому для визначення того, по якій з двох гілок, що залишилися, повинен йти обчислювальний процес після невиконання першої умови, необхідно використовувати ще один умовний блок. Після обчислення по будь-якій з формул здійснюється перехід в загальну гілку до блоку виведення результату.

Схема алгоритму наведена на рис.2.

Приклад 3.

Обчислити

$$z = \frac{x^3}{y}, \quad де \quad y = \sin(nx) + 0{,}5$$

Здавалося б, що рішення цієї задачі описує алгоритм лінійної структури. Проте для задоволення властивості масовості і результативності алгоритму необхідно, щоб при будь-яких початкових даних був отриманий результат або повідомлення про те, що задача не може бути вирішена при заданих даних. Дійсно, якщо y=0, то задача не може бути вирішена, оскільки

ділити на нуль не можна. Тому в алгоритмі необхідно передбачити таку умову і видати як результат інформацію про те, що y=0.

Схема алгоритму наведена на рис.3.

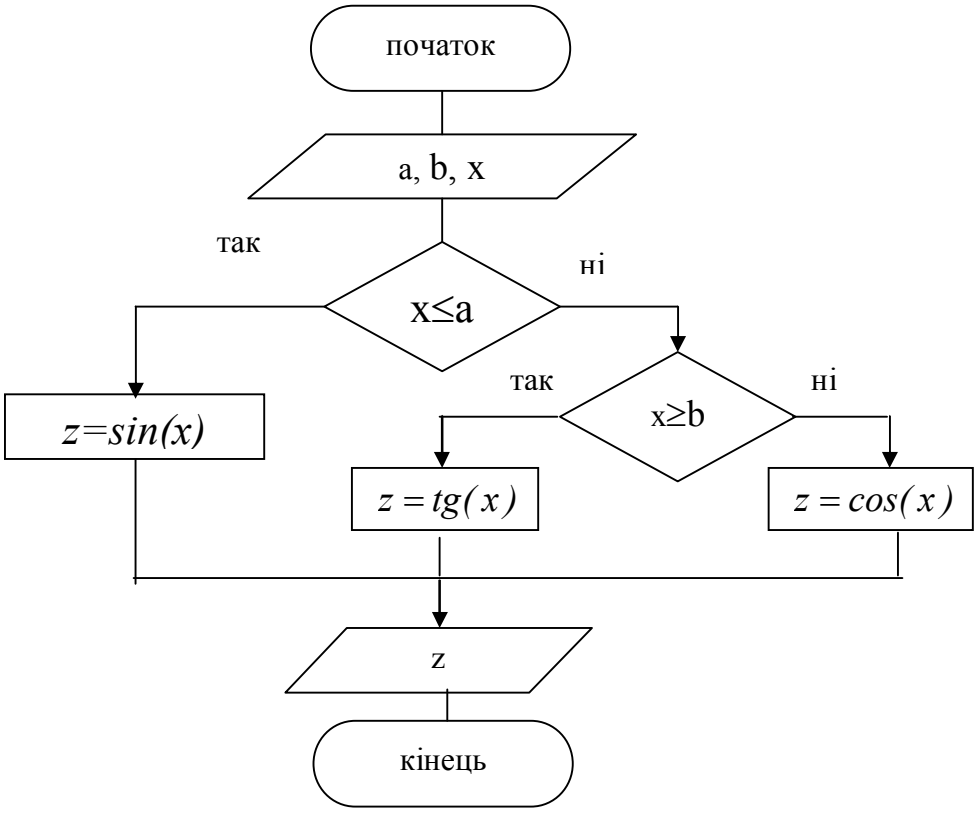


Рис.2 - Алгоритм розгалуженої структури.

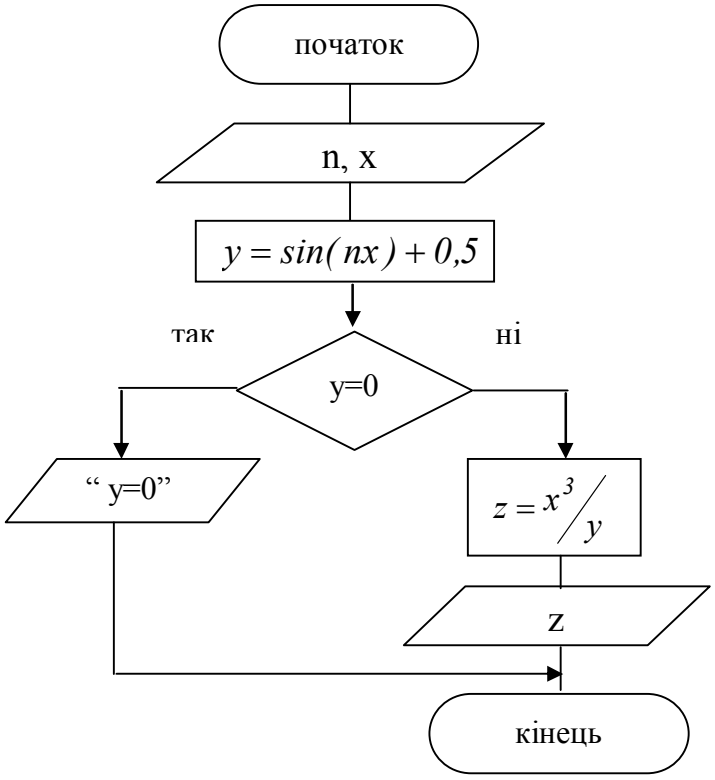


Рис.3 - Алгоритм розгалуженої структури.

1.3.2 Алгоритми циклічної структури

Часто при рішенні задач виникає необхідність багато разів обчислювати значення по одних і тих же математичних залежностях для різних вхідних величин. Багато разів повторюваний етап обчислювального процесу називається ***циклом***, а обчислювальний процес, що містить такі етапи, – ***циклічним***.

Для організації циклічних обчислень необхідно виконати наступні дії:

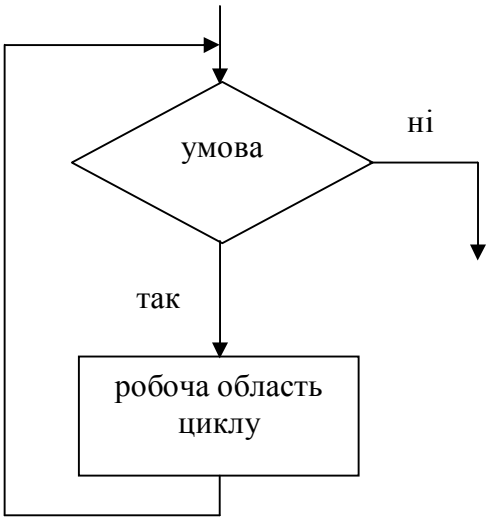
- задати перед циклом початкове значення змінної, яка буде змінюватися у циклі;
- змінювати змінну перед кожним новим повторенням циклу;
- перевіряти умову закінчення або повторення циклу, тобто переходити до його початку, якщо він не закінчений, або виходити з нього після закінчення.

Змінна, що змінюється у циклі і від якої залежить умова виходу з циклу, називається ***параметром циклу***.

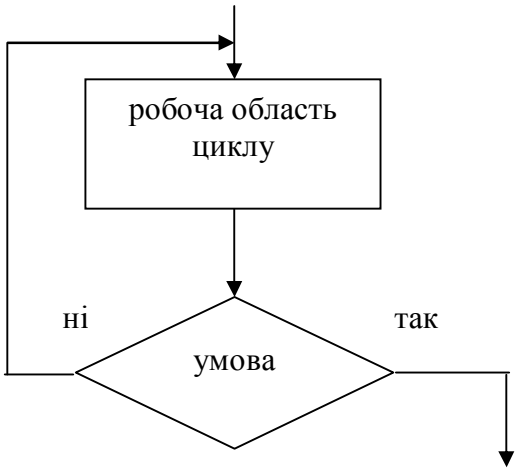
Можливі три способи організації циклічних структур алгоритмів:

1) цикл з передумовою або цикл «ПОКИ»
2) цикл з постумовою або цикл «ПОВТОРЮВАТИ … ДО»
3) цикл з відомим числом повторень.

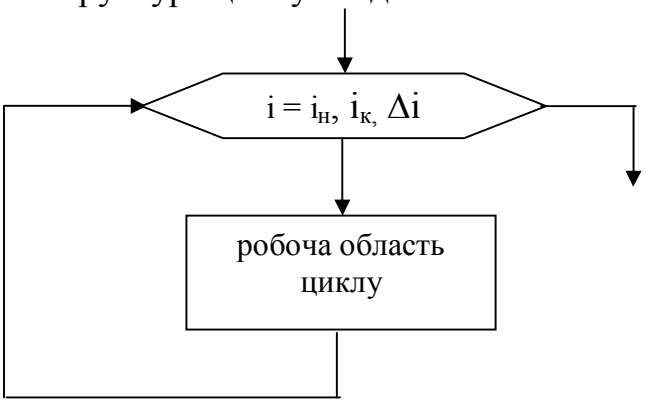
Структура циклу з передумовою:



Структура циклу з постумовою:



Структура циклу з відомим числом повторень:



Приклад 4.

Обчислити значення y:

$$y = sin(ax)$$

для $1 \le x \le 100$ $\Delta x = 1$ $a = 1.67$

Початкові дані: $x_н$=1 $x_к$=100 Δx=1 a=1.67

Схеми організації циклів наведені на рис. 4 і 5.

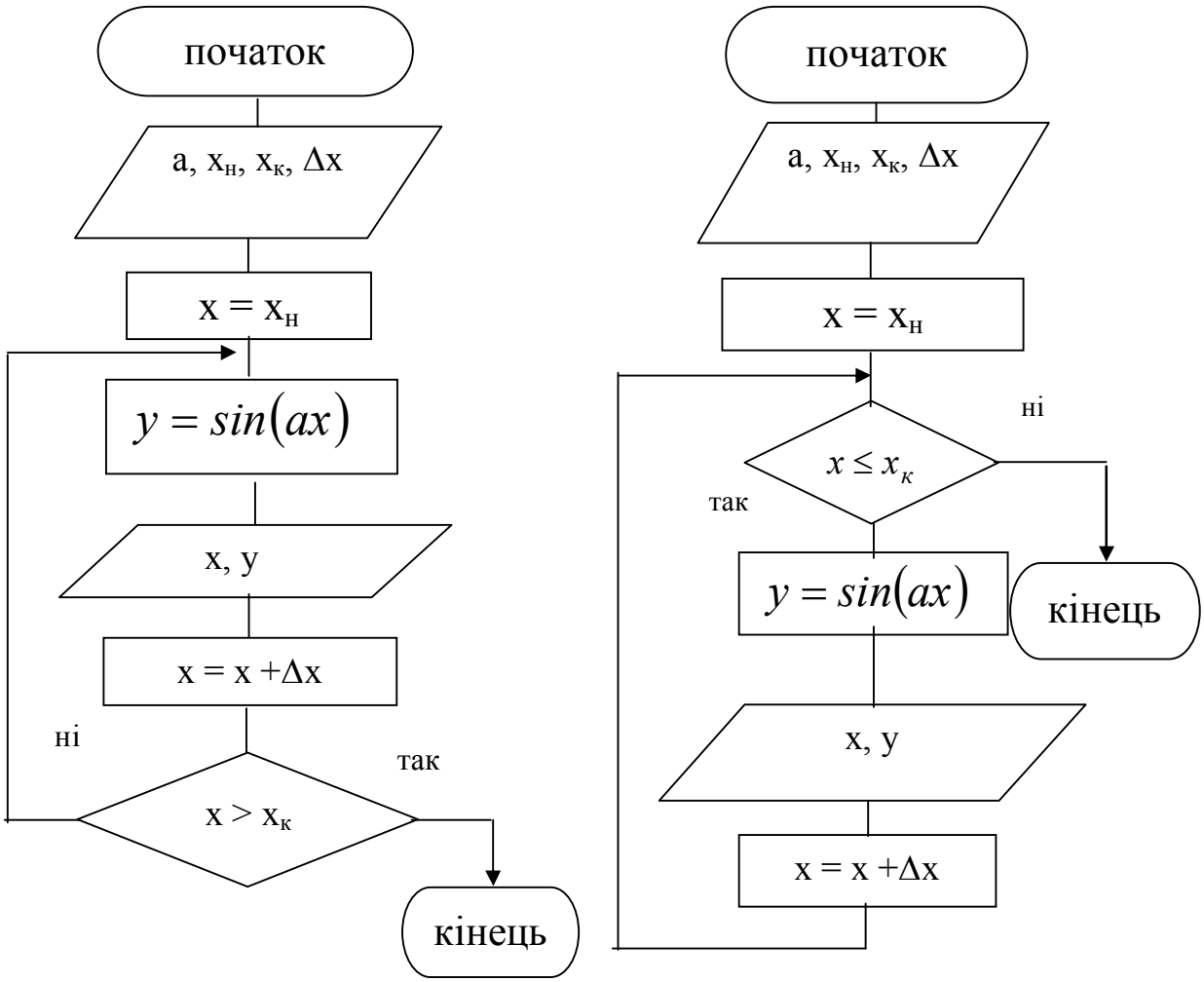


Рис.4 - Цикл з постумовою і з передумовою

Для організації циклу з відомим числом повторень використовується блок модифікації. У цьому блоці об'єднано 3 дії: установка початкового значення параметра, його зміна у процесі виконання циклу і перевірка умови закінчення циклу.

На рис.5 відображається алгоритм, в якому параметром циклу є змінна i, що змінюється від 1 до m, де m – кількість повторень циклу, що визначується по формулі:

$$m = \left\lfloor \frac{x_к - x_н}{\Delta x} \right\rfloor + 1$$

Дужки у формулі указують на те, що береться ціла частина від числа.

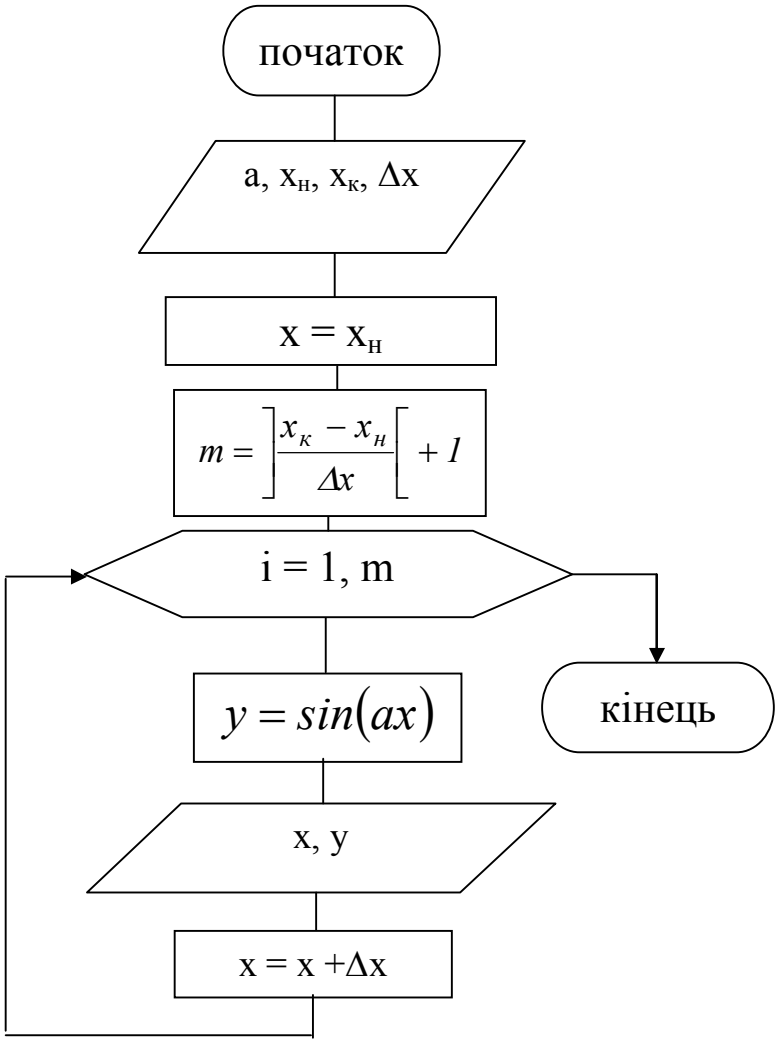


Рис.5 - Цикл з відомим числом повторень

1.3.3 Алгоритми із структурою вкладених циклів

На практиці часто зустрічаються завдання, в яких одночасно змінюється декілька параметрів. В цьому випадку з'являються структури з вкладеними циклами. Усередині простого циклу може знаходитися ще один або декілька циклів. При цьому перший цикл називається зовнішнім, а вкладені в нього – внутрішніми.

Правила їх організації нічим не відрізняються від правил організації простого циклу, причому один і той же цикл може бути зовнішнім по відношенню до одного і внутрішнім по відношенню до іншого циклу.

Зовнішній і внутрішній цикл мають свої параметри.

Для кожного значення параметра зовнішнього циклу параметр внутрішнього циклу приймає послідовно всі свої значення. Іншими словами, завжди виконується в першу чергу самий внутрішній цикл.

Межі внутрішнього циклу не можуть виходити за межі зовнішнього по відношенню до нього циклу, але можуть співпадати з ними.

Приклад 5.

Обчислити значення x та y

$$x = \frac{2{,}5a - c}{a^2 - 2ac} \qquad y = \frac{|x + 2c|}{c^2 + 1}$$

$$-3 \le a \le 3; \quad \Delta a = 1{,}5;$$
$$-1 \le c \le 1; \quad \Delta c = 0{,}2$$

Початкові дані: $a_н$, $a_к$, Δa, $c_н$, $c_к$, Δc

Для організації зовнішнього і внутрішнього циклів використовуються цикли з постумовою.

Схема алгоритму наведена на рис.6.

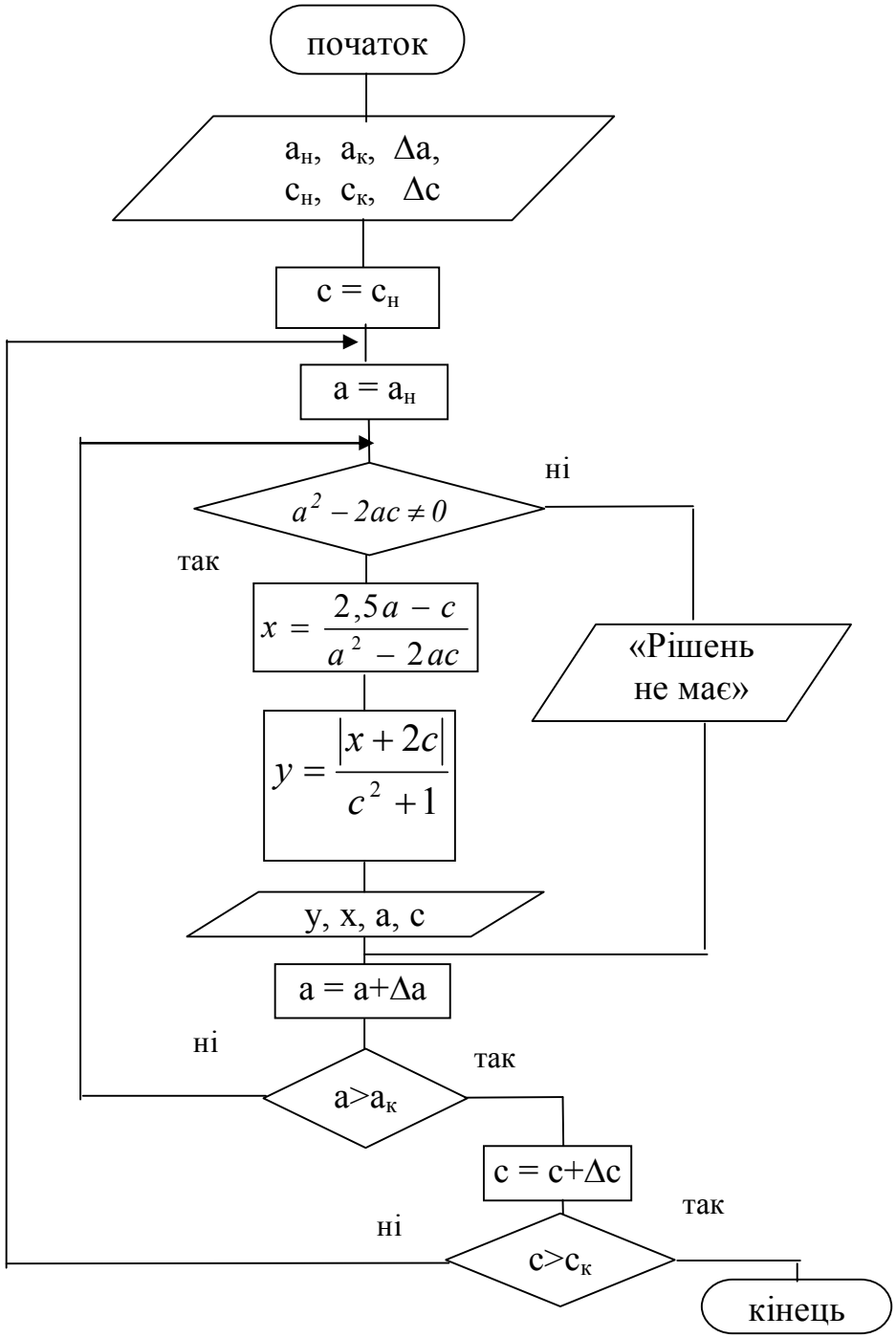


Рис.6 - Організація алгоритму із структурою вкладених циклів

1.3.4 Алгоритми визначення суми, кількості і добутку

Часто разом з обчисленням значень величини або сукупності величин необхідно визначити суму, кількість або добуток деяких значень.

При обчисленні кількості, суми або добутку доцільно використовувати принцип поступового накопичення. При цьому заздалегідь необхідно задати початкове значення: для суми – нульове, для кількості – нульове, для добутку – одиницю.

Приклад 6.

Обчислити значення y:

$$y = \frac{a^2 + x^2}{\ln 3 - |x|} \qquad -0.5 \le x \le 2.5 \quad \Delta x = 0.1 \quad a = 3.8$$

Визначити: кількість (k) y<0 $P = \prod_{y<0} y$ $S = \sum y$

Схема алгоритму наведена на рис.7.

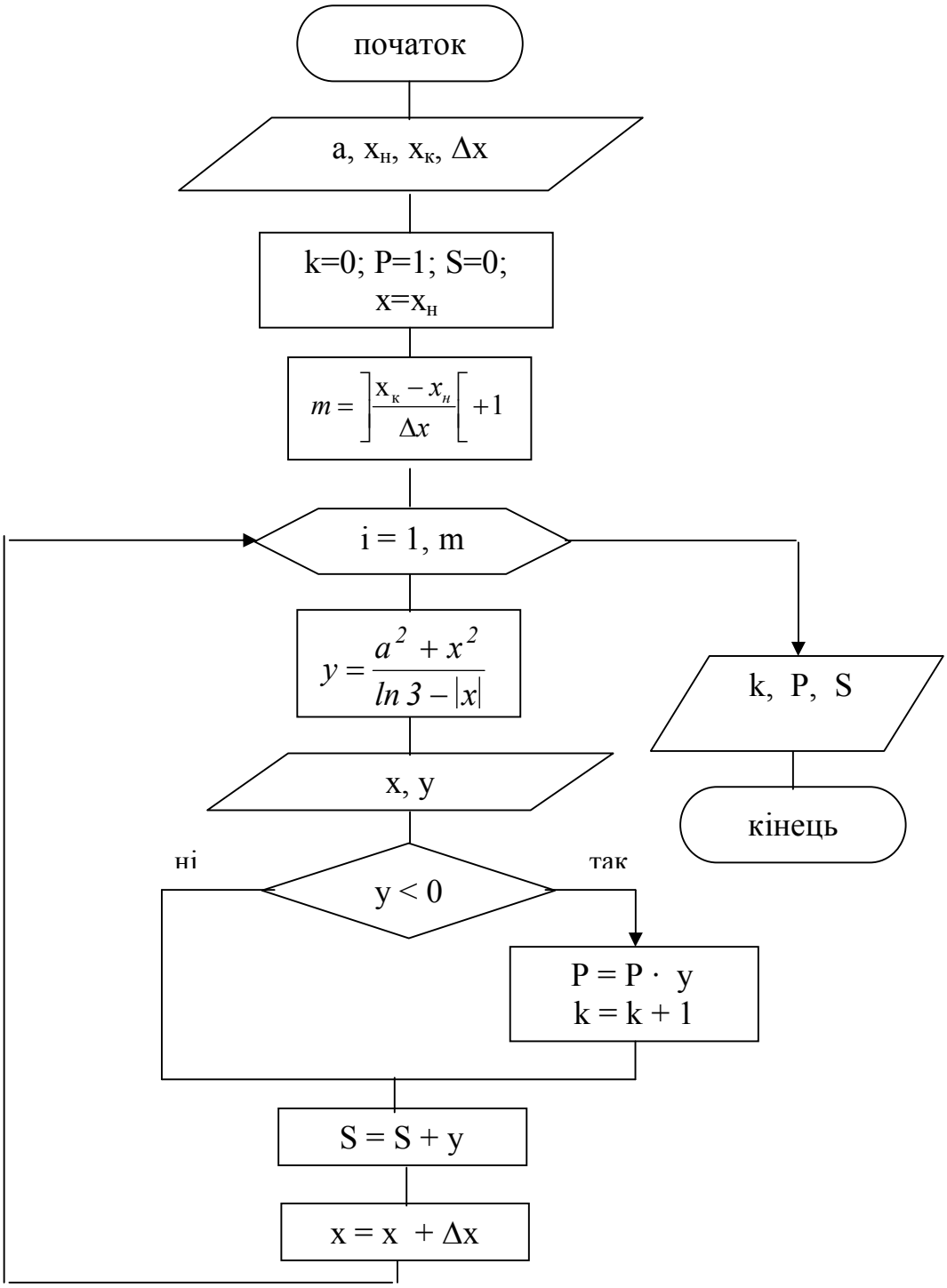


Рис.7 - Організація алгоритму визначення суми, кількості і добутку

Перед організацією циклу задаються початкові значення для величин k, S, P. У циклі відбувається накопичення добутку від'ємних значень y і підрахунок їх кількості, а також накопичення суми всіх значень y. Виведення значень k, S, P здійснюється після закінчення циклу, тобто значення необхідно вивести один раз, коли вони будуть остаточно обчислені.

1.3.5 Алгоритми формування і обробки одновимірних масивів

У математиці, економіці, інформатиці часто використовуються впорядковані набори даних, наприклад послідовності чисел, таблиці, списки прізвищ. Для обробки даних одного типу вводиться поняття масиву.

Масив – це впорядкована сукупність змінних одного типу, позначених одним ім'ям. Кожен елемент масиву позначається ім'ям масиву з індексом. Індекс визначає місцеположення елементу у масиві, наприклад X_1 – це перший елемент масиву X. Елементи масиву впорядковані по значеннях індексу: $X=(X_1, X_2, \ldots, X_n)$, тобто індекс елементів масиву змінюється від 1 до n. Змінна n указує на кількість елементів в масиві X, тобто на розмірність масиву.

Якщо кожен елемент масиву містить тільки один індекс, то такий масив називається **одновимірним**.

Приклад 7.

Визначити кількість ненульових елементів в масиві У, елементи якого обчислюються за формулами:

$$y_i = \begin{cases} a \ln x_i, & якщо \quad x_i \geq 1 \\ \sqrt{x^2{}_i + 2{,}5}, & якщо \quad x_i < 1 \end{cases} \qquad \text{де} \quad i = 1, M$$

Схема алгоритму наведена на рис.8.

Приклад 8.

Записати елементи масиву $x = (x_1, x_2, \ldots, x_{15})$, які менш за 1 в масив $y = (y_1, y_2, \ldots, y_k)$. Визначити кількість і суму елементів масиву У.

Схема алгоритму наведена на рис.9.

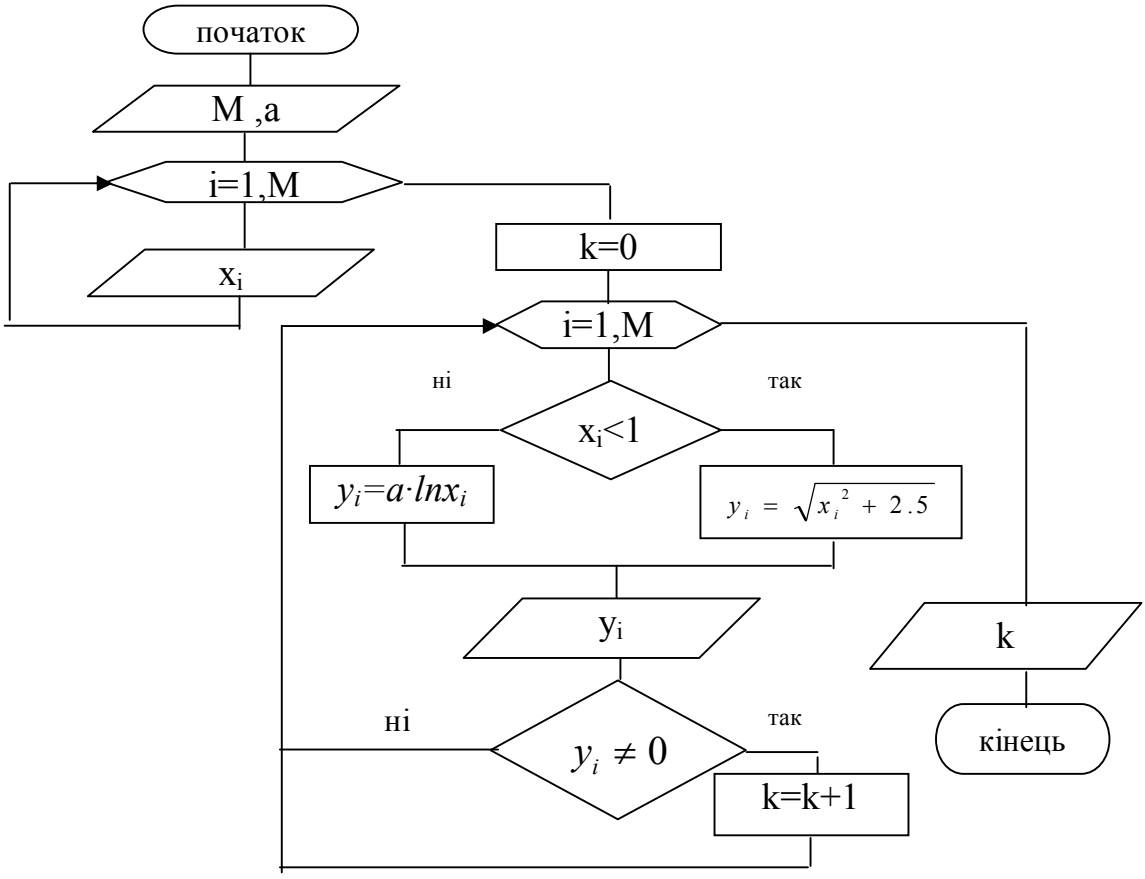


Рис.8 - Алгоритм обчислення елементів масиву У і визначення кількості ненульових елементів в даному масиві

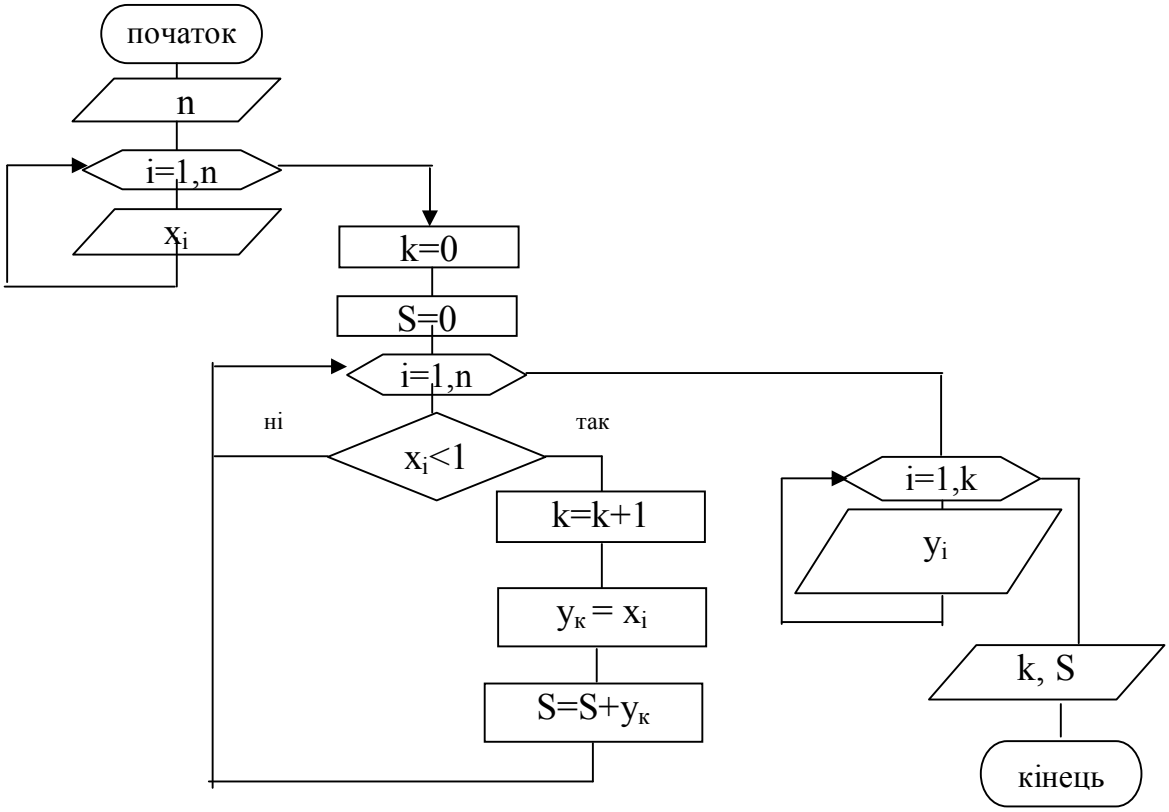


Рис.9 - Алгоритм формування елементів масиву У

1.3.6 Алгоритми формування і обробки двовимірних масивів

Двовимірний масив - це деяка матриця з певним числом рядків і стовпців. Кількість рядків і стовпців визначають розмірність двовимірного масиву.

Кожен елемент масиву визначається двома індексами: перший з них указує номер рядка, а другий номер стовпця, на перетині яких розташовується даний елемент, наприклад X_{13} – це елемент масиву X, що розташовується на перетині першого рядка і третього стовпця.

Приклад 9.

Для масиву *A(n,m)* знайти відсоток нульових елементів.

Початкові дані: *A* – двовимірний масив;

n – число рядків в масиві *A*;

m – число стовпців в масиві *A*.

Формула визначення відсотка нульових елементів: $p = \frac{k}{n \cdot m} \cdot 100\%,$ де *k* – кількість нульових елементів в масиві *A*.

Схема алгоритму наведена на рис.10.

Приклад 10.

З масиву *A(n,m)* сформувати масив *B(m)*, кожен елемент якого рівний кількості від'ємних елементів відповідного стовпця масиву *A*.

Початкові дані: *A* – двовимірний масив;

n – число рядків в масиві *A*;

m – число стовпців в масиві *A*.

Схема алгоритму наведена на рис.11.

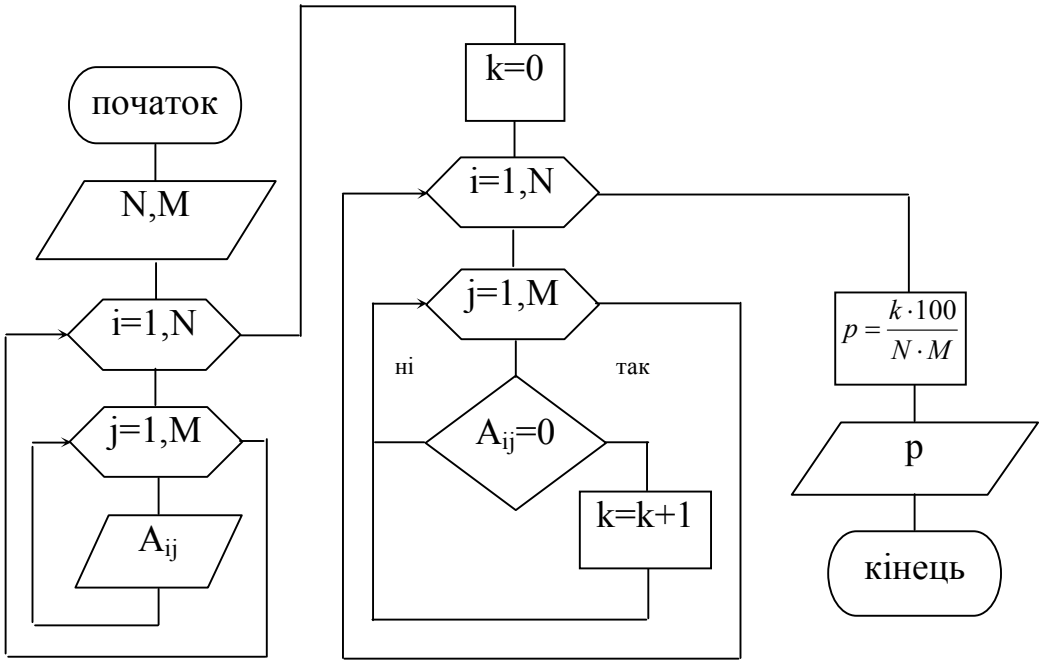


Рис.10 - Алгоритм визначення відсотка нульових елементів в масиві А

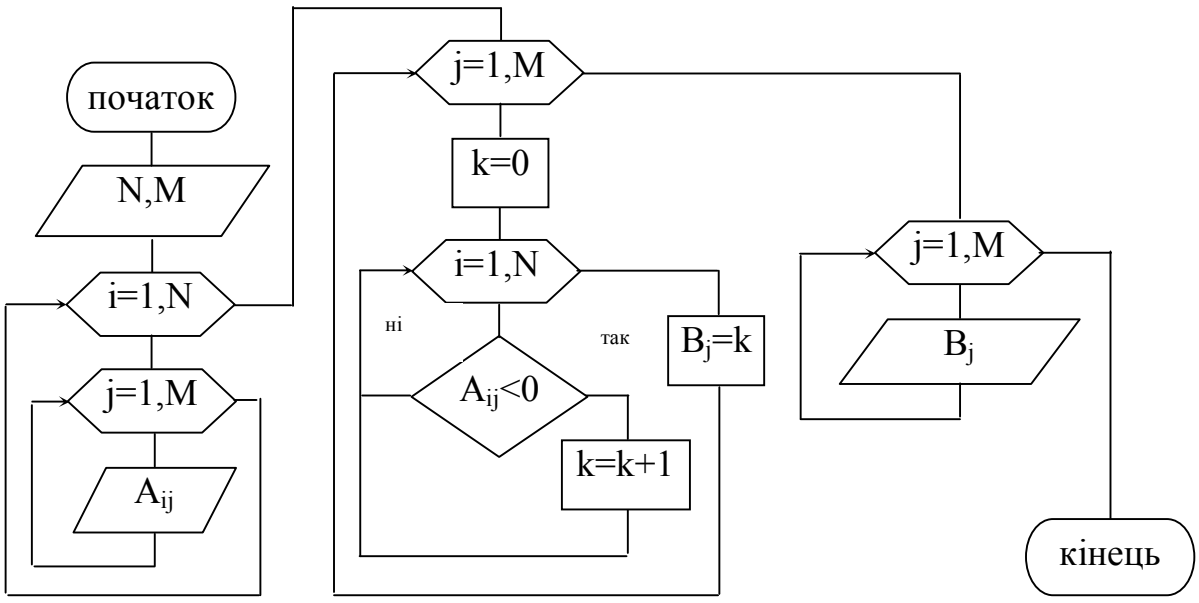


Рис.11 - Алгоритм формування елементів масиву В

1.3.7 Алгоритми визначення найбільшого і найменшого значення

Найбільше або найменше значення зазвичай визначається для сукупності величин, тобто масиву. Їх знаходження здійснюється в циклі в результаті порівняння деякого поточного значення з найбільшим (найменшим) зі всіх попередніх. При цьому якщо поточне значення більше найбільшого (менше найменшого) зі всіх попередніх, то найбільшому (найменшому) привласнюється значення поточного.

Приклад 11.

Для масиву Z(n) визначити максимальний (max) і мінімальний (min) елементи.

Схема алгоритму наведена на рис.12.

1.3.8 Алгоритми, які містять допоміжні підзадачі

При складанні складніших програм неминуче виникають труднощі, обумовлені наявністю великої кількості об'єктів, безліч різних зв'язків між цими об'єктами. Тому краще розбивати програму на окремі смислові блоки.

У практиці часто зустрічаються випадки, коли по ходу виконання розрахунків в завданнях доводиться проводити одні і ті ж обчислення, але при різних початкових даних.

Щоб виключити повторення однакових дій і зробити тим самим алгоритм простіше і зрозуміліше, можна виділити ці дії, що повторюються, в самостійну частину у вигляді допоміжної підзадачі, яка може бути використана багато разів в міру необхідності. Така допоміжна підзадача, що реалізовує певний алгоритм і що допускає звернення до неї з різних частин блок-схеми основної задачі, називається підпрограмою.

Підпрограми оформляються у вигляді окремих алгоритмів, що мають чітко позначений вхід і вихід.

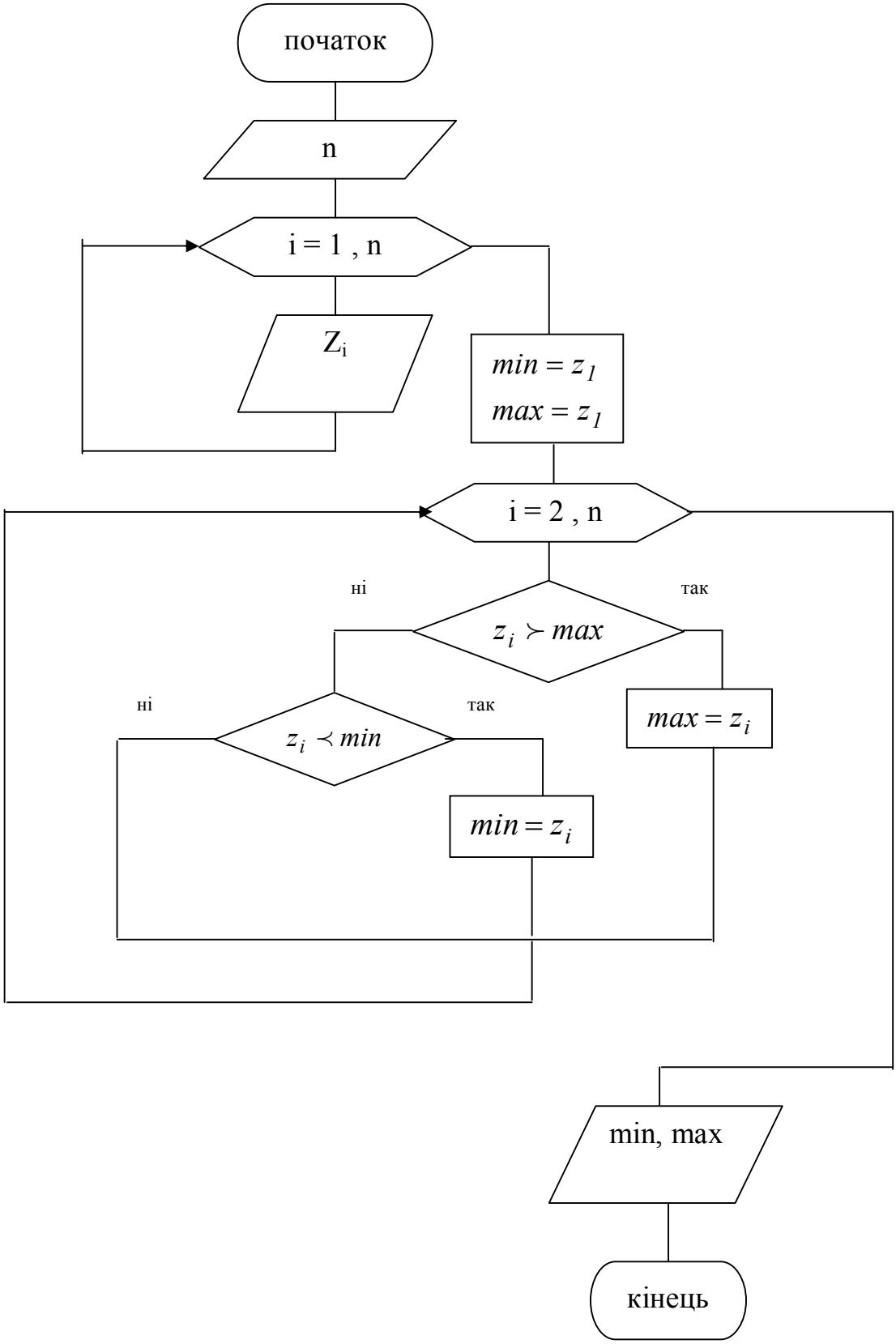


Рис.12- Алгоритм визначення найбільшого і найменшого значення в масиві Z

При складанні блок-схеми основної задачі обов'язково передбачити звернення до підпрограми.

Підпрограми бувають двох видів: процедури і функції. Основна відмінність їх полягає в тому, що результат роботи процедури – це цілий набір значень, а результат виконання функції – це одне значення.

Приклад 12.

Обчислити різницю середніх арифметичних значень елементів масиву $X = (X_1, X_2 .., X_{50})$ з парними індексами і елементів з непарними індексами:

$$R = \frac{1}{25} \sum_{i=2,4,\ldots}^{50} X_i - \frac{1}{25} \sum_{i=1,3,\ldots}^{49} X_i$$

Пояснення.

Для обчислення R необхідно двічі визначити середньо арифметичне значення 25 чисел. Тому в якості допоміжної підзадачі можна виділити обчислення середнього арифметичного значення 25 чисел:

$$S = \frac{1}{25} \sum_{j=k,k+2,\ldots.}^{n} X_j$$

Перший раз необхідно задати k = 1 і n = 49.

Другий раз необхідно задати k = 2 і n = 50.

Допоміжну підзадачу можна оформити у вигляді підпрограми функції.

Блок-схема алгоритму допоміжної підзадачі починається стрілкою «вхід» з вказівкою списку початкових даних, а закінчується стрілкою «вихід» із списком вихідних даних (див. рис.13).

Результат функції привласнюється змінним S1, S2.

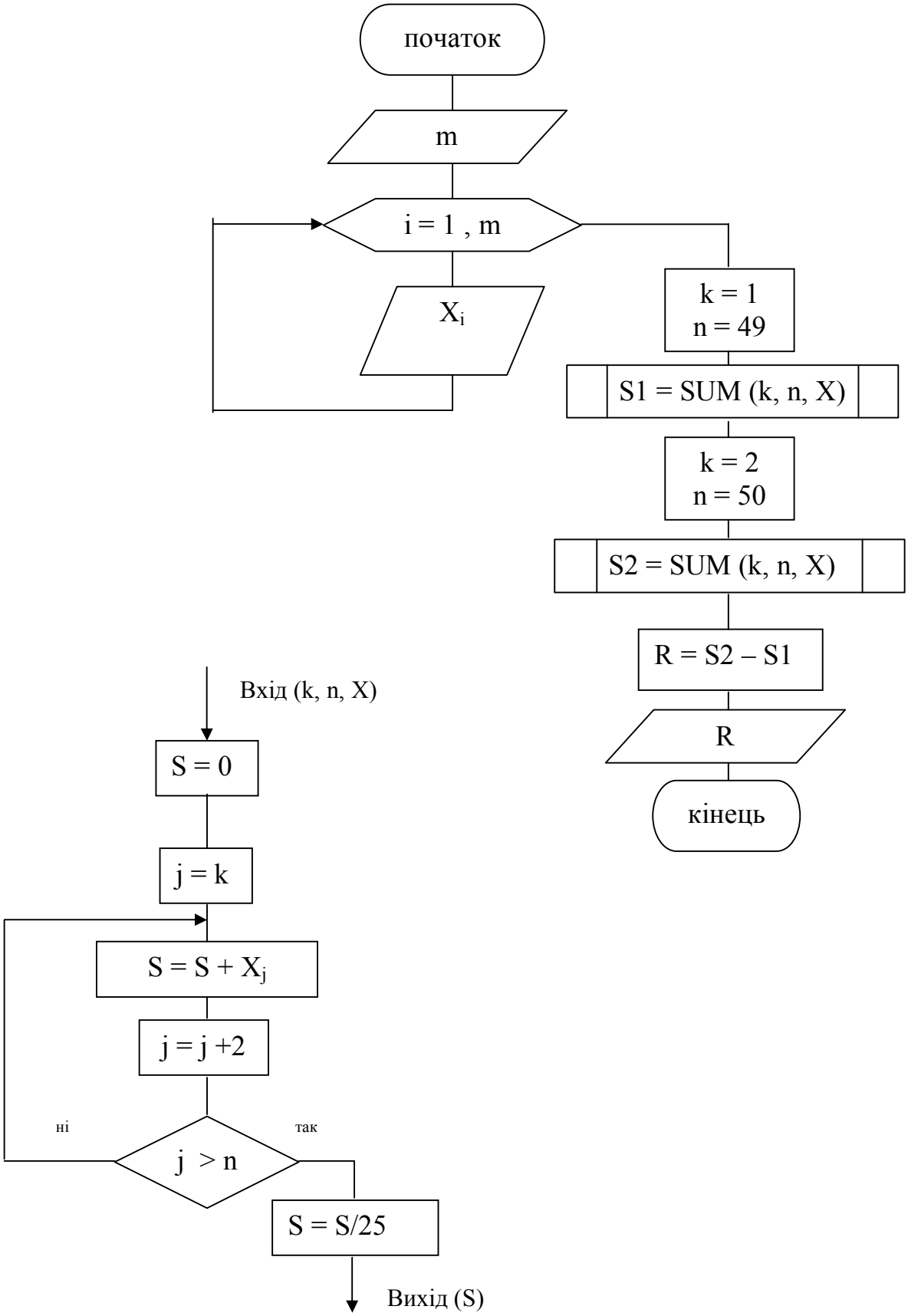


Рис.13 - Алгоритм основної задачі і допоміжної функції SUM

Приклад 13.

Сформувати масиви, елементи яких визначаються по формулах:

$$a_k = 2.8(k - 3.6)^2 + \sqrt{k + 2}$$
$$b_j = 5.4(j + 2.6)^2 + \sqrt{j + 2}$$
$$k = 1, m \qquad j = 1, n$$

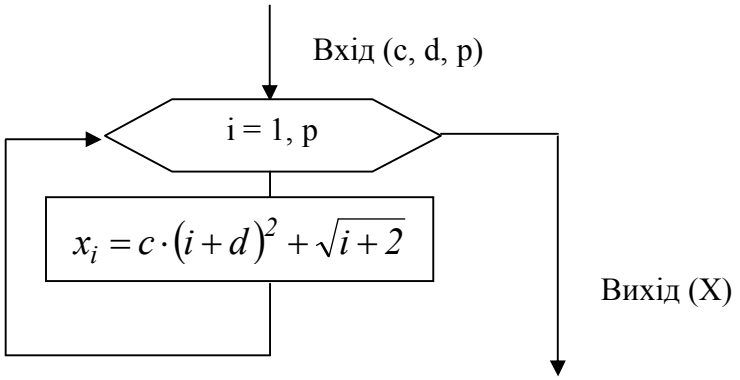
Пояснення.

В якості допоміжної підзадачі можна виділити обчислення елементів масиву за формулою: $x_i = c \cdot (i + d)^2 + \sqrt{i + 2}$ і виведення елементів масиву.

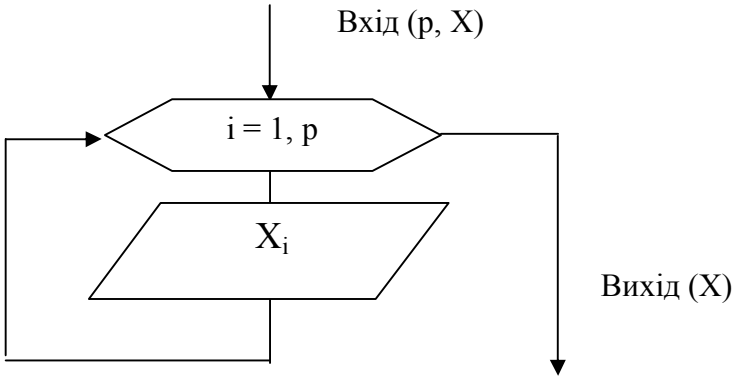
Допоміжну підзадачу можна оформити у вигляді підпрограми процедури.

Схема алгоритму основної задачі наведена на рис.14.

Процедура розрахунку елементів масиву – FORM



Процедура виведення елементів масиву – VUVOD



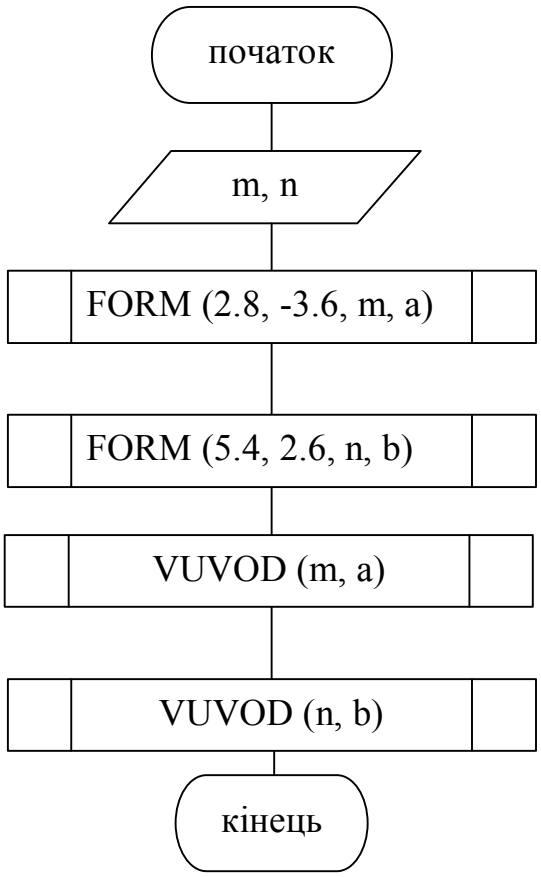


Рис.14 - Алгоритм основної задачі

2. МЕТОДИЧНІ ВКАЗІВКИ ПО ПРОГРАМУВАННЮ У СЕРЕДОВИЩІ TURBO PASCAL

2.1 Середовище програмування Turbo Pascal

При програмуванні важливо знати не тільки мову програмування, але і добре орієнтуватися в середовищі розробки. Turbo Pascal – це середовище програмування, орієнтоване на мову Pascal, що дозволяє створювати, редагувати, відладжувати і виконувати програми. Обов'язковим файлом при роботі з системою Turbo Pascal є файл turbo.exe.

Після завантаження файлу turbo.exe на екрані з'являється вікно середовища програмування Turbo Pascal (рис.2.1).

У верхньому рядку екрану відображається головне меню, потім вікно редагування Edit і в нижньому рядку – рядок підказки про призначення функціональних клавіш.

Всі дії, які необхідні для роботи в системі Turbo Pascal (виклик програми, компіляція, запуск, відлагодження і т.д.) доступні через головне меню, перехід в яке здійснюється клавішею **F10**.

Для завершення роботи з системою Turbo Pascal необхідно натиснути комбінацію клавіш **ALT+X**, що відповідає команді **Quit** (пункт меню File).

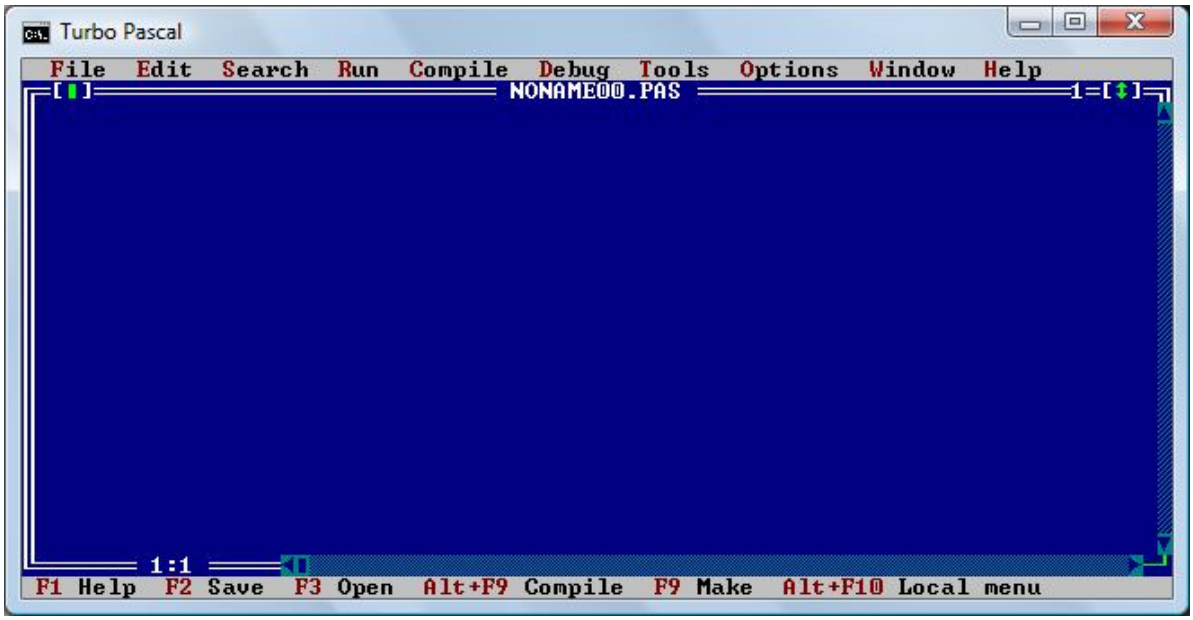


Рис.2.1 - Вид середовища програмування Turbo Pascal

На екрані одночасно може бути відкрито декілька вікон. Одне з видимих вікон завжди є поточним, що підтверджується подвійною лінією по його верхній межі і яскравішим кольором заголовка (рис.2.2).

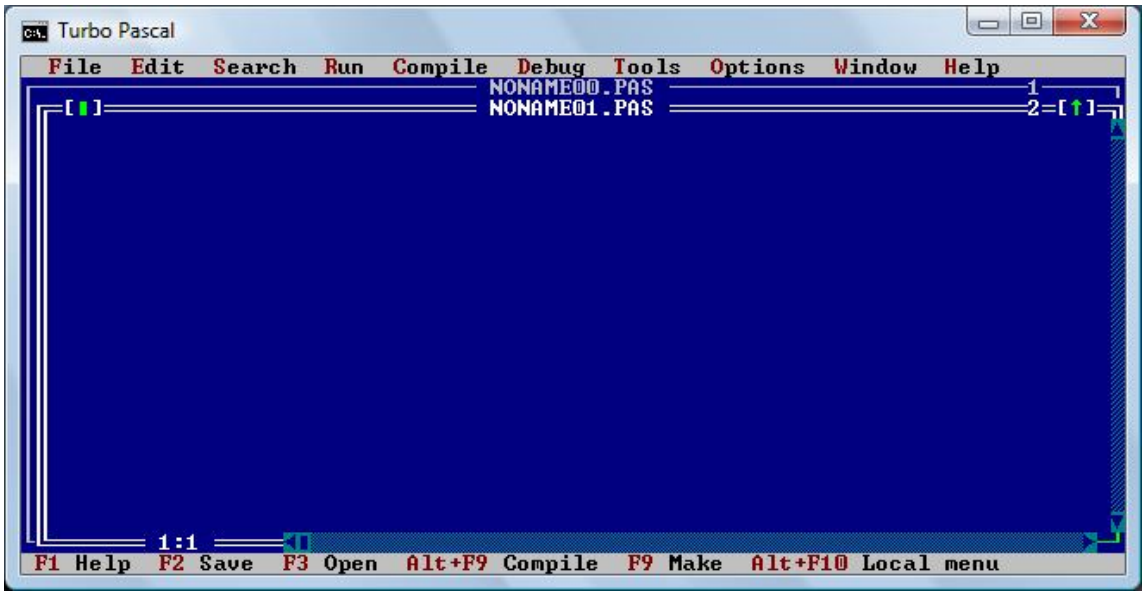
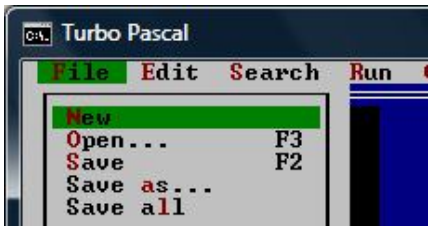


Рис.2.2 - Середовище програмування ТР містить декілька відкритих вікон

Послідовність дій для створення файлу.

1. Запустити середовище Turbo Pascal.
2. Написати текст програми.
3. У головному меню вибрати пункт меню **File** і команду **Save as…** для збереження файлу.



З'явиться діалогове вікно, в якому необхідно ввести ім'я файлу і вказати його місце зберігання (рис.2.3).

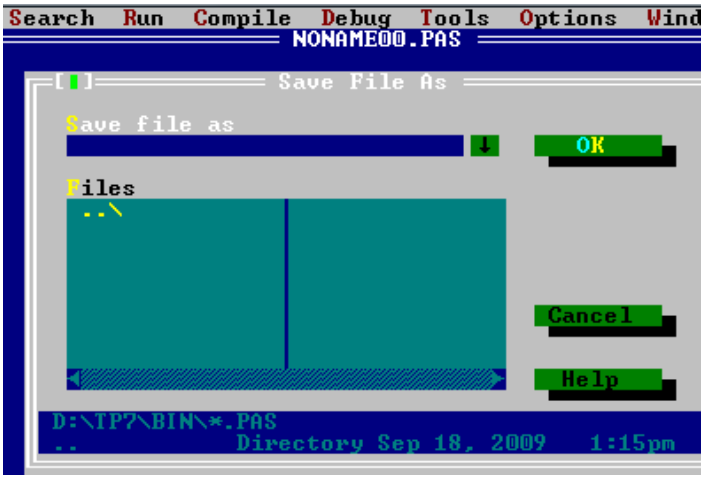
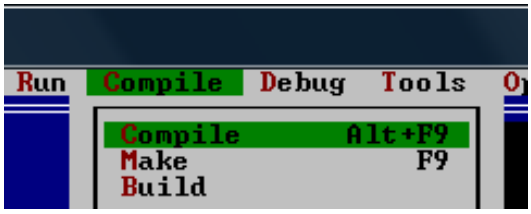
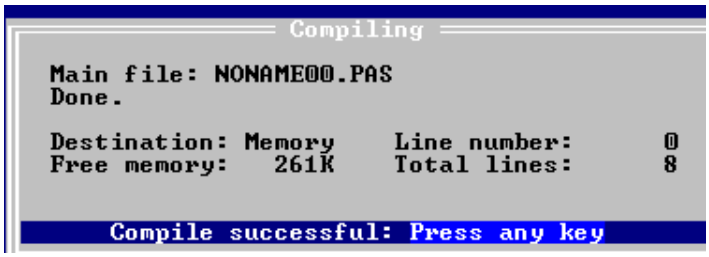


Рис.2.3 - Діалогове вікно для збереження файлу

4. Виконати компіляцію програми, використовуючи команду **Compile** (пункт меню **Compile**) або натиснути комбінацію клавіш **ALT+F9** (компіляція програми) чи **F9** (компіляція програми і створення завантажувального модуля).



Після компіляції програми на екрані з'являється вікно з результатами компіляції.



У разі виявлення помилок в тексті програми, компіляція припиняється і з'являється повідомлення про помилку. Необхідно виправити помилку і знову виконати компіляцію.

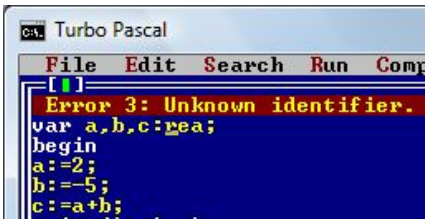
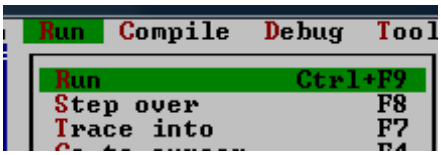


Рис.2.4. Повідомлення про помилки при виконанні компіляції програми

5. Запустити програму на виконання, використовуючи команду **Run** (пункт меню **Run**) або натиснути комбінацію клавіш **Ctrl+F9**. Команда **Trace info** використовується для відрядкового виконання програми.



Іноді на стадії виконання програми теж виникають помилки (неприпустимі операції – наприклад, ділення на нуль). При виникненні таких помилок виконання програми буде перервано. З'явиться код помилки і дана фізична інтерпретація її виникнення. Необхідно виправити помилку і знову виконати компіляцію.

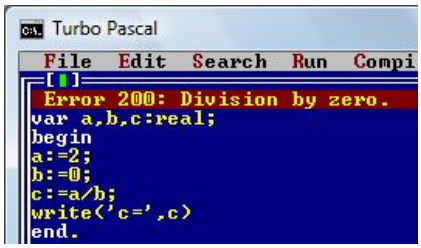


Рис.2.5. Повідомлення про помилки при виконанні програми

Щоб побачити результати розрахунків необхідно натиснути комбінацію клавіш **ALT+ F5**.

2.2 Алфавіт мови Pascal

Алфавіт – це сукупність допустимих в мові символів або груп символів, що розглядаються як єдине ціле:

- букви від A до Z, від a до z;
- цифри від 0 до 9;
- спеціальні символи + - * / = < > . : ; ‘ ^ $ # _ { } [ ] ( )

З символів алфавіту формуються ключові слова і ідентифікатори.

Ключові слова – це зарезервовані слова, які використовуються в програмі тільки по своєму прямому призначенню.

Ідентифікатори (імена) використовуються для позначення констант, типів, змінних, процедур, функцій і т.д. Ідентифікатори складаються з букв, цифр, знаку підкреслення і повинні починатися з букви. Довжина ідентифікатора може бути довільною. Спеціальні символи не можуть входити до складу ідентифікаторів, а також зарезервовані слова не можна використовувати як ідентифікатори.

Приклади допустимих імен:

X Y Sum P1_1 myprogram

Приклади неприпустимих імен:

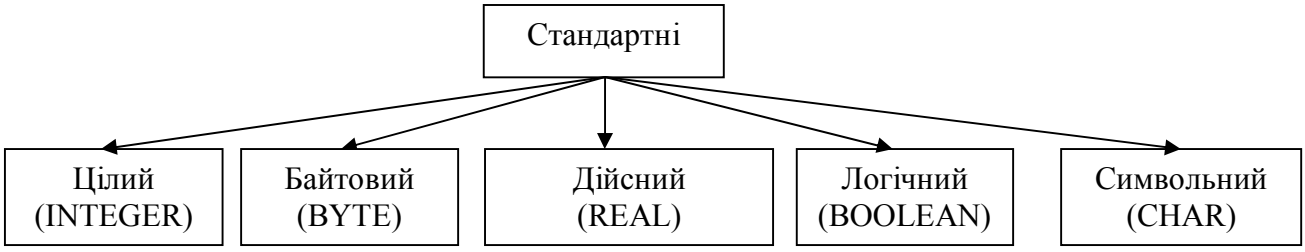
5th	починається з цифри
Type	зарезервоване слово
Sum.6	містить спеціальний символ
X 5	містить пропуск
Корінь	містить російські букви

У тексті програми можна використовувати коментарі. **Коментар** – це будь-яка послідовність символів, ув’язнена у фігурні дужки: { це коментар}.

Замість фігурних дужок можна використовувати пари символів (* і *). Коментар може займати будь-яке число рядків і під час виконання програми ігнорується. Основне призначення коментаря – пояснення по тексту програми.

2.3 Типи даних

Програми обробляють дані, які можуть приймати числові, логічні і буквені значення. Дані представляються у вигляді констант і змінних. **Константа** – це область пам'яті, яка має незмінне значення. **Змінною** називається ім'я, що визначає область пам'яті для зберігання величини, яка може змінюватися під час роботи програми. Всі типи даних можна розділити на прості і складні. Прості – це стандартні і змінні типи даних.



Цілий тип призначений для представлення цілих чисел в діапазоні від -32767 до 32767.

Байтовий тип призначений для представлення цілих чисел з інтервалу від 0 до 255.

Дійсний тип призначений для представлення дійсних чисел, тобто чисел, що мають дробову частину, нуль і відповідні від'ємні числа. Дійсні числа мають дві форми запису:

- з фіксованою крапкою m.n,

де m – ціла частина числа, n – дробова частина числа.

Наприклад, 27.3 5.0 -16.003

- з плаваючою комою mEp,

де m – мантиса числа, E – ознака запису числа з десятковим порядком, p – порядок числа.

Наприклад,

математичний запис	запис на мові Pascal
$4 \cdot 10^{-5}$	4E-05
$0{,}62 \cdot 10^{4}$	0.62E+04
$-10{,}88 \cdot 10^{12}$	-10.88E+12

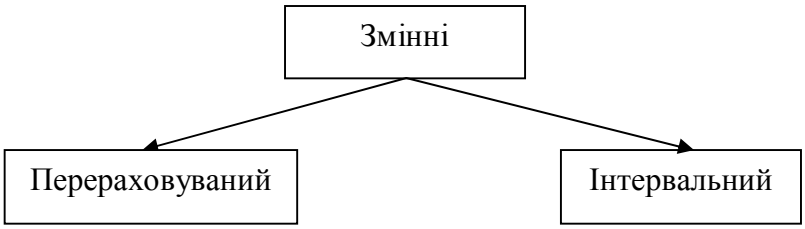
Логічний тип: дані цього типу можуть приймати одне з двох значень істина (TRUE) або лож (FALSE).

Наприклад,

5 > 1.25 результат - true 10 > 25.33 результат - false

Символьний тип: дані символьного типу призначені для зберігання одного символу. У тексті програми значення змінних і констант символьного типу повинні бути розташовані в апострофах: 't' '*' '5'.

Змінні типи даних визначаються користувачем.



Перераховуваний тип задається перерахуванням тих значень, які він може отримувати. Наприклад, A: (red, white, blue)- це означає, що змінна A може приймати тільки одне з цих трьох значень.

Інтервальний тип задається двома константами, що визначають межі діапазону для змінної. Наприклад, M: 1..12 (змінна M може приймати значення від 1 до 12). Накладати обмеження на дійсний тип не можна.

Складні – це різні комбінації простих типів (масиви, множини, записи, файли). Введення складних типів даних робить мову програмування могутнішою і дозволяє складати програми широкого класу завдань.

У математиці, економіці, інформатиці часто використовуються впорядковані набори даних, наприклад, послідовності чисел, таблиці, списки прізвищ. Для обробки наборів даних одного типу вводиться поняття масиву.

Масив – це група однотипних змінних, які об'єднані одним загальним ім'ям. Доступ до заданого елементу масиву здійснюється за допомогою індексу.

Під **множиною** розуміють обмежений, неврегульований набір різних елементів однакового типу. Всій множині в цілому дається ім'я.

У багатьох інформаційних завданнях обробляються відомості, документи, списки. При цьому з'являється необхідність об'єднати дані різного типу в одну групу. Для роботи з групою даних введено поняття запису. **Запис** це сукупність обмеженого числа даних різного типу.

Під **файлом** розуміють будь-який набір даних. Так, початкові дані в програмі можна вважати файлом. Файлом можуть бути і результати, і сама програма.

Перед використанням змінної або константи в програмі їх необхідно спочатку оголосити, тобто задати їх тип і область використання.
Оголошення змінної здійснюється за допомогою службового слова Var.

Var ім'я змінної: тип змінної;

Наприклад,

```
Var a, b,c: integer;          // наприклад, 0   12    -6
    S: real;                  // наприклад,  1.2    1.8E-03
    K: char;                  //  наприклад,  'ABC'  '1234'
```

Оголошення константи здійснюється за допомогою службового слова Const.

Const ім'я константи = значення;

Наприклад,

```
Const pi=3.141492;    //константа дійсного типу з фіксованою
                      крапкою
      K=100;          // константа цілого типу
      Abc='p';        // константа символьного типу
```

2.4 Операції і стандартні функції мови Pascal. Вирази

Для обчислення значень використовуються операції і стандартні функції (див. табл. 2.1). Операції визначають, які дії слід виконати над даними. Операції застосовуються до операндів. Наприклад, операція «+» виконує операцію складання двох чисел або виразів, що є операндами.

Всі операції можна розділити на декілька груп:

1) **арифметичні операції** використовуються для виконання основних арифметичних операцій над даними:

* множення

/ ділення

+ складання

- віднімання

div цілочисельне ділення двох чисел

mod визначення залишку при діленні двох чисел

Пояснення операцій: ділення, div і mod

Вираз	Результат
8/4	2,0
8 div 4	2
8 mod 4	0
6/4	1,5
6 div 4	1
6 mod 4	2

2) **логічні операції** призначені для виконання різних логічних операцій:

or логічне складання (або)

and логічне множення (і)

not логічне заперечення (ні)

До логічних зазвичай відносяться також дві операції зрушення над цілими числами (i та j - вирази цілого типу):

i shl j - зрушення змісту i на j розрядів вліво; молодші розряди, що звільнилися, заповнюються нулями;

i shr j - зрушення змісту i на j розрядів управо; старші розряди, що звільнилися, заповнюються нулями.

Таблиця 2.1 – Стандартні функції мови Pascal

Функція	Назва	Тип аргументу	Тип результату
abs(x)	абсолютне значення	ціле або дійсне	співпадає з типом аргументу
sin(x)	синус	ціле або дійсне	дійсний
cos(x)	косинус	ціле або дійсне	дійсний
exp(x)	експонента	ціле або дійсне	дійсний
ln(x)	натуральний логарифм	ціле або дійсне	дійсний
arctan(x)	арктангенс	ціле або дійсне	дійсний
sqr(x)	квадрат аргументу	ціле або дійсне	співпадає з типом аргументу
sqrt(x)	квадратний корінь	ціле або дійсне	дійсний
trunc(x)	виділення цілої частини дійсного числа	ціле або дійсне	цілий
round(x)	округлення числа	ціле або дійсне	цілий
pi	число 3,14159	аргумент відсутній	дійсний
odd(x)	непарність числа, якщо число непарне – true, парне - false	цілий	логічний
pred(x)	визначає попереднє значення x	цілий логічний символьний	цілий логічний символьний
succ(x)	визначає наступне значення x	цілий логічний символьний	цілий логічний символьний
ord(x)	визначає порядковий номер символу x в наборі символів	логічний символьний	цілий
chr(i)	визначає символ з набору символів по порядковому номеру i	цілий	символьний
frac(x)	дробова частина аргументу: x - int(x)	цілий	дійсний
int(x)	ціла частина аргументу	цілий	цілий

3) операції порівняння

призначені для виконання операцій порівняння логічних виразів:

= <> < > <= >=

Вирази визначають послідовність обчислення значення. Вирази можуть включати константи, змінні, стандартні функції, які розділяються дужками і знаками операцій.

Щоб уникнути помилок при написанні виразу рекомендується виконувати привласнення тільки між сумісними типами даних. Якщо вираз містить декілька операцій, то значення компонентів виразу розраховуються в певному порядку, який називається пріоритетом операцій (табл.2.2). Якщо вираз містить операції різних типів, то першими виконуються арифметичні операції, потім операції порівняння і в останню чергу логічні операції.

Таблиця 2.2 - Пріоритет операцій

Операції	Пріоритет
not	1 (вищий)
* / div mod and shl shr	2
+ - or	3
= <> < > <= >=	4 (нижчий)

Наприклад, запис виразу на мові програмування Pascal

1) $\ln\left|arctg(x)-\sin(ax)\right|+\sqrt[3]{ax}$

має вигляд: ln(abs(arctan(x) -sin(a*x))) + exp (1/3*ln(a*x))

2) $\sin^2 z+\sqrt{|x-y|}$ має вигляд: sqr(sin(z)) + sqrt (abs(x-y))

3) $\dfrac{\sin a-b}{|b|+tgb}$ має вигляд: (sin(a) -b) / (abs(b)+sin(b)/cos(b))

4) $\sqrt[3]{e^{x-1/\sin x}}$ має вигляд: exp((x-1/sin(x))/3)

2.5 Структура програми

Програма на мові Pascal складається із заголовка, розділу описів і розділу операторів:

```
Program ім'я;
розділ описів
begin
розділ операторів
end.
```

Заголовок програми починається з службового слова **Program**, після указується ім'я програми. Закінчується заголовок символом «;» (крапка з комою), наприклад, Program p1_1;

Розділ описів призначений для оголошення всіх в програмі даних, які зустрічаються, і їх характеристик. Існує певний порядок в цьому розділі:

- розділ міток;
- розділ констант;
- розділ типів;
- розділ змінних ;
- розділ процедур і функцій.

Не всі перераховані розділи обов'язкові в кожній програмі. У простих програмах може бути потрібно, наприклад, тільки розділи оголошення констант і змінних. Після кожного розділу опису ставиться символ «;» (крапка з комою).

Розділ міток починається з службового слова **Label**, потім слідує список міток. Мітка – це ідентифікатор або ціле число без знаку.

Label ім'я мітки;

Наприклад, Label rb, 32;

Розділ констант починається з службового слова **Const**

Const ім'я константи = значення;

Наприклад, Const P = 2.26;

Розділ опису типів починається з службового слова **Type**

Type ім'я типу = значення типу;

Наприклад, Type answer = (No, Yes);

Розділ опису змінних починається з службового слова **Var**

Var ім'я змінної : тип;

Наприклад, Var x: real;

Розділ опису процедур і функцій мають структуру, подібну до основної програми.

Розділ операторів береться в операторні дужки вигляду: Begin (почати) і End (закінчити), при цьому після End ставиться крапка. Крапка – ознака кінця програми. У розділі операторів записується послідовність виконуваних операторів. Оператори відділяються один від одного символом «;».

```
begin
оператор 1;
оператор 2;
.....
оператор n
end.
```

2.6 Оператори

2.6.1 Оператор привласнення

Оператор привласнення – основний оператор будь-якої мови програмування. Він призначений для заміни поточного значення змінній новим значенням.

Форма **оператора привласнення** має вигляд:

Змінна : = вираз;

При виконанні цього оператора значення **Виразу** обчислюється і привласнюється **Змінній**.

Наприклад,

a := b+c;

w := sin(sqr(t))/(s + ln(v));

s := ‘рядок’;

2.6.2 Введення-виведення даних

Для завдання змінним їх числових значень можна використовувати оператор привласнення, наприклад: A:=5; B:=-0.12. Проте в цьому випадку програма стає не універсальною, оскільки виконується тільки при цих значеннях змінних. Для виконання програми при різних значеннях змінних використовується оператор введення READ.

Як тільки під час виконання програми зустрічається оператор READ, програма припиняє свою роботу і чекає, поки на клавіатурі будуть набрані потрібні дані і натиснута клавіша Enter.

Оператор введення має вигляд:

read (елемент1, елемент2, …);

Після натиснення клавіші Enter, введені значення привласнюються змінним, імена яких вказані в операторі READ.

Числові значення повинні бути набрані в одному рядку і розділені пропусками.

Наприклад, змінним X, Y, Z необхідно привласнити значення в процесі виконання програми: X:=5.25, Y:=10, Z:=-0.05. Оператор введення буде мати вигляд: read (x, y, z), а числові значення необхідно ввести наступним чином: 5.25 10 -0.05.

Допускається використання оператора введення

readln (елемент1, елемент2, …);

який спочатку вводить значення змінних, а потім здійснює перехід на новий рядок.

Для виведення даних на екран дисплея використовується оператор виведення WRITE. Форма запису оператора має вигляд:

write (елемент1, елемент2, …);

де елемент1, елемент2 і т.д. є в простому випадку або змінними, або рядком символів, які ув'язнені в апострофи.

Наприклад, оператор

write (‘ Значення B= ‘, B);

виводить на екран дисплея текст

Значення B=

а потім значення змінної B.

Для виведення цілих і дійсних чисел можна указувати формати в операторі WRITE. Формат указується через двокрапку після змінної. Для дійсних чисел формат складається з двох величин. Перша величина позначає загальне поле значення, яке виводиться, друге – поле дробової частини. Загальне поле включає знак числа, кількість цифр в цілій частині, крапку і кількість цифр в дробовій частині. Так, виведення значення Y відповідно формату WRITE(Y:5:2) означає, що на зображення всього значення Y відведено п'ять позицій, з них дві – на дробову частину.

Якщо формат відведений більше, ніж кількість позицій, займаних числом, то перед цілою частиною буде відведено відповідну кількість пропусків, а після дробової частини – відповідна кількість нулів.

Наприклад, оператор виведення для значення Y=1.76 має вигляд:

WRITE(‘Y=’,Y:8:3);

Тоді інформація на екрані буде представлена наступним чином:

Y=_ _ _ 1.760

8 позицій

Для виведення цілих чисел формат дробової частини не указується. Наприклад, необхідно вивести значення цілого числа N=125, то оператор

виведення буде мати вигляд: WRITE(‘N=’,N:3). На зображення числа N відведено 3 позиції.

Якщо формат вказати більш ніж 3, наприклад WRITE(‘N=’,N:5), то перед числом буде два пропуски: N=_ _ 125, а для від’ємного числа – один пропуск: N=_ - 125.

Допускається використання оператора виведення

writeln (елемент1, елемент2, …);

який спочатку виводить значення змінних, а потім здійснює перехід на новий рядок.

Перед введенням даних рекомендується давати пояснюючий текст за допомогою оператора WRITE. Цим самим встановлюється діалог користувача і комп'ютера. Наприклад, для введення значень A, B, C краще вказати:

```
WRITE(‘Ввести значення A, B, C’);
READ(A, B,C);
```

Таким чином, перед введенням числових значень на екрані з'явиться повідомлення: Ввести значення A, B, C
після чого можна вводити значення.

Приклад.

```
Program rabota;
Var x,a,b:real;
Begin
Writeln(‘vvod a’);
Readln(a);
x:=2*a;     b:=x+a;
writeln(‘x=’,x:5:2,’    b=’,b:5:2);
end.
```

Результат роботи програми:

```
vvod a
2.5
x= 5.00   b= 7.50
```

2.6.3 Умовний оператор If

Оператор If призначений для реалізації розгалуженого обчислювального процесу.

Умовний оператор **If ... Then … Else** є простою формою перевірки умов і має наступний синтаксис:

If умова Then оператор 1 Else оператор 2;

Оператор 1 виконується в тому випадку, якщо **Умова** істинна, інакше виконується **Оператор 2**. Умова – це вираз логічного типу.

В умовному операторові допустиме використання складеного оператора. **Складений оператор** – це об'єднання декількох операторів в одну групу. Форма запису даного оператора:

```
If умова Then
    Begin
        оператор 1;
        оператор 2;
        ….
        оператор n
    End
Else
    Begin
        оператор 1;
        оператор 2;
        ….
        оператор n
    End;
```

У цій конструкції службові слова **Begin** і **End** називаються операторними дужками. Складений оператор використовується в тих випадках, коли за правилами мови програмування Pascal дозволяється використовувати один оператор, а програмістові для вирішення завдання необхідно виконати групу операторів.

Оскільки гілка Else є необов'язковою, умовний оператор може бути записаний в короткій формі:

If умова Then оператор;

або

```
If умова Then
   Begin
      оператор 1;
      оператор 2;
      ….
      оператор n
   End;
```

Приклад.

Обчислити значення y використовуючи формулу:

$$y = \begin{cases} \frac{1}{2\sqrt{x}}, & x \geq 1 \\ \frac{1}{3\sqrt[3]{x}}, & 0 < x < 1 \\ \frac{1}{4x}, & y \ інших \ випадках \end{cases}$$

Текст програми рішення задачі має вигляд:

```
Program pr2;
Var x, y: real;
begin
writeln (‘Enter x’);   readln (x);
if  x>=1  then  y:=1/(2*sqrt(x))
else
If  (x>0)  and  (x<1)  then
y:=1/(3*exp(1/3*ln(x)))
else
y:=1/(4*x);
writeln (‘y=’, y:5:2);
end.
```

2.6.4 Оператор безумовного переходу

У мові Pascal всі оператори виконуються послідовно один за іншим в тому порядку, як вони записані. Проте в практиці програмування завдань виникає необхідність порушення послідовності виконання операторів. Для цього призначений оператор переходу. Оператор переходу – це оператор, що передає управління в програмі на інший оператор, перед яким стоїть мітка, що вказана в операторі Goto.

Оператор безумовного переходу має вигляд:

Goto мітка;

Мітка – це довільний ідентифікатор (ім'я) або ціле число без знаку, яка повинна бути оголошена в розділі опису Label.

Оператор переходу слід використовувати у виняткових ситуаціях, оскільки він утрудняє розуміння програми, робить її заплутаною і складною у відладці.

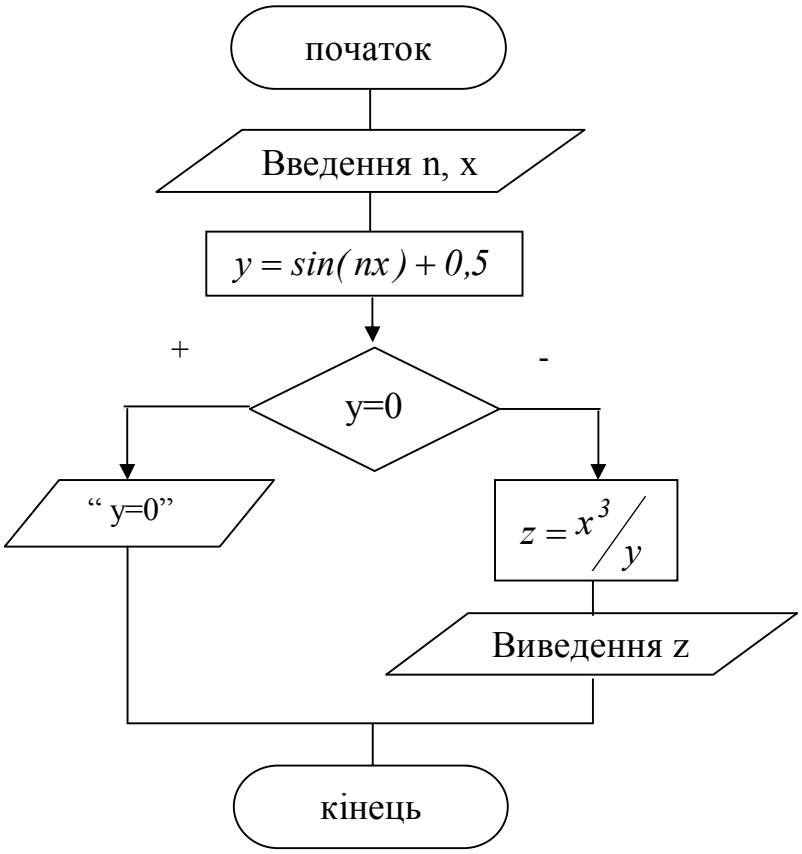
Приклад використання умовного оператора і оператора безумовного переходу:

```
…
If  x < 2  Then
begin    y := a + 2 * x ; Goto 1 end
Else  y := sqr ( x );
b := sin(a)+ abs(x);
writeln (‘b=’, b:5:2);
1:  writeln (‘y=’, y:5:2);
…
```

Приклад.

Обчислити значення $z = \frac{x^3}{y}$, $де$ $y = \sin(nx) + 0{,}5$

Схема алгоритму має наступний вигляд:



Текст програми рішення задачі має вигляд:

1 варіант:

```
Program rabota;
Label 1;
var x, y, n, z: real;
begin
writeln (‘vvod x, n’);  readln (x, n);
y:=sin(n*x)+0.5;
if  y=0 then
begin writeln (‘y=0’);  goto 1 end
else z:=sqr(x)*x/y;
writeln(‘z=’,z:5:2);
1: end.
```

2 варіант:

```
Program rabota;
var x, y, n, z: real;
begin
writeln (‘vvod x, n’);  readln (x, n);
y:=sin(n*x)+0.5;
if  y=0 then  writeln (‘y=0’)
else
begin
z:=sqr(x)*x/y; writeln(‘z=’,z:5:2)
end
end.
```

.

2.6.5 Оператори циклу

Циклічна структура програми дозволяє проводити багатократні обчислення групи операторів при зміні одного або декількох параметрів одночасно.

Розрізняють цикли з відомим числом повторень, коли значення параметра циклу змінюється від деякого початкового до деякого кінцевого значення з постійним кроком; а також цикли з невідомим числом повторень, в яких умова повторення або закінчення циклу задається по деякому проміжному результату, наприклад, поки не буде досягнута необхідна точність обчислень.

У мові Pascal є оператори циклу REPEAT, WHILE та FOR.

Оператор циклу з постумовою REPEAT використовується для організації циклу з невідомим числом повторень.

Оператор циклу Repeat має вигляд:

Repeat
оператор 1;
оператор 2;
……
оператор N
Until логічний вираз;

Оператор циклу виконується наступним чином. Спочатку виконується група операторів циклу, потім обчислюється значення логічного виразу. Якщо значення виразу має дійсне значення,то цикл припиняється, інакше виконується знов група операторів циклу.

Оператор циклу з передумовою WHILE використовується для організації циклу з невідомим числом повторень.

Оператор циклу While має вигляд:

While логічний вираз do оператор;

Оператор циклу виконується наступним чином. Обчислюється значення логічного виразу. Якщо значення логічного виразу має дійсне значення, то виконується оператор, що стоїть після слова ключового слова do. Потім управління знов передається на початок оператора While, знову обчислюється логічний вираз і процес повторюється. Цикл припиняється якщо значення логічного виразу дорівнює false.

Якщо в циклі необхідно виконати декілька операторів, то оператор циклу While буде мати наступний вигляд:

While логічний вираз do
begin
оператор 1;
оператор 2;
……
оператор n
end;

Оператор циклу FOR використовується для організації циклу з відомим числом повторень.

Існує два види запису оператора:

- при збільшенні значення параметра циклу

For i:= n1 to n2 do оператор;

- при зменшенні значення параметра циклу

For i:= n2 downto n1 do оператор;

де

i - параметр циклу;

n1 – початкове значення параметра циклу;

n2 – кінцеве значення параметра циклу.

Оператор циклу виконується наступним чином. Спочатку обчислюється початкове і кінцеве значення параметра циклу. Параметру привласнюється початкове значення. Далі значення параметра порівнюється з кінцевим значенням. Поки параметр менше або рівний кінцевому значенню

(перший варіант) або більше або рівний кінцевому значенню (другий варіант), виконується оператор. Інакше відбувається вихід з циклу. Після виконання оператора циклу, параметру циклу привласнюється наступне значення.

Якщо в циклі необхідно виконати групу операторів, то необхідно використовувати складений оператор:

```
For i := n1 to n2 do
begin
оператор 1;
……
оператор n
end;
```

2.7 Обробка та формування масивів

Масив – це структурований тип даних, що складається з фіксованого числа елементів. Масив позначається одним ім'ям. Елементами масиву можуть бути дані будь-якого типу. Так, сукупність дійсних чисел

1.6 14.9 -5.2 0.45

можна вважати масивом і позначити одним ім'ям, наприклад А:

A = (1.6 14.9 -5.2 0.45)

Всі елементи масиву пронумеровані. Номер елементу масиву називається ***індексом***. Доступ до кожного елементу масиву здійснюється за допомогою індексів. Кожен елемент масиву позначається ім'ям масиву з індексом, наприклад A[1]=1.6; A[2]=14.9. Масив, що є простим списком даних одного і того ж типу, називають ***одномірним масивом***.

Якщо виникає необхідність зберігання даних у вигляді таблиць, у форматі рядків і стовпців, необхідно використовувати багатовимірні масиви. Особливо широкого поширення набули ***двомірні масиви***, які мають назву - матриця.

Наприклад, зображення цілих чисел послідовне в декількох рядках є матрицею

$$B = \begin{pmatrix} 2.3 & -2.7 & 7.4 \\ -0.4 & 3.2 & -4.9 \\ 12.7 & -1.4 & 0.8 \end{pmatrix}$$

Для доступу до даних, що зберігаються в цьому масиві, необхідно вказати ім'я масиву і два індекси, перший відповідає номеру рядка, а другий – номеру стовпця в якому зберігається необхідний елемент.

Наприклад, B[1,2]=-2.7; B[3,1]=12.7.

Якщо в програмі використовується масив, то він повинен бути описаний або в розділі змінних VAR, або в розділі типів TYPE.

1 варіант

Var ім'я масиву: array [тип індексу] of тип елементів масиву;

Наприклад,

```
Var  A: array [1..4] of  real;        //A – це одномірний масив А, що
                                      складається з 4 дійсних чисел.
     B:array[1..3,1..3] of real;   // B - це двомірний масив розміром 3
                                      на 3, що складається з дійсних чисел.
```

2 варіант

Type ім'я типу = array [тип індексу] of тип елементів масиву;

Var ім'я масиву: ім'я типу;

Наприклад,

```
Type mas1= array [1..4] of real;
     mas2= array[1..3,1..3] of real;
Var A: mas1;
    B: mas2;
```

Для введення і виведення числових значень масиву використовуються цикли.

Введення і виведення елементів одномірного масиву A(n)

Введення елементів масиву	*Виведення елементів масиву*
Write(‘Vvod n’); Readln(n); for i := 1 to n do read(a[i]);	for i := 1 to n do write(a[i]);

Введення і виведення елементів двомірного масиву B(n,m)

Введення елементів масиву	*Виведення елементів масиву*

Write(‘Vvod n,m’);
Readln(n,m);
for i := 1 to n do
for j := 1 to m do read(b[i,j]);
	for i := 1 to n do
	for j := 1 to m do
	write(b[i,j]);

Приклад.

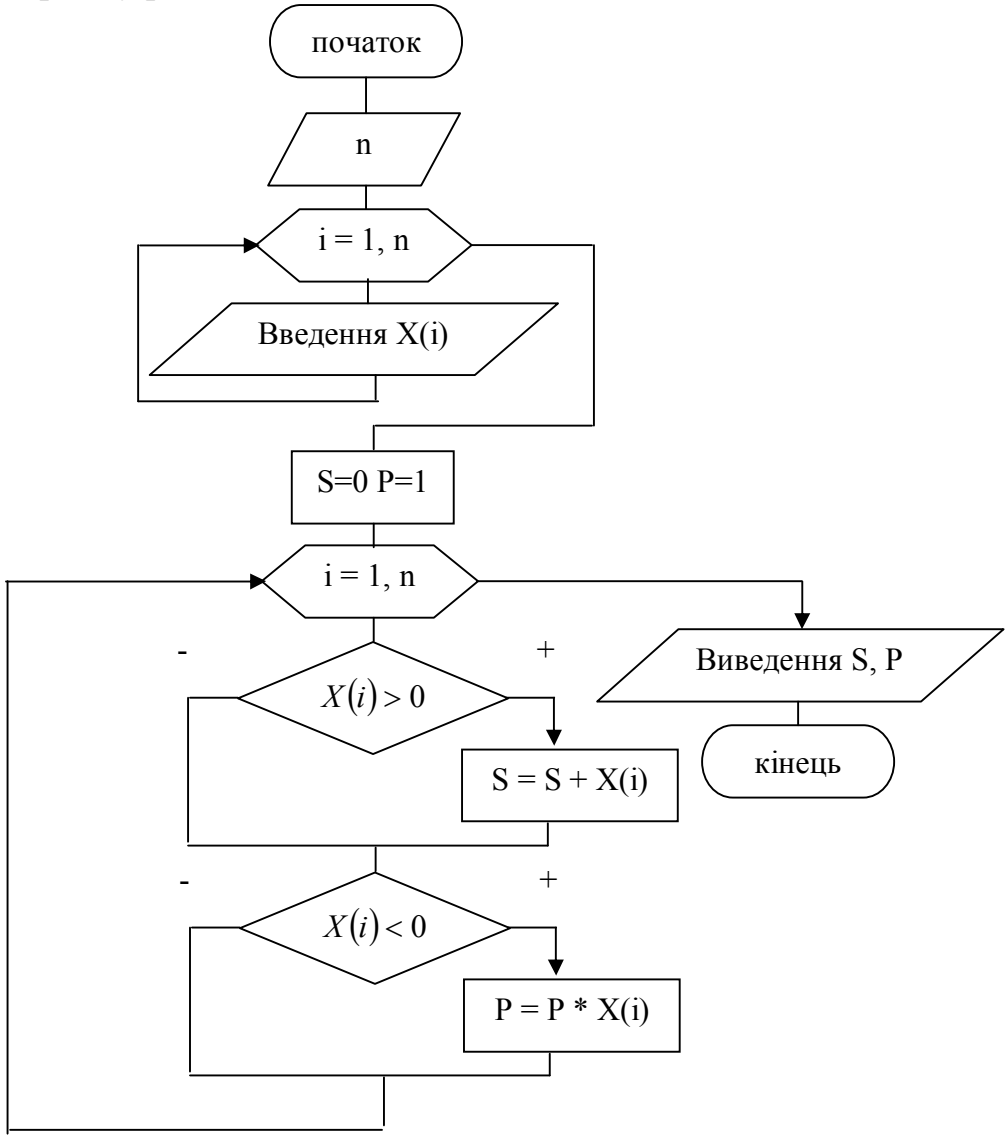
Визначити суму додатних і добуток від’ємних елементів масиву X(N).

Початкові дані: масив X(N).

Текст програми рішення задачі має вигляд:

```
Program pr1;
var x: array [1 . . 10] of real;  i,n: integer;  s,p: real;
begin
writeln(‘Enter n’);  readln (n);
writeln(‘ vvod  massiva ’);
for i := 1 to n do  read (x [i] );
s := 0; p := 1;
for i := 1 to n do
      begin
      if x [i] > 0 then s := s + x [i];
      if x [i] < 0 then p := p * x [i]
      end;
writeln(‘S =’, S : 5 : 3, ’   P=’, P : 5 : 3);
end.
```

Схема алгоритму рішення задачі має вигляд:



Результат роботи програми:

```
enter n
10
vvod massiva
2 4 -3 -0.6 3.1 6.8 0.3 -5.9 -4.1 0.2
s=16.400  p=43.542
```

2.8 Загальні відомості про підпрограми

Частина програми, що реалізовує певний алгоритм і що допускає звернення до неї з різних частин основної програми, називається підпрограмою. Підпрограми оформляються у вигляді замкнутих ділянок програми, що мають чітко позначений вхід і вихід.

У мові програмування Pascal виділяють два види підпрограм: процедуру (PROCEDURE) і функцію (FUNCTION).

Будь-яка програма може містити декілька процедур і функцій. Процедури і функції оголошуються в розділі опису услід за розділом змінних. Приклад структури програми з двома підпрограмами.

```
Program  ім'я;
Label ..;
Const...;
Type....;
Var...;
Підпрограма P1;
Підпрограма P2;
Begin
Оператори;
Виклик підпрограми P1;
Оператори;
Виклик підпрограми P1;
Оператори;
Виклик підпрограми P2;
Оператори;
Виклик підпрограми P2;
Оператори;
End.
```

Виконання програми починається з операторів основної програми. При необхідності викликається підпрограма, і починають виконуватися її оператори. Потім управління передається в основну програму, яка продовжує виконуватися. Підпрограма оформляється аналогічно основній програмі, тобто складається із заголовка, розділу описів і розділу операторів.

2.8.1 Процедури

Структура процедури має вигляд:

```
Procedure ім'я (список формальних параметрів);
Розділ описів процедури
Begin
Оператор 1;
Оператор 2;
...
Оператор n
End;
```

У заголовку указується службове слово Procedure, за яким слідує ім'я процедури і формальні параметри в круглих дужках. Розділ описів процедури подібний до програми і складається з розділів міток, констант, типів, змінних і, у свою чергу, процедур і функцій. Розділ операторів розташовується в операторних дужках Begin… End.

Процедура викликається по її імені:

Ім'я (список фактичних параметрів);

За допомогою формальних і фактичних параметрів дані передаються з програми в процедуру і, навпаки, з процедури в програму. Кожен формальний параметр указується разом з своїм типом. Відповідний йому фактичний параметр указується без типу. Між формальними і фактичними параметрами повинна бути відповідність по кількості параметрів, за їх типом і порядком слідування.

Якщо декілька формальних параметрів мають однаковий тип, тоді їх можна об'єднати в список, тобто перерахувати параметри через кому, а потім вказати тип. Наприклад, заголовок процедури має вигляд:

Procedure Summa (a,b,c:integer); де a,b,c – формальні параметри.

Викликати процедуру можна таким чином:

Summa (5, m, 7); де 5, m, 7 – фактичні параметри.

Кожен формальний параметр приймає значення відповідного фактичного параметра. Таким чином, в процедурі матимемо наступні значення: a=5; b= m; c=7.

Серед параметрів процедури виділяють параметри-значення і параметри-змінні. Параметри-значення виконують роль вхідних параметрів процедури. Вони можуть приймати значення фактичних параметрів, але не можуть передавати свої значення фактичним параметрам. Параметри-змінні виконують роль як вхідних, так і вихідних параметрів процедури. Вони можуть приймати значення фактичних параметрів, змінювати їх в процедурі і повертати нові значення фактичним параметрам. Для виділення параметрів-змінних перед ними ставиться слово Var у формальних параметрах.

Приклад процедури.

```
Procedure calc (a,b,c: real; Var x,y: real);
Var z: real;
Begin
z:= a+b+c;   x:=sqr(z);  y:=sqrt(z)
End;
```

Цю процедуру можна викликати таким чином: calc (25, 44, 30, x1, y1). Результатом виконання процедури є X,Y, які передають свої значення відповідним фактичним параметрам X1,Y1.

Якщо в процедуру потрібно передати як параметр не просто одне значення, а масив, то фактичним параметром повинне бути ім'я масиву. При цьому формальний параметр указується після слова Var разом з типом масиву. Сам же опис масиву робиться в розділі Type основної програми.

Наприклад, в основній програмі є опис масиву A:

```
Const n=10;
Type mas=array[1..n] of real;
Var a:mas;
```

Тоді формальні параметри процедури Prim можуть мати вигляд: Procedure Prim (k:integer; Var x:mas) при наступних фактичних параметрах: Prim (n, a).

Формальний параметр k приймає значення фактичного параметра n, а формальний параметр x – значення масиву a.

Приклад 1.

```
Program primer1;
 Var  a, b, c, d : real;        S1, S2, S3 : real;
  Procedure lab (x, y, f : real; Var S : real);
     Var  k : real;
      Begin  k := x + y / 2;   S := 2 * k + f     End;
Begin
    Writeln (‘a, b, c, d’);  Readln (a, b, c, d);
```

```
    Lab (a, b, c, S1);   Lab (b, d, a, S2);
    S3 := S1 + S2;   Writeln('S3 =',  S3:5:2);
End.
```

Допускається використання процедур без параметрів. В цьому випадку формальні і фактичні параметри відсутні. В цьому випадку передача початкових даних в процедуру і вихідних даних з процедури в основну програму здійснюється за допомогою глобальних параметрів. А саме, початкові дані і вихідні дані процедури описуються як глобальні в розділі змінних основної програми. Оператор виклику процедури в цьому випадку складатиметься тільки з імені процедури без фактичних параметрів.

Приклад 2.

```
Program primer2;
    Var a, b, c, d, x, y, f, S, S1, S2, S3 : real;
    Procedure lab;
        Var k : real;
        Begin   k := x + y / 2;    S := 2 * k + f   End;
Begin
        Writeln ('a, b, c, d');    Readln (a, b, c, d);
        x := a;  y := b;  f := c;   Lab;  S1 := S;
        x := b;  y := d;  f := a;   Lab;  S2 := S;
        S3 := S1 + S2; Writeln ('S3 =',  S3:5:2);
End.
```

2.8.2 Функції

Структура функції має вигляд:

```
Function  ім'я (список формальних параметрів) : тип результату;
Розділ описів функції
Begin
Оператор 1;
...
Оператор n;
Ім'я := результат
End;
```

У функції формальні параметри підкоряються тим же правилам, що і в процедурі. Результатом функції є одна проста змінна, значення якої привласнюється в тілі функції імені функції.

Викликається функція по її імені з вказівкою фактичних параметрів. Виклик функції можна робити безпосередньо усередині виразу.

Приклад.

```
Program primer;
Var a, b, c, d, S : real;
Function  lab (x, y, f : real) : real;
   Var k : real;
    Begin
    k := x + y / 2;
    lab := 2 * k + f
    End;
Begin
    Writeln (‘a, b, c, d’);       Readln (a, b, c, d);
     S := Lab (a, b, c) + Lab (b, d, a);
     Writeln(‘S =’, S:5:2);
End.
```

2.8.3 Приклад програмування завдання з використанням підпрограм

Задані масиви B(i), H(j), U(k), S(n), де i=1 – 6; j=1 – 8; k= 1 – 9; n= 1 – 7. Скласти підпрограми :

- розрахунку значень елементів масивів:

$B(i) = \sin(i + 0.5) + \cos(i - 2.9)$ $\qquad$ $H(j) = \sin(j - 1.4) + \cos(j + 3.7)$

$U(k) = \sin(k + 2.2) + \cos(k + 7.8)$ $\qquad$ $S(n) = \sin(n - 4.3) + \cos(n + 9.2)$

- визначення різниці максимального і мінімального елементів масивів.

Рішення.

1) Елементи масивів B, H, U, S обчислюються по загальній формулі: $Y(f) = \sin(f + A) + \cos(f + C)$, де f змінюється від 1 до M.

Для кожного масиву можна розрахувати елементи, якщо підставити відповідні коефіцієнти в цю формулу. Наприклад, для масиву B індекс f буде змінюється від 1 до 6, тобто M=6, а коефіцієнти A=0.5 і C=-2.9.

Обчислення елементів масиву Y(M) можна оформити у вигляді окремої процедури з ім'ям Dan. Початковими даними для процедури будуть параметри A, C дійсного типу і розмірність масиву M цілого типу. Результатом виконання процедури Dan буде масив Y, тип якого описується в розділі типів основної програми. Індекс масиву f буде допоміжною змінною в процедурі при організації циклу, тому описується як локальна змінна в розділі описів процедури. Процедура Dan матиме вигляд:

```
Procedure Dan (A,C: real; M: integer; Var Y: mas);
Var f: integer;
Begin
    For f:=1 to M do Y[f]:=sin(f+A)+cos(f+C);
End;
```

Ця процедура використовуватиметься чотири рази для розрахунку елементів даних масивів.

2) Визначення різниці максимального і мінімального елементів масиву Y(M) можна оформити у вигляді окремої процедури з ім'ям Ras. Початковими даними для процедури Ras буде розмірність масиву M цілого типу і масив Y типу mas. Результатом виконання процедури Ras буде змінна R дійсного типу (різниця між максимальним і мінімальним елементом масиву). Індекс масиву f, змінна max (значення максимального елементу) і змінна min (значення мінімального елементу) будуть допоміжними в процедурі і описуються як локальні змінні в розділі описів процедури. Процедура Ras матиме вигляд:

```
Procedure Ras (M: integer; Y: mas; Var R: real);
Var f: integer; max, min: real;
Begin
        max:=Y[1]; min:=Y[1];
```

```
        For f:=2 to M do
        If Y[f]>max then max:=Y[f]  Else  If Y[f]< min then min:=Y[f];
        R:=max-min;
    End;
```

3) Для виведення елементів масиву Y(M) можна оформити процедуру Vuvod. Початковими даними для процедури будуть розмірність масиву M цілого типу, масив Y типу mas, символьна змінна Z для вказівки імені масиву при друці. Результатом виконання процедури Vuvod буде друк елементів масиву Y. Процедура Vuvod матиме вигляд:

```
    Procedure Vuvod (M: integer; Y: mas; Z: char);
    Var f: integer;
    Begin
        Writeln('Масив  ', Z);
        For f:=1 to M do  Write(Y[f]:5:3, '  ');
        Writeln;
    End;
```

4) У основній програмі приймаються наступні позначення:

NB, NH, NU, NS – розмірності масивів B, H, U, S відповідно;

RB, RH, RU, RS – різниці між максимальним і мінімальним елементів масивів B, H, U, S відповідно.

Програма матиме вигляд:

```
Program primer1;
Type mas=array[1..50] of real;
Var  B, H, U, S: mas;
NB, NH, NU, NS: integer; RB, RH, RU, RS: real;
(* Процедура Dan *)
Procedure Dan (A,C: real; M: integer; Var Y: mas);
Var f: integer;
Begin
    For f:=1 to M do Y[f]:=sin(f+A)+cos(f+C);
End;
```

```
(* Процедура Ras *)
Procedure Ras (M: integer; Y: mas; Var R: real);
Var f: integer; max, min: real;
Begin
        max:=Y[1]; min:=Y[1];
        For f:=2 to M do
        If Y[f]>max then max:=Y[f] Else If Y[f]< min then min:=Y[f];
        R:=max-min;
End;

(* Процедура Vuvod *)
Procedure Vuvod (M: integer; Y: mas; Z: char);
Var f: integer;
Begin
        Writeln('Масив  ', Z);
        For f:=1 to M do  Write(Y[f]:5:3, '   ');
        Writeln;
End;

(* Основна програма *)
Begin
Write ('Ввести розмірність масиву'); Readln(NB);
Dan (0.5,-2.9,NB, B); Ras (NB, B, RB); Vuvod (NB, B, 'B');
Write ('Ввести розмірність масиву H'); Readln(NH);
Dan (-1.4,3.7,NH, H); Ras (NH, H, RH); Vuvod (NH, H, 'H');
Write ('Ввести розмірність масиву U'); Readln(NU);
Dan (2.2,7.8,NU, U); Ras (NU, U, RU); Vuvod (NU, U, 'U');
Write ('Ввести розмірність масиву S'); Readln(NS);
Dan (-4.3,9.2,NS, S); Ras (NS, S, RS); Vuvod (NS, S, 'S');
Writeln ('RB=',RB:5:3 '  RH=',RH:5:3 '  RU=',RU:5:3 '   RS=',RS:5:3);
End.
```

У даному прикладі можна процедуру Ras оформити процедурою без параметрів. В цьому випадку початкові дані розмірність масиву M і масив Y,

а також вихідна змінна R повинні оголошуватися в основній програмі як глобальні змінні.

Програма в цьому випадку матиме вигляд:

```
Program primer2;
Type mas=array[1..50] of real;
Var  B, H, U, S, Y: mas;
NB, NH, NU, NS, M: integer; RB, RH, RU, RS, R: real;

(* Процедура Dan *)
Procedure Dan (A,C: real; M: integer; Var Y: mas);
Var f: integer;
Begin
        For f:=1 to M do Y[f]:=sin(f+A)+cos(f+C);
End;

(* Процедура Ras *)
Procedure Ras;
Var f: integer; max, min: real;
Begin
        max:=Y[1]; min:=Y[1];
        For f:=2 to M do
        If Y[f]>max then max:=Y[f] Else If Y[f]< min then min:=Y[f];
        R:=max-min;
End;

(* Процедура Vuvod *)
Procedure Vuvod (M: integer; Y: mas; Z: char);
Var f: integer;
Begin
        Writeln('Масив  ', Z);
        For f:=1 to M do  Write(Y[f]:5:3, '   ');
        Writeln;
End;
```

```
(* Основна програма *)
Begin
Write (‘Ввести розмірність масиву’); Readln(NB);
Dan (0.5,-2.9,NB, B); Vuvod (NB, B, ‘B’); M:=NB; Y:=B; Ras; RB:=R;
Write (‘Ввести розмірність масиву H’); Readln(NH);
Dan (-1.4,3.7,NH, H); Vuvod (NH, H, ‘H’); M:=NH; Y:=H; Ras; RH:=R;
Write (‘Ввести розмірність масиву U’); Readln(NU);
Dan (2.2,7.8,NU, U); Vuvod (NU, U, ‘U’); M:=NU; Y:=U; Ras; RU:=R;
Write (‘Ввести розмірність масиву S’); Readln(NS);
Dan (-4.3,9.2,NS, S); Vuvod (NS, S, ‘S’); M:=NS; Y:=S; Ras; RS:=R;
Writeln (‘RB=’,RB:5:3 ‘  RH=’,RH:5:3 ‘  RU=’,RU:5:3 ‘  RS=’,RS:5:3);
End.
```

Програма Primer2 є громіздкішою в порівнянні з програмою Primer1. Чим більше число початкових і вихідних даних процедури без параметрів, тим більше громіздкою буде основна частина програми.

У даному прикладі можна також використовувати функцію користувача. Процедури Dan і Vuvod представити у вигляді функції не можна, оскільки в них як дані використовується масив Y структурованого типу, а результатом функції може бути тільки проста змінна. Процедуру Ras можна оформити у вигляді функції, оскільки вихідною величиною в ній є проста змінна R дійсного типу. Цей результат можна привласнити імені функції.

Програма в цьому випадку матиме вигляд:

```
Program primer3;
Type mas=array[1..50] of real;
Var  B, H, U, S: mas;
NB, NH, NU, NS: integer; RB, RH, RU, RS: real;

(* Процедура Dan *)
Procedure Dan (A,C: real; M: integer; Var Y: mas);
Var f: integer;
Begin
```

```
        For f:=1 to M do Y[f]:=sin(f+A)+cos(f+C);
End;

(* Функція  Ras *)
Funcrtion  Ras (M: integer; Y: mas): real;
Var f: integer; max, min: real;
Begin
        max:=Y[1]; min:=Y[1];
        For f:=2 to M do
        If Y[f]>max then max:=Y[f] Else If Y[f]< min then min:=Y[f];
        Ras:=max-min;
End;

(* Процедура Vuvod *)
Procedure Vuvod (M: integer; Y: mas; Z: char);
Var f: integer;
Begin
        Writeln('Масив  ', Z);
        For f:=1 to M do  Write(Y[f]:5:3, '   ');
        Writeln;
End;

(* Основна програма *)
Begin
Write ('Ввести розмірність масиву'); Readln(NB);
Dan (0.5,-2.9,NB, B); Vuvod (NB, B, 'B'); RB:=Ras (NB, B);
Write ('Ввести розмірність масиву H'); Readln(NH);
Dan (-1.4,3.7,NH, H); Vuvod (NH, H, 'H'); RH:=Ras (NH, H);
Write ('Ввести розмірність масиву U'); Readln(NU);
Dan (2.2,7.8,NU, U); Vuvod (NU, U, 'U'); RU:=Ras (NU, U);
Write ('Ввести розмірність масиву S'); Readln(NS);
Dan (-4.3,9.2,NS, S); Vuvod (NS, S, 'S'); RS:=Ras (NS, S);
Writeln ('RB=',RB:5:3 '  RH=',RH:5:3 '  RU=',RU:5:3 '   RS=',RS:5:3);
End.
```

3. ЗАВДАННЯ ДЛЯ КОНТРОЛЬНИХ РОБІТ

Завдання 1. Програмування алгоритмів розгалужених процесів

Мета роботи: скласти блок-схему алгоритму і програму обчислення і виведення значень.

№ п/п	Вид завдання	№ п/п	Вид завдання
1	$v=\begin{cases} b^2-\sqrt{x+b}, & якщо\ \ x<2 \\ \dfrac{bx-1}{\ln(x-b)}, & якщо\ \ x\ge 2 \end{cases}$	2	$z=\begin{cases} \sqrt[3]{ax+1}, & якщо\ \ x>1.5 \\ \dfrac{ax+1}{\sin x}, & якщо\ \ x\le 1.5 \end{cases}$
3	$z=\begin{cases} t-\dfrac{1}{\ln(t-a)}, & якщо\ \ t\le 2 \\ \cos^2\dfrac{x}{a}, & якщо\ \ t>2 \end{cases}$	4	$v=\begin{cases} y+\sqrt[3]{ax}, & якщо\ \ x<0 \\ \sin y^2+a, & якщо\ \ x=0 \\ \dfrac{y+x}{\ln(x-y)}, & якщо\ \ x>0 \end{cases}$
5	$z=\begin{cases} \ln\lvert x^3\rvert-b^3, & якщо\ \ x<1 \\ \dfrac{\lvert\sin x\rvert+b}{x-b}, & якщо\ \ x\ge 1 \end{cases}$	6	$s=\begin{cases} \sqrt[5]{ax-1}, & якщо\ \ x<5 \\ \dfrac{2.5}{\ln(x-a)}, & якщо\ \ x\ge 5 \end{cases}$
7	$z=\begin{cases} 1.5a-\ln(x-a), & якщо x\le 2 \\ \dfrac{2.3}{\sqrt{8.5-xa}}, & якщо x>2 \end{cases}$	8	$y=\begin{cases} e^{\frac{1}{x-1}}, & якщо\ \ x<2 \\ \sqrt[3]{x^2-4}, & якщо\ \ x\ge 2 \end{cases}$
9	$y=\begin{cases} \dfrac{t\cdot e^{-x}}{\ln(x+1)}, & якщо\ \ x<1 \\ \sin x+t^2, & якщо\ \ x\ge 1 \end{cases}$	10	$y=\begin{cases} \ln\lvert ax+1\rvert, & якщо\ x<-0.5 \\ \sqrt{ax+1}, & якщо\ -0.5\le x\le 2 \\ ax+1, & якщо\ x>2 \end{cases}$

Завдання 2. Програмування алгоритмів циклу

з невідомим числом повторень

Мета роботи: скласти блок-схему алгоритму і програму обчислення і виведення значень.

№ п/п	Вид завдання	Вид завдання
1	$z=\begin{cases}\sqrt[3]{ax+1}+b, & x<5\\ Sin(bx+2), & x=5\\ b\cdot cos(cx+3), & x>5\end{cases}$ $y=\sqrt{z}+\frac{b\cdot cosz}{ln(zx)}$ $a,b,c;\quad -2\le x\le 4\qquad \Delta x=1$	$y=\begin{cases}e^{Sinx}, & a^2x<b^3\\ (b^2-a)/\sin x, & a^2x=b^3\\ \cos^2 4x, & a^2x>b^3\end{cases}$ $F=5{,}37y+\ln\left(x^3+x^2+x\right)$ Рахувати F до тих пір, поки вираз під знаком логарифма більше 0. Визначити $P=\prod_{F>0}F,\quad S=\sum_{F<0}F$ a, b; $x\le 3$ $\Delta x=-0{,}1$
2	$z=\begin{cases}ax^2+1, & x<3\\ bx+\cos^2 bx, & x=3\\ \frac{1}{\sqrt{ax-bx}}, & x>3\end{cases}$ $y=\sin(za)+\cos(za)$ $a,b;\quad -1\le x\le 8\qquad \Delta x=0.5$	$z=\begin{cases}y^2+\frac{a+y}{ay}, & y\le 1\\ 3ay+\cos^2 ay, & y>1\end{cases}$ $F=z^2\sqrt{0{,}1+y}+\frac{3z}{\sqrt{0{,}1+y}}+b$ Рахувати F до тих пір, поки підкорінний вираз не перевищує значення Q. Визначити кількість z>0 , $S=\sum_{z\le 0}z$ a, b, Q; $y\ge 0$, $\Delta y=0{,}5$
3	$v=\begin{cases}y+\sqrt{0{,}5+\sin x}, & y\le a\\ 3\cdot e^{xy+1}, & y>a\end{cases}$ $y=\sin\left(\frac{\pi}{2}x\right)$ $x=\sqrt{t^2-0.5}$ a; $0\le t\le 10\qquad \Delta t=2$	$z=\begin{cases}\frac{x^2}{x+a}+\sqrt{x}, & x\le 1\\ \sqrt{ax}+3x\sin^3 x, & x>1\end{cases}$ $y=2bz\cdot\sin\lvert\pi a\rvert+\sqrt{x+t}$ Рахувати y до тих пір, поки підкорінний вираз $x+t\ge 0$. Визначити M=K!, де K - кількість обчислених y. a, b, t; $x\le 5$ $\Delta x=-0{,}5$

4	$a=\begin{cases}1+\sin x, & x<3\\ 0.5\cos^2\dfrac{x}{x+b}, & x=3\\ 1/(1+x^2), & x>3\end{cases}$ $y=e^{-x}+b\cdot\ln(ax)$ b; $-2\le x\le 2$ $\Delta x=1$	$z=\begin{cases}a^2+0,2\mid\sin ax\mid, & x\ge 1\\ \sqrt{2x^2-3x}+e^{-ax}, & x<1\end{cases}$ $y=a\cdot\cos x^2+\ln\lvert\sin 3z\rvert$ Рахувати y до тих пір, поки F<Q. Визначити $F=\sum y$, кількість $z\ge 0$. a, Q; $x\ge 1$ $\Delta x=0,2$
5	$y=\begin{cases}1-\sin^3 ax, & x\succ 0\\ -\sin(ax^2), & x=0\\ \dfrac{a+x}{\sqrt{\cos ax}}, & x\prec 0\end{cases}$ $z=a^2+\ln(ay)$ a; $-3\le x\le 5$ $\Delta x=2$	$z=\begin{cases}\sqrt{\dfrac{x}{a+0,3x}}+a^3, & x\le 2\\ \cos(a+0,3x)+5x, & x>2\end{cases}$ $F=\sqrt{\dfrac{0,2x}{x^2+1}}+\sin^3 z^2$ Рахувати F до тих пір, поки підкорінний вираз більше 0. Визначити кількість обчислених F. a; $x\le 3$ $\Delta x=-0,2$
6	$y=\begin{cases}ab\cdot(x+\sin^2 x), & x<-5\\ (a+b)/(1+x), & -5\le x\le 5\\ \ln(ab-x)-x^3, & x>5\end{cases}$ $z=\cos y+a^3/bx$ a,b; $-10\le x\le 25$ $\Delta x=5$	$t=\begin{cases}(a-b)\sqrt{\ln(b+x)}, & x<0\\ \sqrt[3]{ax}+\sin^3 ax, & x\ge 0\end{cases}$ $z=1,5t-\ln\left(\dfrac{x}{ax+b}\right)$ Рахувати Z до тих пір, поки вираз під знаком логарифма більше 0. Визначити кількість обчислених z, $P=\prod t$, $S=\sum_{z>t} z$ a,b; $x\ge 1$ $\Delta x=-0,1$
7	$z=\begin{cases}\ln\lvert x\rvert+\dfrac{2}{\sin bx}, & x<1\\ (1+x^2)\cdot\cos b, & x=1\\ b\cdot e^x, & x>1\end{cases}$ $y=z^3+\sqrt{b\cdot\cos z}$ b; $-3\le x\le 3$ $\Delta x=1$	$y=\begin{cases}(a+b)^3+e^x, & x\ge 2\\ ax^3+\ln(b+x), & x<2\end{cases}$ $z=\dfrac{a+b}{2x-b}\sin(x+y)$ Рахувати Z до тих пір, поки F∈ [-2;2]. Визначити кількість обчислених z, $S=\sum_{y>0} y$, $F=\prod z$ a,b; $x\ge 0$ $\Delta x=0,5$

8	$y=\begin{cases}\sqrt{a+bx}, & x<4\\ a+\ln bx, & 4\le x\le 6\\ (a-b)\cdot\sin x, & x>6\end{cases}$ $b=(x^2+a)/\ln x$ $a;\quad -2\le x\le 10\quad \Delta x=1$	$b=\begin{cases}\sqrt{\lvert a-x\rvert}\ln(a+x), & x<2\\ a+\sqrt{x^3}, & x\ge 2\end{cases}$ $y=\sin^2 b+\cos(x-\pi)+1$ Рахувати y до тих пір, поки добуток P менше 10. Визначити $P=\prod y,\quad C=\sum_{b>0} b+\sum_{y<0} y$ a; x≥0 Δx= 0,1
9	$y=\begin{cases}\sqrt{b^3+x}, & x<a^2\\ \lvert x^5\rvert-ax, & x=a^2\\ \dfrac{ax^4+4x^3}{bx}, & x>a^2\end{cases}$ $b=sin\ x+cos(xa)$ $a;\quad -2\le x\le 10\quad \Delta x=0.2$	$y=\begin{cases}0{,}5a\dfrac{x}{1-x}+\sin^2 xa, & x\le 1\\ 4a+e^{-x}+2\cos^3 x^2, & x>1\end{cases}$ $z=2{,}3y+\sqrt{\dfrac{x}{2x^2+1}}$ Рахувати z до тих пір, поки підкорінний вираз додатний. Визначити кількість і суму додатних значень z. a; x≤10 Δx= -0,5
10	$z=\begin{cases}\ln\lvert x\rvert-b^2, & x<0.2\\ \sin x+\sqrt{bx}, & x=0.2\\ b+\ln x, & x>0.2\end{cases}$ $b=(a+\sin x)^2/\cos x$ $a;\quad -8\le x\le 8\quad \Delta x=0.4$	$y=\begin{cases}\sin a^2+\cos(a-\pi)+x^3, & x>0\\ 3\cdot\ln(1+e^x)+\sqrt{\lvert x\rvert}, & x\le 0\end{cases}$ $F=\dfrac{\pi}{2}\sin 2y+\ln(2x-x^2)$ Рахувати F до тих пір, поки вираз під знаком логарифма більше 0. Визначити $P=\prod F,\quad S=\sum_{y<0} y,$ кількість F>0. a; x≤1 Δx= -0,1

Завдання 3. Програмування алгоритмів циклу з відомим числом повторень

Мета роботи: скласти блок-схему алгоритму і програму обчислення і виведення значень.

№ п/п	Вид завдання
1	$y = \begin{cases} 2\sin^2 x + x^3, & x > 0 \\ \dfrac{\lvert x \rvert}{x^2 - 1}, & x \le 0 \end{cases}$ $a = ln \lvert y \rvert + 0{,}5\, y^2$ $-2 \le x \le 5 \quad \Delta x = 0{,}1$ Визначити середнє арифметичне додатних значень *y* і добуток від'ємних значень *a*.
2	$z = \begin{cases} 2x^3 + 3\cos x, & x \ge 5 \\ 7 + \ln(x+6), & 1 \le x < 5 \\ -\dfrac{2}{x^3}, & x < 1 \end{cases}$ $y = (x^2 + 2) \cdot \sin z$ $-2 \le x \le 10 \quad \Delta x = 1$ Визначити кількість z $\in$[-1; 1] і суму z $\notin$[-1; 1].
3	$z = \begin{cases} \sqrt{xy}, & x^2 + y^2 < 1 \\ x + \dfrac{x+y}{x-y}, & x^2 + y^2 = 1 \\ 2x + sin(x+y), & x^2 + y^2 > 1 \end{cases}$ $y = (a+x)^3 + cos\, a^3$ $a; \quad -2 \le x \le 2 \quad \Delta x = 0{,}5$ Визначити суму від'ємних значень z, кількість z > 0 і добуток всіх значень *y*.
4	$y = \begin{cases} ax + \ln x, & x < 0{,}5 \\ \sin^2 x + e^{-x}, & x = 0{,}5 \\ \sqrt{x} + \dfrac{a}{bx}, & x > 0{,}5 \end{cases}$ $a = \sin(xb) + \sqrt{2 - x}$ $b; \quad -1 \le x \le 1 \quad \Delta x = 0{,}2$ Визначити кількість y $\ge$ 0 і добуток a < 0.

5	$f=\begin{cases}-4, & x<-1\\ x^2+\frac{3}{x}+4, & -1\le x\le 1\\ \cos(x+4)^2, & x\succ 1\end{cases}$ $Z=\sin^3 x+\lvert f\rvert$ $-2\le x\le 2\quad \Delta x=0{,}2$ Визначити суму і кількість $f>0$, добуток значень Z.
6	$q=\begin{cases}1+cos2a, & e^{-x}>0{,}1\\ \pi\, sin\frac{a}{2}, & e^{-x}\le 0{,}1\end{cases}$ $a=2x+\sqrt{x^3+4}$ $-0{,}5\le x\le 2\quad \Delta x=0{,}5$ Визначити кількість q>0, суму значень *a* і добуток значень (q - *a*).
7	$z=\begin{cases}cos^2\frac{\pi}{4}x+ln\lvert x\rvert, & y>x^2\\ \frac{y}{(x+1)}, & y\le x^2\end{cases}$ $y=sin^2 x+0{,}5\,cos\,x^2$ $-2\le x\le 2\quad \Delta x=0{,}5$ Визначити суму і кількість значень (y + z) за умови, що y >z.
8	$y=\begin{cases}x^2+\sin^3 x, & a\le x\le b\\ \sqrt[3]{x}-\frac{ab}{\lvert x\rvert}, & c\le x\le d\\ 0, \quad в\quad інших\quad випадках\end{cases}$ $f=xe^y+1$ $a,b,c,d;\quad -1\le x\le 1\quad \Delta x=0{,}1$ Визначити суму значень *f*, кількість і добуток y > 0.
9	$f=\begin{cases}cos^2 x-z, & z>0\\ \sqrt{z+3x}+x^3, & -1\le z\le 0\\ \lvert x+0{,}38z\rvert, & z<-1\end{cases}$ $z=x^3+5\,ln\lvert x\rvert$ $-1\le x\le 5\quad \Delta x=0{,}5$ Визначити добуток значень z, суму значень $f<0$ і кількість $f>$ z.
10	$z=\begin{cases}x+3\sin\frac{\pi}{2}x, & y<0\\ \ln(x-2)+x^3, & y\ge 0\end{cases}$ $y=a^2-a+\sqrt{x}$ $a;\ -1\le x\le 4\quad \Delta x=0{,}5$ Визначити суму перших п'яти значень z і добуток y > 0.

Завдання 4. Програмування алгоритмів із структурою вкладених циклів

Мета роботи: скласти блок-схему алгоритму і програму обчислення і виведення значень.

№ п/п	Вид завдання	Величини, які необхідно вивести
1	$x=\frac{-\lvert a+5\rvert}{c^2}+bc^2 \qquad y=b^2c+\left\lvert\frac{a}{x}\right\rvert+1$ $c,\quad 2\le a\le 6\quad \Delta a=1\quad -4{,}5\le b\le 4{,}5\quad \Delta b=3$	a, b, x, y
2	$x=b^3a-\frac{\lvert ac\rvert}{ac-10} \qquad y=\frac{\lvert x^2+2x\rvert}{b^3-c+5}$ $a,\quad 3\le b\le 9\quad \Delta b=2\quad 1\le c\le 5\quad \Delta c=1$	b,c,x,y
3	$x=ay^2-\left(by^2+c\right) \qquad y=\frac{ab^2-cb}{\lvert ac\rvert}$ $c,\quad 3\le a\le 6\quad \Delta a=1\quad -4{,}5\le b\le 4{,}5\quad \Delta b=3$	a, b, x, y
4	$x=\frac{(a+2)^2}{\lvert bc\rvert}-\frac{abc}{a+2} \qquad y=\frac{\lvert(b-c)^3\rvert}{x^2}+bc$ $c,\quad -2{,}5\le a\le 1{,}5\quad \Delta a=1\quad 2\le b\le 5\quad \Delta b=1$	a, b, x, y
5	$x=\frac{a^2-b^3}{\lvert abc\rvert}-(b+1)^2 \qquad y=\frac{x^2}{b+1}-2ab$ $b,\quad 1\le a\le 5\quad \Delta a=1\quad 3\le c\le 9\quad \Delta c=2$	a, c, x, y
6	$x=\frac{\lvert a-c^2\rvert}{2a}+ac \qquad y=\frac{(3a+4)x-10}{ab-c^2}$ $a,\quad -7{,}5\le b\le 8{,}5\quad \Delta b=4\quad 3\le c\le 9\quad \Delta c=2$	b, c, x, y

7	$x = \dfrac{4a + \lvert bc - 3\rvert}{ab - bc}$ $\quad y = \dfrac{\lvert ax - c^2\rvert}{bc - 3}$ $b, \quad -3{,}5 \le a \le 4{,}5 \quad \Delta a = 2 \quad 2 \le c \le 5 \quad \Delta c = 1$	a, c, x, y
8	$x = a(b + c) \cdot \lvert abc - 19\rvert$ $\quad y = \dfrac{a^3 + \lvert x - 2b^2 c\rvert}{ac^2 + ab - 1}$ $a, \quad -2 \le b \le 2 \quad \Delta b = 1 \quad 1{,}5 \le c \le 6 \quad \Delta c = 1{,}5$	b, c, x, y
9	$x = \dfrac{bc + c^2 a}{\lvert 2a - b^3\rvert}$ $\quad y = \dfrac{abc}{\lvert a^2 + bc\rvert} + 3ac$ $c, \quad 2 \le a \le 8 \quad \Delta a = 1 \quad -3 \le b \le 3 \quad \Delta b = 1{,}5$	a, b, x, y
10	$x = \dfrac{ab + c}{a^3} - \lvert b + 2\rvert$ $\quad y = \dfrac{a(b + 2)}{\lvert x\rvert} + \dfrac{(b + 2)^2}{a - c}$ $a, \quad -4{,}5 \le b \le 4{,}5 \quad \Delta b = 3 \quad 2 \le c \le 6 \quad \Delta c = 1$	b, c, x, y

Завдання 5. Програмування алгоритмів формування і обробки одновимірних масивів

Мета роботи: скласти блок-схему алгоритму і програму обчислення і виведення значень.

№ п/п	Вид завдання
1	а) Визначити кількість і суму від'ємних елементів масиву *r* і вивести їх індекси $r_i = \begin{cases} \dfrac{a - 3}{\sqrt{x_i^2 + 2}} & , x_i < 1 \\ 2{,}3ax_i & , x_i \ge 1 \end{cases} \qquad i = 1, N$ б) Записати додатні елементи масиву X = (x_1, x_2, …, x_{12}) підряд в масив Y = (y_1, y_2, …,$y_к$). Визначити максимальний елемент в масиві Y.

2	а) Визначити кількість і добуток додатних елементів в масиві *s* $s_k = \begin{cases} B \cdot \ln\left(x_k^2 + 1\right) & , x_k < 1 \\ -2{,}7 \sin^2 x_k & , x_k \geq 1 \end{cases} \qquad k = 1, M$ б) Записати елементи масиву A = (a_1, a_2, …, a_{15}) з парними індексами підряд в масив B=(b_1, b_2, ..., b_k). Обчислити суму від'ємних елементів масиву B.
3	а) Визначити суму від'ємних і добуток додатних елементів массиву *c* $c_i = \begin{cases} tg \dfrac{x_i}{\pi} & , x_i < 1 \\ 2{,}8 z x_i^2 & , x_i \geq 1 \end{cases} \qquad i = 1, M$ б) Записати елементи масиву X=(x_1, x_2, ..., x_{12}), які задовольняють умові x_i є [1;2], підряд в масив У=(y_1, y_2, ..., $y_к$). Визначити мінімальний елемент в масиві Y.
4	а) Визначити суму і кількість ненульових елементів масиву *a* $a_k = \begin{cases} \dfrac{3{,}5d}{\lvert x_k \rvert + 2} & , x_k < 2 \\ d^2 - x_k^2 & , x_k \geq 2 \end{cases} \qquad k = 1, Z$ б) Записати додатні елементи масиву X=(x_1, x_2, ..., x_{14}) підряд в масив У=(y_1, y_2, ..., $y_к$). Обчислити додаток елементів масиву з парними індексами.
5	а) Визначити суму і кількість тих елементів масиву *t*, які менше Q $t_k = \begin{cases} -\sqrt{x_k + 3} & , x_k \geq 4 \\ 2\pi \cos x_k & , x_k < 4 \end{cases} \qquad k = 1, M$ б) Записати елементи масиву X=(x_1, x_2, ..., x_{12}) з непарними індексами підряд в масив У=(y_1, y_2, ..., $y_к$). Обчислити додаток від'ємних елементів масиву Y.
6	а) Визначити середнє арифметичне додатних елементів масиву *v* $v_k = \begin{cases} (a - x_k) \sin x_k & , x_k < 2{,}5 \\ 2\sqrt{x_k - 1} & , x_k \geq 2{,}5 \end{cases} \qquad k = 1, M$

	б) Записати від'ємні елементи масиву X=(x_1, x_2, ..., x_{10}) підряд в масив У=(y_1, y_2, ..., $y_к$). Визначити максимальний елемент в масиві Y і замінити його сумою перших двох елементів.
7	а) Визначити відсоток додатних і від'ємних елементів в масиві *a* $a_k = \begin{cases} 0{,}5x_k + b & ,x_k < 0 \\ 3x_k\sqrt{x_k} - b & ,x_k \ge 0 \end{cases} \qquad k = 1,P$ б) Записати елементи масиву X=(x_1, x_2, ..., x_{17}), які задовольняють умові x_i є [2;3], підряд в масив У=(y_1, y_2, ..., $y_к$). Визначити максимальний елемент масиву У і його номер.
8	а) Визначити середнє арифметичне від'ємних елементів масиву *u* $u_i = \begin{cases} a\ln\left(1 + x_i^2\right) & ,x_i < 1 \\ 2{,}5\sin x_i & ,x_i \ge 1 \end{cases} \qquad i = 1,M$ б) Записати елементи масиву X=(x_1, x_2, ..., x_{12}) підряд в масив У. Визначити мінімальний елемент в масиві У і поміняти його місцями з елементом y_1.
9	а) Визначити кількість елементів масиву *d*, які задовольняють умові $0<d_k<1$ і вивести їх індекси. $d_k = \begin{cases} 3{,}8e^{-x_k} + a & ,x_k < 0 \\ a\sqrt{x_k + 2} & ,x_k \ge 0 \end{cases} \qquad k = 1,B$ б) Записати елементи масиву X=(x_1, x_2, ..., x_{14}) з парними індексами підряд в масив У=(y_1, y_2, ..., $y_к$). Визначити мінімальний по модулю елемент масиву У і його номер.
10	а) Визначити кількість і суму елементів $z_i<a$ і вивести їх індекси $z_i = \begin{cases} \dfrac{b^2 + 3}{x_i^2 + 1} & ,\lvert x_i\rvert \ge 1 \\ 4{,}2\sin^2 x_i & ,\lvert x_i\rvert < 1 \end{cases} \qquad i = 1,R$ б) Записати сім перших додатних елементів масиву X=(x_1, x_2, ..., x_{15}) підряд в масив У=(y_1, y_2, …, y_7). Визначити максимальний елемент в масиві У і замінити його значенням (-1).

Завдання 6. Програмування алгоритмів формування і обробки двовимірних масивів

Мета роботи: скласти блок-схему алгоритму і програму обчислення і виведення значень.

№ п/п	Вид завдання
1	а) Для масиву D(N,M) знайти суму всіх елементів, розташованих в рядках з від'ємними елементами на головній діагоналі. б) Сформувати масив H(N), кожен елемент якого рівняється сумі абсолютних значень елементів відповідного рядка масиву C(N,M).
2	а) Для масиву K(N,M) знайти відношення мінімального елементу до максимального елементу. Вивести індекси мінімального і максимального елементів. б) Сформувати масив X(M), кожен елемент якого рівняється добутку абсолютних значень елементів відповідного стовпця масиву T(N,M).
3	а) Для масиву X(N,M) знайти відношення суми елементів, розташованих нижче за головну діагональ, до суми елементів, розташованих вище за неї. Якщо остання сума рівна 0 – вивести відповідне повідомлення. б) Сформувати масив B(N), кожен елемент якого рівняється мінімальному елементу відповідного рядка масиву K(N,M).
4	а) Для масиву Z(N,M) знайти середнє арифметичне елементів і кількість елементів, які менше середнього арифметичного значення. б) Сформувати масив B(M), кожен елемент якого рівняється максимальному елементу відповідного стовпця масиву C(N,M).
5	а) Для масиву T(N,M) знайти відношення кількості додатних елементів до кількості від'ємних. Якщо від'ємних елементів немає – вивести відповідне повідомлення. б) Сформувати масив B(N), кожен елемент якого рівняється максимальному елементу відповідного рядка масиву K(N,M).

6	а) Для масиву A(N,M) знайти мінімальний і максимальний елементи і поміняти їх місцями. б) Сформувати масив H(N), кожен елемент якого рівняється добутку від'ємних елементів відповідного рядка масиву T(N,M).
7	а) Для масиву X(N,M) знайти мінімальний елемент і замінити його сумою від'ємних елементів. б) Сформувати масив B(M), кожен елемент якого рівняється середньому арифметичному відповідного стовпця масиву A(N,M).
8	а) Для масиву Z(N,M) знайти суму додатних і добуток від'ємних елементів k-го стовпця. б) Для масиву T(N,M) знайти в кожному рядку мінімальний елемент і помістити його на місце першого елементу відповідного рядка.
9	а) Для масиву L(N,M) знайти різницю між середнім арифметичним і середнім геометричним елементів. б) Для масиву T(N,M) замінити останній елемент кожного рядка сумою елементів цього ж рядка.
10	а) Для масиву K(N,M) елементи, розташовані нижче за головну діагональ зменшити в двоє, а елементи, розташовані вище за головну діагональ, – збільшити удвічі. б) Для масиву A(N,M) замінити останній елемент кожного стовпця сумою елементів цього ж стовпця.

Завдання 7. Програмування алгоритмів, які містять підпрограми

Мета роботи: скласти блок-схему алгоритму і програму основного завдання і допоміжної під задачі.

№ п/п	Вид завдання
1	За допомогою підпрограми-функції скласти таблиці значень функцій $y(x)=\frac{3{,}2-\log_x\left(x^2+1\right)}{\log_{x+1}(x+2)}$, $g(x)=\sqrt{1+\log_x^2(x+3)}$ де $2 \le x \le 12 \quad \Delta x = 1{,}5$

2	Скласти таблиці значень функцій на відрізку [1,2;3,6], *Δx=0,4* $y(x)=\frac{3{,}5\log_{x+2}x+1}{\log_{x^2}(x+2)}$ і $g(x)=\frac{a\log_{2x}(x+1)}{a^2+\log^2_{x+3}x}$. Використовувати підпрограму-функцію для обчислення $\log_k n=\frac{\ln n}{\ln k}$.
3	Сформувати масиви чисел $a_j=\frac{2{,}8\sin(j^2+2)}{\cos^2(j^2+2)+3{,}6}$ і $b_k=\frac{k\sin(k^2+3)}{4\cos^2(k^2-0{,}6)-8{,}2}$ де $j=1,m;\quad k=1,n$ з використанням підпрограми-функції. Передбачити введення змінної *z*.
4	Сформувати масиви чисел $n_k=\frac{2{,}6\sqrt{\sin k+3{,}4}}{1+\sqrt{\sin k^2+1{,}8}}$ і $m_i=\frac{3{,}8\sqrt{\sin i^2+2{,}4}}{4+5\sqrt{\sin i+1{,}6}}$ де $k=1,a;\quad i=1,b$ з використанням підпрограми-функції.
5	Скласти таблиці значень функцій $f(x)=\frac{a+2\ln^2(x+2{,}5)}{3{,}6-3\ln^2(x+0{,}8)}$ і $g(x)=\frac{8{,}2+3{,}6\ln^2(x+0{,}7)}{2{,}8-2{,}4a\ln^2(x+1{,}4)}$ *для* $-5\le x\le 10,\quad \Delta x=1{,}5$ з використанням підпрограми-функції. Передбачити введення змінної *a*.
6	Обчислити з використанням підпрограми-функції : $y=hf(x_1)+\frac{h^2}{2}f(x_2^2)+\frac{h^3}{3}f^2(x_1+x_2)$, де $f(x)=\frac{ax\sqrt{x}+\ln\lvert x+b\rvert}{2{,}58c}$, $h=\frac{x_2-x_1}{10}$ для $1\le a\le 8,\quad \Delta a=0{,}5$. Передбачити введення змінних *b, c, x_1, x_2*.
7	Скласти таблиці значень функцій $z(x)=\frac{3{,}4+\log_x(x+2)}{4\log_{x+1}x}$ і $g(x)=\frac{5{,}9-\log_{x^2}(x+1)}{2{,}8\log_{x+1}x}$ для $2\le x\le 10\quad \Delta x=1{,}5$. Для $\log_k n=\frac{\ln n}{\ln k}$ використовувати підпрограму-функцію.
8	Сформувати масиви $x_i=1{,}5i^2-6{,}9\sin(i-1)+8{,}2$ і $z_k=2{,}3k^2+8{,}1\sin(k+2)-3{,}4$, якщо $i=1,p;\quad k=1,h$ з використанням підпрограми-функції.

9	За допомогою підпрограми-функції скласти таблиці значень функцій $Q(x)=e^{-2x}+\sqrt[3]{\sin x+2{,}8}$ і $P(x)=e^{x+1}+\sqrt[3]{\sin x-0{,}4}$ для $0{,}2\le x\le 1{,}6 \quad \Delta x=0{,}3.$
10	Скласти таблиці значень функцій $S(x)=1{,}5\cos^3 x+\frac{2}{3}\sqrt[3]{1+\sin^2 x}$ і $t(x)=2{,}6\cos^3 x+\frac{1}{7}\sqrt[3]{1-\sin^2 x}$ для $0{,}2\le x\le 1{,}4 \quad \Delta x=0{,}3$. Використовувати підпрограму-функцію.

ДОДАТОК 1

ПОРЯДОК ВИБОРУ ВАРІАНТУ ЗАВДАННЯ

Номери варіантів завдань вибираються по буквах прізвища студента відповідно до таблиці:

Номер варіанту завдання	1	2	3	4	5	6	7	8	9	10
Букви	А	Б	В	Г	Д	Е	Ж	З	И	К
	Л	М	Н	О	П	Р	С	Т	У	Ф
	Х	Ц	Ч	Ш	Щ	Ы	Ь	Э	Ю	Я

Наприклад, необхідно виконати завдання 1, 2, 3. Для студента з прізвищем Іванов варіанти в кожному завданні будуть такими:

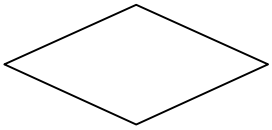
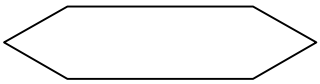
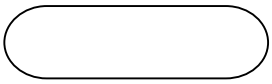
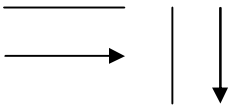
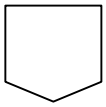
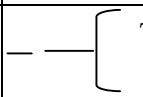
1 завдання – варіант 9 (буква И)

2 завдання – варіант 3 (буква В)

3 завдання – варіант 1 (буква А)

ДОДАТОК 2

ГРАФІЧНІ СИМВОЛИ, ЯКІ ВЖИВАЮТЬСЯ ПРИ СКЛАДАННІ БЛОК-СХЕМ

№ п/п	Найменування	Позначення	Функція, яка виконується
1	Введення, виведення		Введення, виведення даних
2	Документ		Виведення, друк результатів на папір
3	Процес		Обчислення арифметичних операцій
4	Рішення		Перевірка умов
5	Модифікація		Початок циклу
6	Зумовлений процес		Обчислення по підпрограмах
7	Початок, кінець		Початок, кінець процесу обробки даних
8	Лінії потоку		Зображення зв'язків між блоками
9	З'єднувач		Вказівка зв'язку між перерваними лініями потоку
10	Міжсторінковий з'єднувач		Вказівка зв'язку між частинами блок-схеми, розташованих на різних листах
11	Коментар		Запис пояснення до блоку або до лінії потоку

ДОДАТОК 3

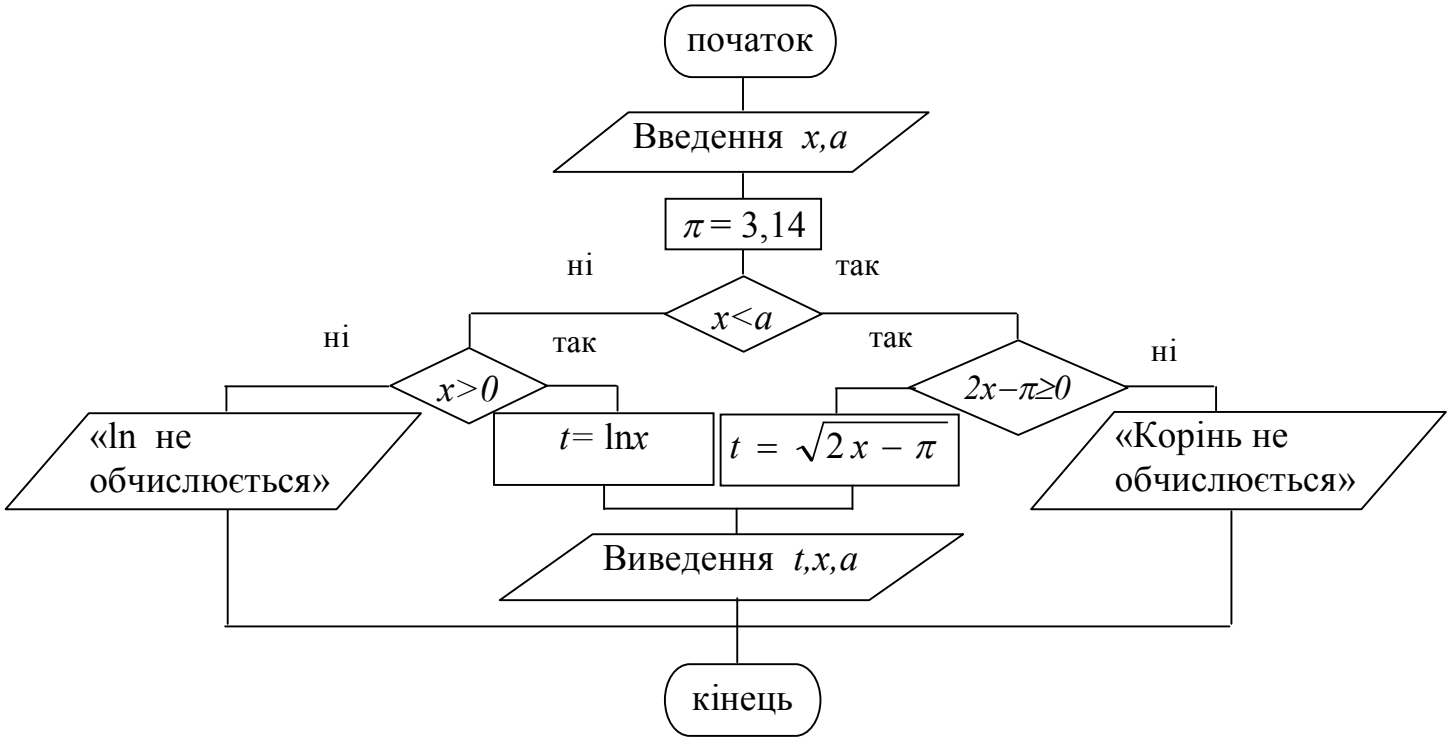
ПРИКЛАД РОЗРОБКИ АЛГОРИТМУ І ПРОГРАМИ РОЗГАЛУЖЕНИХ ПРОЦЕСІВ

Обчислити значення t використовуючи формулу:

$$t = \begin{cases} \sqrt{2x - \pi}, & якщо \quad x < a \\ \ln x, & якщо \quad x \geq a \end{cases}$$

Початкові дані: x, a.

Схема алгоритму рішення задачі має вигляд



Текст програми рішення задачі має вигляд:

```
Program rabota;
Label 1;
var x, a, t: real;
begin
writeln (‘vvod x, a’);  readln (x, a);
If x < a Then
        If    2 * x - pi >= 0 Then  t: = Sqrt(2 * x - pi)
        Else    begin writeln (‘Корінь не обчислюється’);  goto 1 end
Else
        If  x > 0 Then   t: = Ln(x)
        Else    begin writeln (‘ln не обчислюється’);  goto 1 end;
Writeln ('При x=’,  x:5:2, ‘   a=’,  a:5:2, ‘     t=’,  t:5:2);
1: end.
```

ДОДАТОК 4

ПРИКЛАДИ РОЗРОБКИ АЛГОРИТМУ І ПРОГРАМИ ЦИКЛІЧНИХ ПРОЦЕСІВ

Приклад 1.

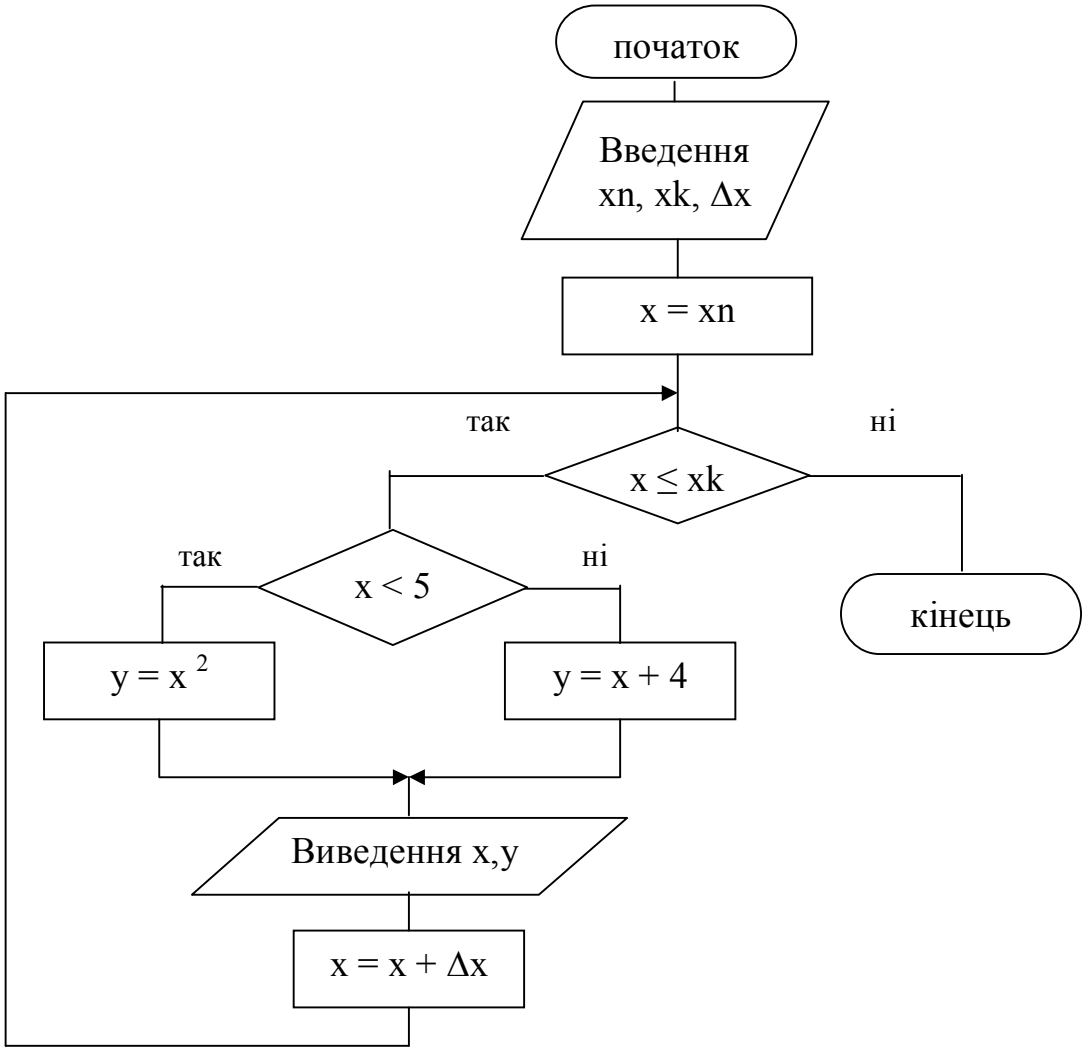
Обчислити значення Y використовуючи формулу:

$$y = \begin{cases} x + 4, & якщо \quad x \geq 5 \\ x^2, & якщо \quad x < 5 \end{cases} \qquad -2 \leq x \leq 8 \quad \Delta x = 1$$

Початкові дані: $x_н$, $x_к$, Δx.

Для обчислення змінної у організувати цикл з передумовою.

Схема алгоритму рішення задачі має вигляд:



Текст програми рішення задачі має вигляд:

```
Program pr1;
Var   xn,xk,dx,x,y:real;
begin
writeln (‘Vvod xn,xk,dx’);  readln (xn,xk,dx);
x := xn;
while x <= xk do
begin
if  x < 5 then  y := sqr(x)  else  y := x + 4;
writeln  (‘x =’, x : 5 : 3, ‘    y =’, y : 5 : 3);
x := x + dx
end;
end.
```

Приклад 2.

Обчислити значення X використовуючи формулу:

$$x = \frac{2{,}5a - c}{a^2 - 2ac}$$

$$-3 \le a \le 3; \quad \Delta a = 1{,}5$$

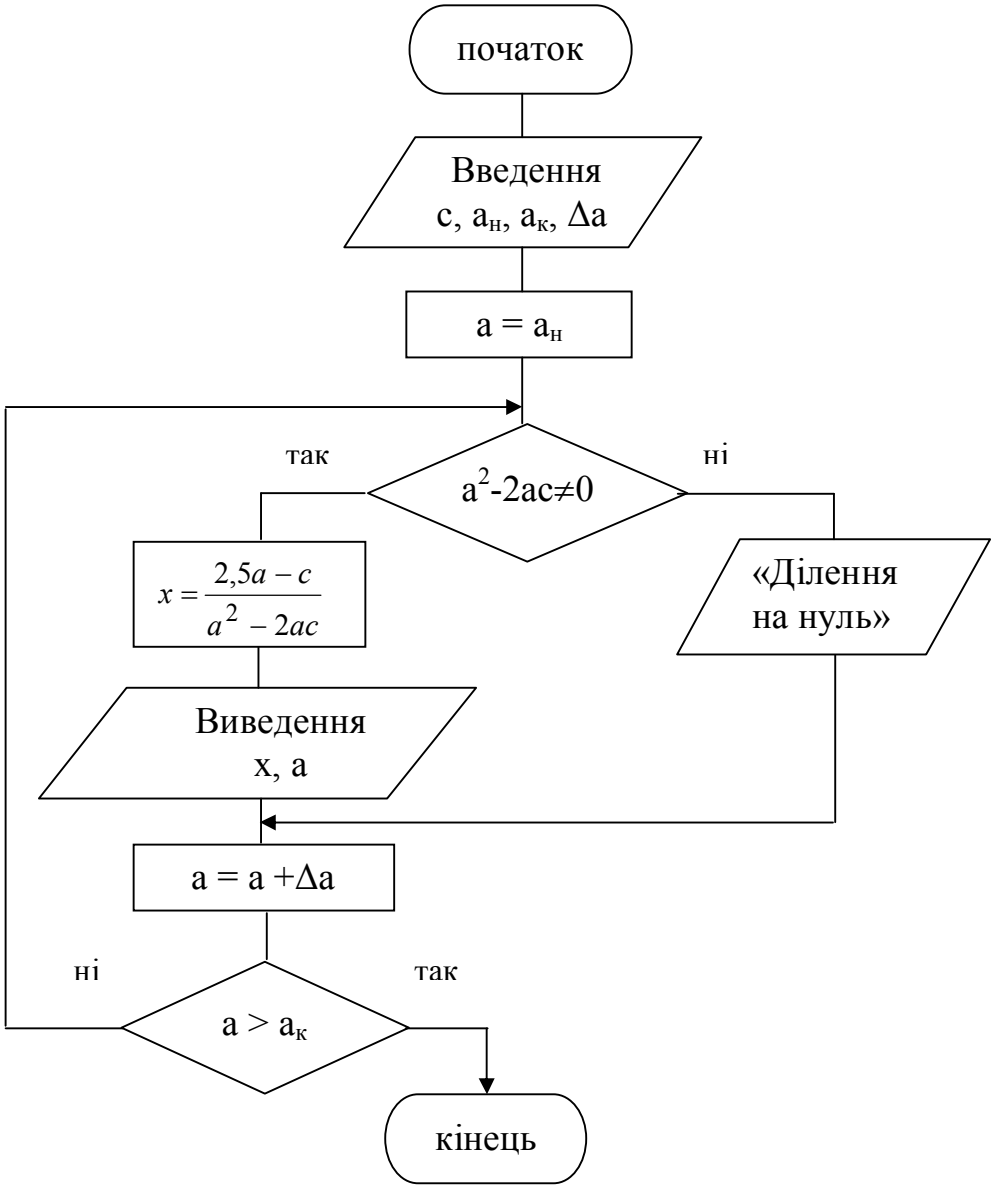
Початкові дані: $a_н$, $a_к$, Δa, c.

Для обчислення змінної X організувати цикл з постумовою.

Текст програми рішення задачі має вигляд:

```
Program pr2;
Label 1;
Var   an,ak,da,a,c,x:real;
begin
writeln (‘Vvod an,ak,da,c’);   readln (an,ak,da,c);
a := an;
repeat
        if sqr(a) – 2 * a * c < > 0 then  x := (2.5 * a - c) / (sqr(a) – 2 * a * c)
        else
        begin  writeln  (‘a =’, a : 5 : 3, ‘    ділення на нуль’); goto 1 end;
        writeln  (‘a =’, a : 5 : 3, ‘    x =’, x : 5 : 3);
        1: a := a + da
until a > ak;
end.
```

Схема алгоритму рішення задачі має вигляд:



Приклад 3.

Обчислити значення Y використовуючи формулу:

$$y = \frac{\sin(ax) + 2}{1 - x}$$

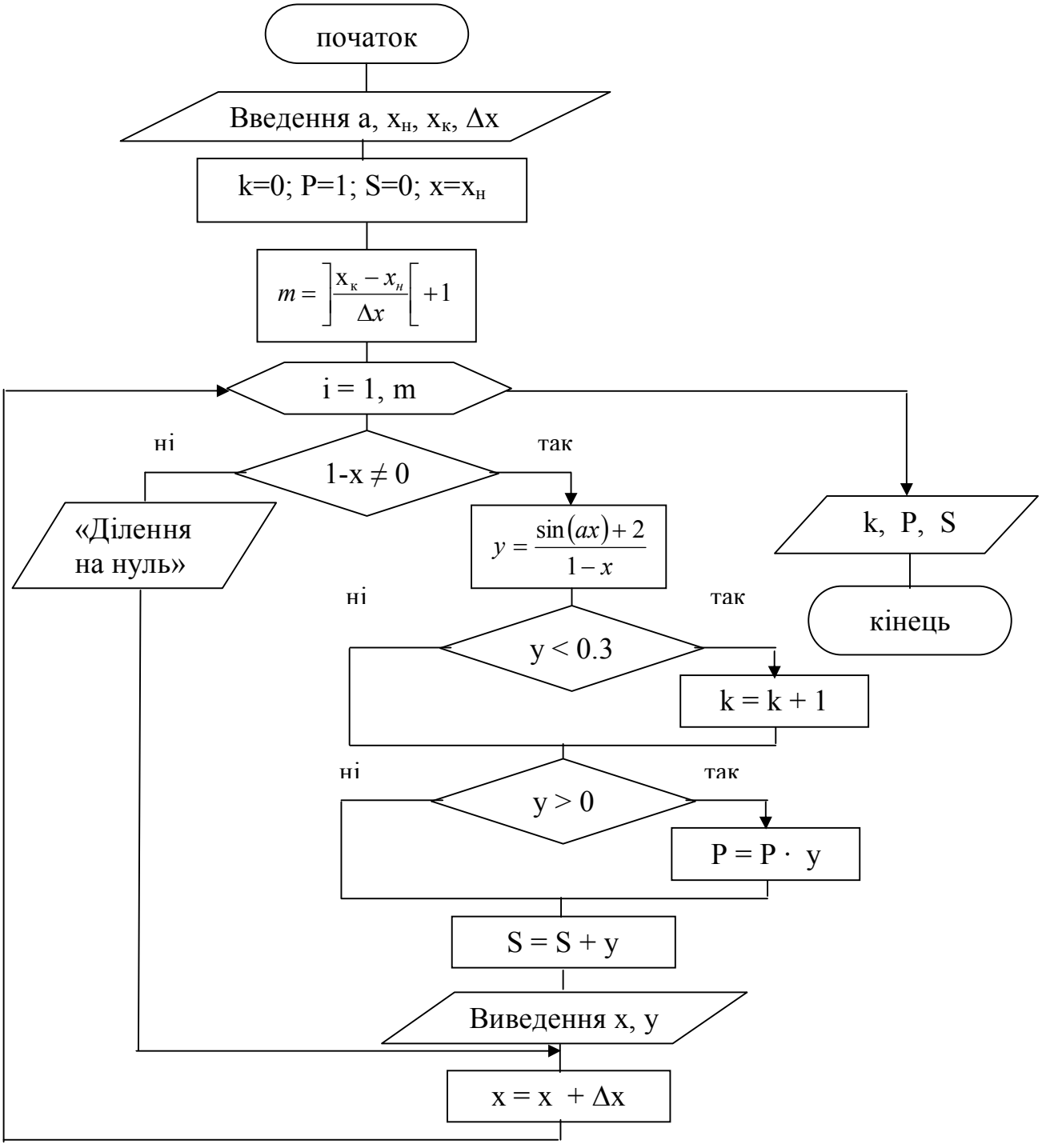
$$-5 \le x \le 5 \quad \Delta x = 1$$

Визначити $P = \prod_{y>0} y$, $S = \sum y$ *та кількість* $y < 0.3$

Початкові дані: $x_н$, $x_к$, Δx, a.

Для обчислення змінної Y організувати цикл з відомим числом повторень.

Схема алгоритму рішення задачі має вигляд:



Текст програми рішення задачі має вигляд:

```
Program pr3;
label 1;
var xn,xk,dx,x,a,s,p,y :  real;    i,m,k : integer;
begin
writeln (‘Vvod xn,xk,dx,a’); readln (xn,xk,dx,a);
```

```
k := 0;  s := 0;  p := 1; x := xn;
m := trunc ((xk-xn) / dx) + 1;
for i := 1 to m do
begin
        if 1 – x < > 0 then  y := (sin (a * x) + 2) / (1 - x)
        else
        begin  writeln (‘x =’, x : 5 : 3, ‘    delenie na 0’);  goto 1 end;
        if y < 0.3 then k := k + 1;
        if y > 0 then p := p * y;
        s := s + y;
        writeln (‘x =’, x : 5 : 3, ‘    y =’, y : 5 : 3);
        1: x := x + dx;
end;
writeln (‘k =’, k : 3, ‘   s =’, s : 5 : 3,  ‘    p =’, p : 5 : 3);
end.
```

Приклад 4.

Обчислити значення X та Y використовуючи формули та вивести результати у формі таблиці:

$$y = \frac{x^2 - ac}{c^2 - 1} \qquad x = \sqrt{|b^2 - a|}$$

$$2 \le a \le 7 \qquad \Delta a = 1$$

$$-1{,}5 \le c \le 2 \qquad \Delta c = 0{,}5$$

$$b = 3{,}2$$

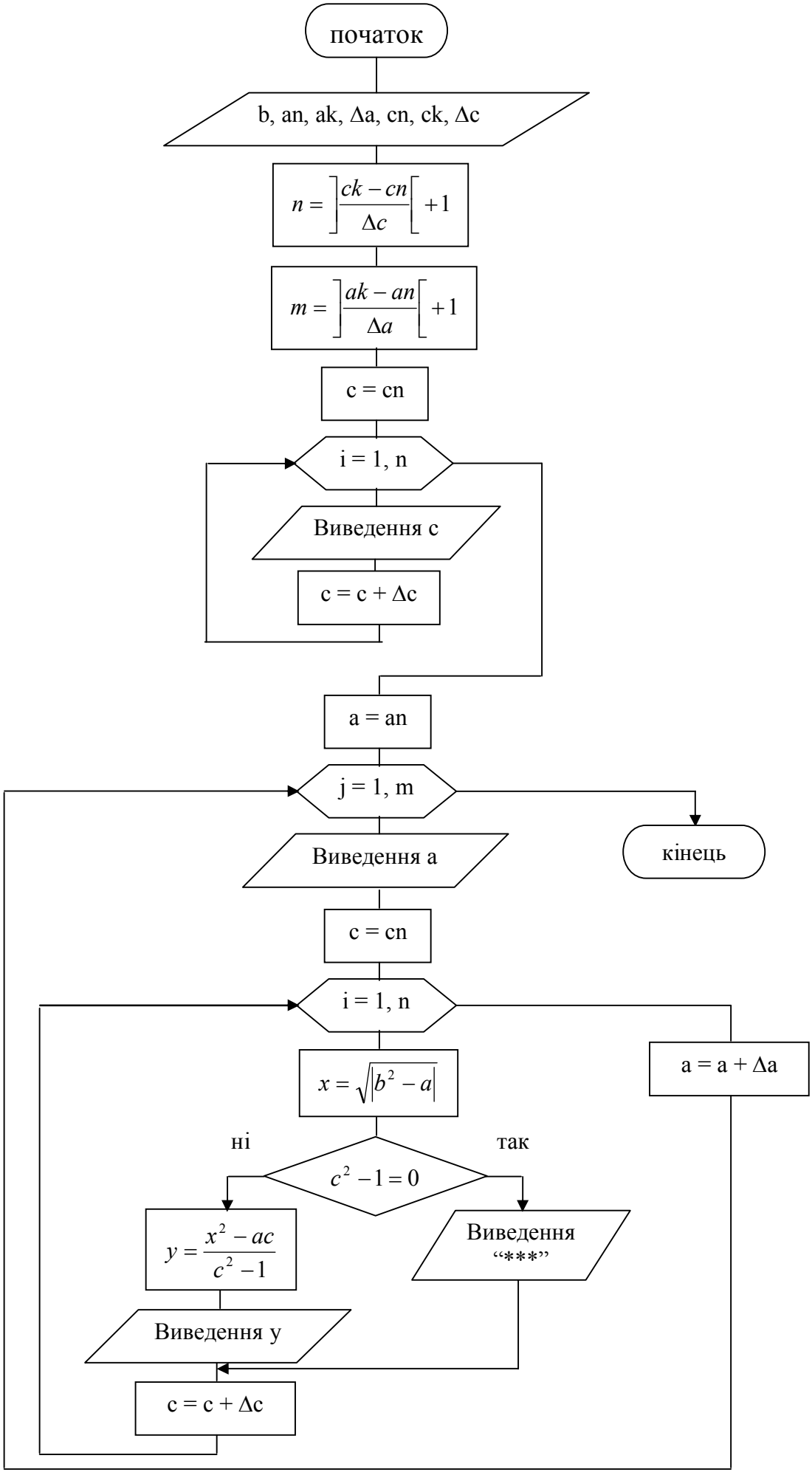
Початкові дані: $a_н$, $a_к$, Δa, $c_н$, $c_к$, Δc, b.

Для організації зовнішнього і внутрішнього циклів використовуватиме цикли з відомим числом повторень.

Виведення значень Y оформити у вигляді таблиці:

a\c	c_1	c_2	…	c_n
a_1				
…				
a_m				

Схема алгоритму рішення задачі має вигляд:


початок
b, an, ak, Δa, cn, ck, Δc
$n = \left\lceil \frac{ck - cn}{\Delta c} \right\rceil + 1$
$m = \left\lceil \frac{ak - an}{\Delta a} \right\rceil + 1$
c = cn
i = 1, n
Виведення c
c = c + Δc
a = an
j = 1, m
Виведення a
кінець
c = cn
i = 1, n
a = a + Δa
$x = \sqrt{|b^2 - a|}$
ні
так
$c^2 - 1 = 0$
$y = \frac{x^2 - ac}{c^2 - 1}$
Виведення "***"
Виведення y
c = c + Δc

Текст програми рішення задачі має вигляд:

```
program pr4;
var b, an, ak, da, cn, ck, dc,  c, a, x, y : real;
m, n, i, j: integer;
begin
writeln('Enter b,an,ak,da,cn,ck,dc');
readln(b,an,ak,da,cn,ck,dc);
n := trunc ((ck-cn) / dc) + 1;
m := trunc ((ak-an) / da) + 1;
c := cn; write ('a \ c');
for i := 1 to n do
begin write('   ' : 4, c : 5 : 3);  c := c + dc    end;
writeln;
a := an;
for j := 1 to m do
        begin
        write (a : 5 : 3);
        c := cn;
        for i := 1 to n do
                begin
                x := sqrt (abs (sqr(b) - a));
                if  sqr(c) – 1 = 0 then  write ('    ***     ')
                else
                begin
                y := (sqr(x) – a * c) / (sqr(c) - 1);
                write ('  ' : 2, y : 5 : 3, '  ' : 2)
                end;
                c := c + dc
                end;
        a := a + da;
        writeln;
        end;
end.
```

ДОДАТОК 5

ПРИКЛАДИ РОЗРОБКИ АЛГОРИТМУ І ПРОГРАМИ ОБРОБКИ ОДНОВИМІРНИХ МАСИВІВ

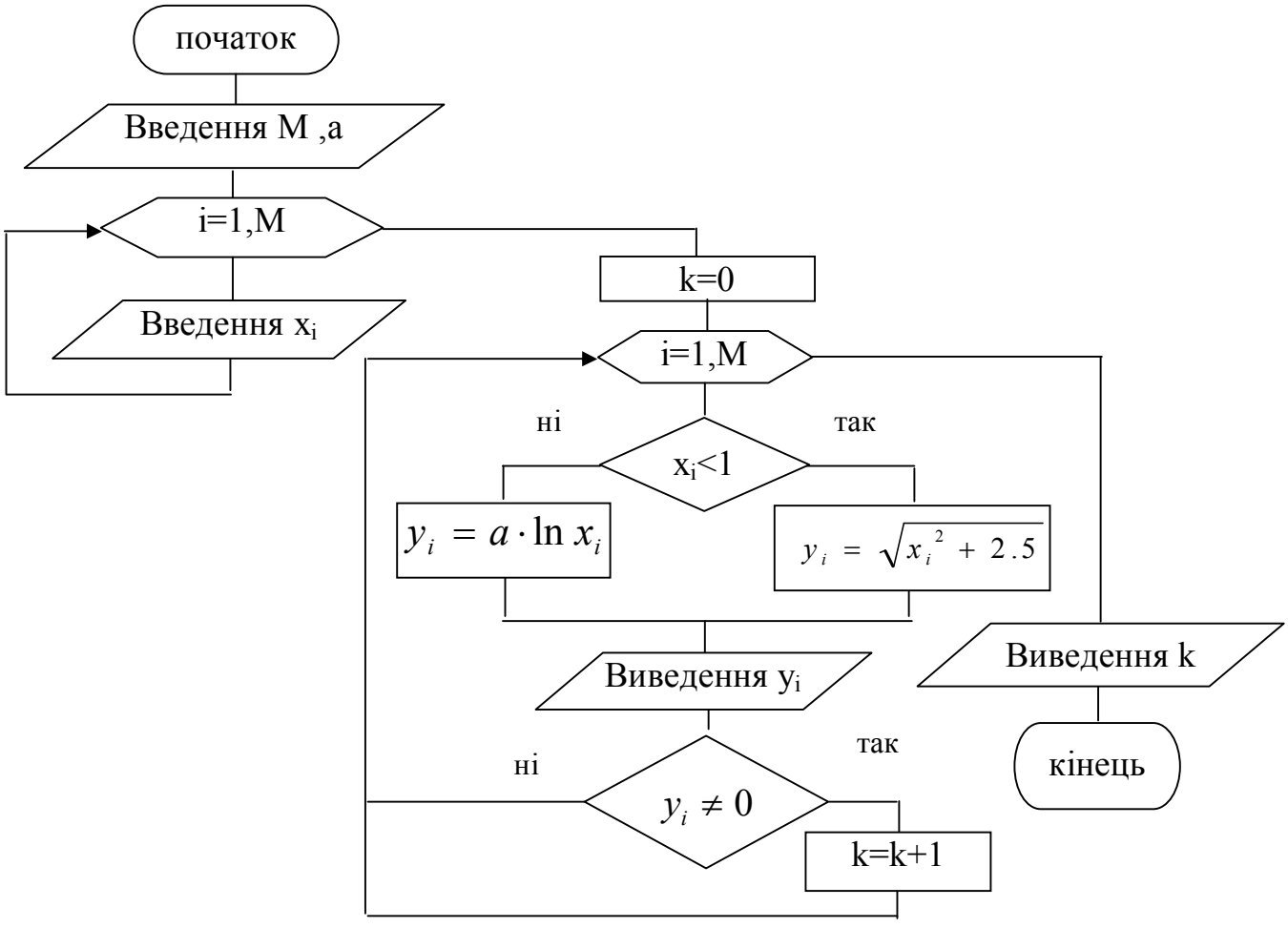
Приклад 1.

Визначити кількість ненульових елементів в масиві У, елементи якого обчислюються за формулами:

$$y_i = \begin{cases} a \ln x_i , & якщо \quad x_i \geq 1 \\ \sqrt{x^2{}_i + 2{,}5} , & якщо \quad x_i < 1 \end{cases} \qquad \text{де} \quad i = 1, M$$

Початкові дані: масив X(M), a.

Схема алгоритму рішення задачі має вигляд:



Текст програми рішення задачі має вигляд:

```
Program pr1;
type mas = array [1 . . 10]  of   real;
```

```
var x,y: mas;    i,m,k: integer;    a: real;
begin
writeln('Vvod m,a');   readln (m,a);
writeln(' vvod   elementiv massiva  X');
for i := 1 to  m  do    read (x [i] );
k := 0;
for i := 1  to  m  do
        begin
        if  x [i] < 1 then  y[i] := sqrt (sqr (x[i]) + 2.5) else y[i] := a * ln (x[i]);
        writeln('y[', i, ']=', y[i] : 5 : 3);
        if  y [i] <> 0 then  k := k  + 1;
        end;
writeln('k =', k: 3);
end.
```

Приклад 2.

Елементи масиву X(N), які більше А записати в масив У. Визначити мінімальне значення у масиві У та його номер.

Початкові дані: масив X(N), А.

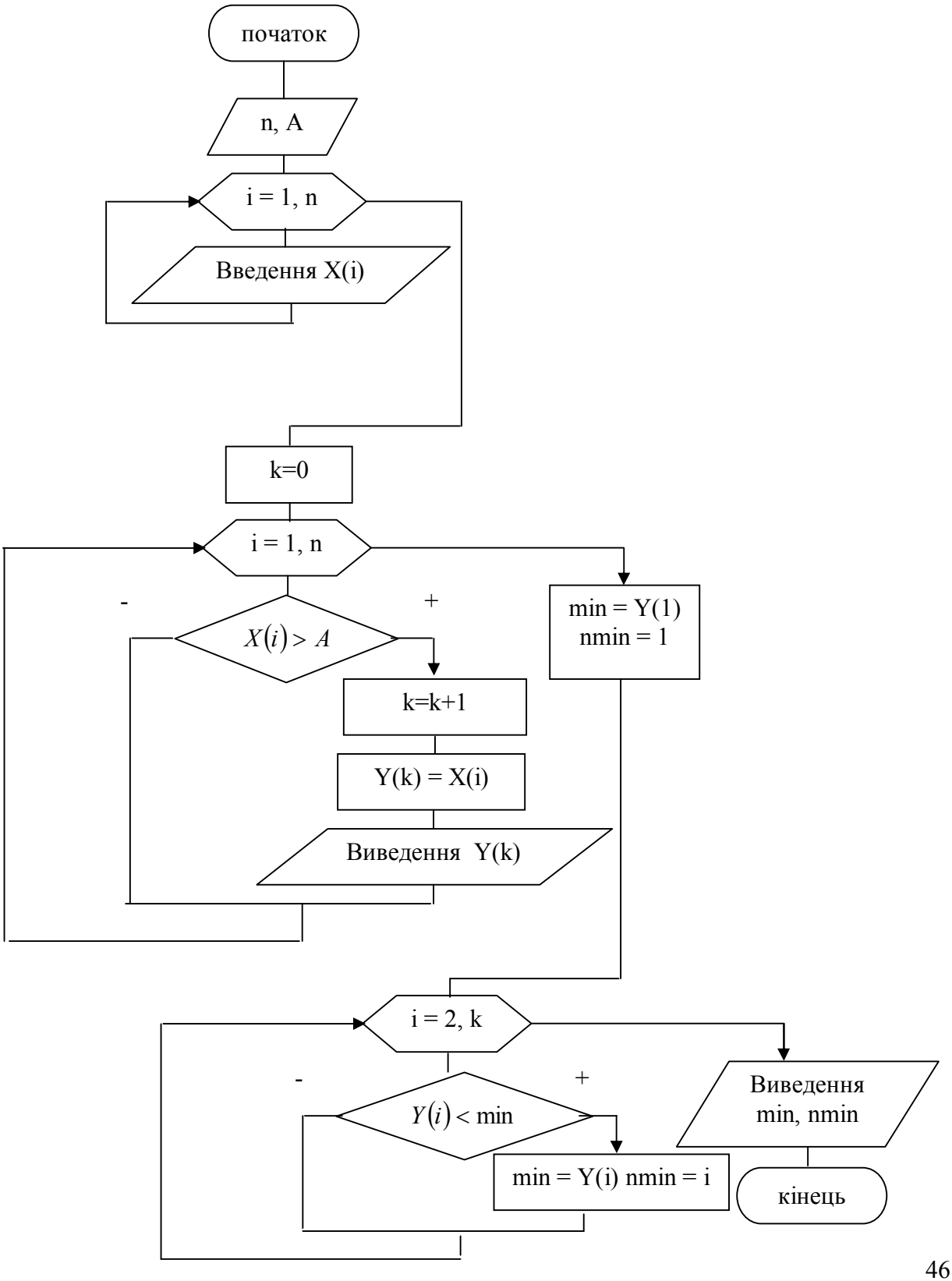
Текст програми рішення задачі має вигляд:

```
Program pr2;
type mas = array [1 . . 10]  of   real;
var  x,y: mas;       a, min: real;        i, k, n, nmin: integer;
begin
writeln('Enter a, n');   readln(a,n);
for i := 1  to  n  do
begin
write ('x ['', i ,' ]=' );  readln (x [i])
end;
k:=0;
for i := 1 to  n  do
if  x [i] > a then
begin
k := k + 1;
y[k] := x[i];
writeln ('y[ ', k, ' ]=', y[k] : 5 : 2);
end;
```

Введення елементів масиву в стовбець

```
min := y [1]; nmin := 1;
for i := 2  to  k do
if  y [i] < min then  begin   min := y [i];  nmin := i  end;
writeln (‘min =’, min : 5 : 2, ’      nmim = ’, nmin : 2);
end.
```

Схема алгоритму рішення задачі має вигляд:

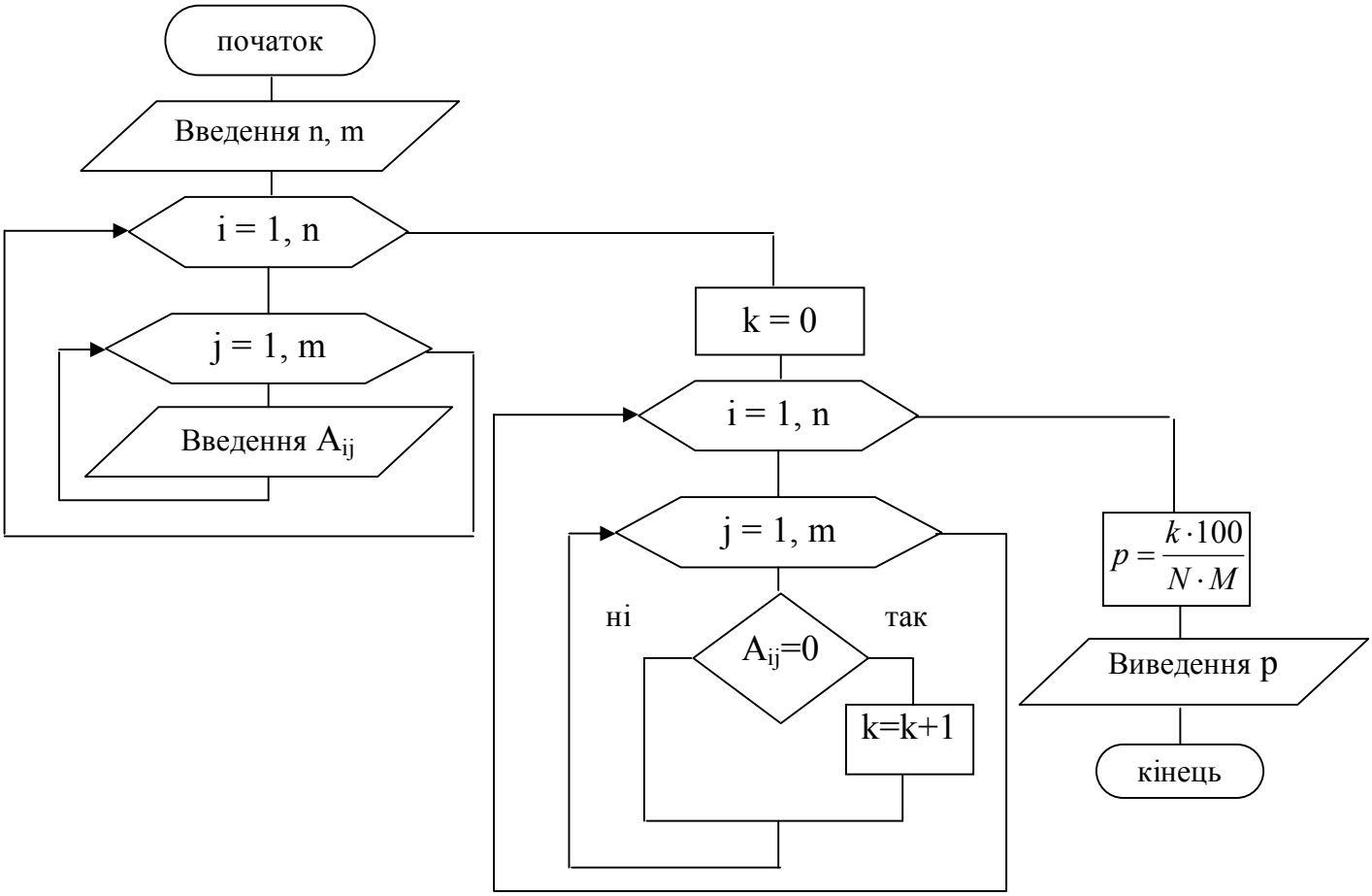


ДОДАТОК 6

ПРИКЛАД РОЗРОБКИ АЛГОРИТМУ І ПРОГРАМИ ОБРОБКИ ДВОВИМІРНИХ МАСИВІВ

Для масиву A(n,m) знайти відсоток нульових елементів.

Схема алгоритму рішення задачі має вигляд:



Текст програми рішення задачі має вигляд:

```
program pr1;
type mas1=array[1..10,1..10] of integer;
var a:mas1;    i,j,n,m,k: integer;     p: real;
begin
writeln(‘vvod n,m’);  readln(n,m);   writeln(‘vvod  massiva A’);
for i:=1 to n do
for j:=1 to m do
read (a[i,j]);
k:=0;
for i:=1 to n do
for j:=1 to m do
if a[i,j] =0 then k:=k+1;
p:=k*100/(n*m);  writeln('p=', p:5:2);  end.
```

МЕТОДИЧНИЙ ПОСІБНИК

ДО ВИКОНАННЯ КОНТРОЛЬНИХ ЗАВДАНЬ

У СЕРЕДОВИЩІ PASCAL

(навчально-методичний посібник для студентів заочного відділення)

Укладачі: Лазєбна Людмила Олександрівна