МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ
Національний технічний університет України
“Київський політехнічний інститут”

СТРУКТУРИ ДАНИХ ТА АЛГОРИТМИ – 2.
СКЛАДНІ СТРУКТУРИ ДАНИХ ТА АЛГОРИТМИ

Завдання до виконання
лабораторних робіт по кредитному модулю
"Структури даних та алгоритми – 2.
Складні структури”
для студентів напрямку 6.050102
«Комп’ютерна інженерія»

Рекомендовано вченою радою факультету прикладної математики НТУУ «КПІ»

Київ 2012

МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ
Національний технічний університет України
“Київський політехнічний інститут”

СТРУКТУРИ ДАНИХ ТА АЛГОРИТМИ – 2.
СКЛАДНІ СТРУКТУРИ ДАНИХ ТА АЛГОРИТМИ

Завдання до виконання
лабораторних робіт по кредитному модулю
"Структури даних та алгоритми – 2.
Складні структури”
для студентів напрямку 6.050102
«Комп’ютерна інженерія»

Затверджено
на засіданні кафедри системного
програмування і спеціалізованих
комп’ютерних систем
ФПМ НТУУ «КПІ»
Протокол № 6 від 12.12.2012

Київ НТУУ «КПІ» 2012

УДК 004.42

Структури даних та алгоритми – 2. Складні структури даних та алгоритми: завдання до виконання лабораторних робіт з дисципліни «Структури даних та алгоритми» для студентів напрямку підготовки 6.050102 «Комп'ютерна інженерія» [Електронне видання] / О.І.Марченко. – К.: НТУУ «КПІ», 2012. – 105 с.

Гриф надано вченою радою ФПМ
(протокол № 5 від 24.12.2012 р.)

Навчальне електронне видання містить завдання до виконання лабораторних робіт по кредитному модулю «Структури даних та алгоритми – 2. Складні структури».

Наведені варіанти індивідуальних завдань. До кожної роботи надаються вказівки щодо виконання завдань, тестування програм та оформлення звіту, а також наведені контрольні питання для підготовки до захисту лабораторних робіт. Призначені для студентів напряму підготовки 6.050102 «Комп'ютерна інженерія».

Навчальне електронне видання

Автор: *Марченко Олександр Іванович*, канд.техн.наук, доц.

Відповідальний редактор: *Орлова М.М.,кандидат техн. наук,доцент.*
Рецензент: Заболотня *Т.М., канд. техн. наук, ст.викладач*

За редакцією автора

ЗМІСТ

ВСТУП

Дисципліна "Структури даних та алгоритми" є базовою спеціальноюнормативною дисципліною підготовки фахівців рівня бакалавр з напрямку 050102 «Комп'ютерна інженерія», а також студентів спеціальностей 8.091.501 "Комп'ютерні системи та мережі", 8.091.502 "Системне програмування" та 8.091.503 "Спеціалізовані комп'ютерні системи" і викладається у 1-му та 2-му семестрах.

Ця дисципліна повинна передувати всім програмістським дисциплінам крім початкового курсу програмування, з яким викладається одночасно.

Кредитний модуль «Структури даних та алгоритми – 2. Складні структури» призначений для вивчення складних структур даних, алгоритмів та алгоритмічних методів для розробки обчислювальних і керуючих алгоритмів та відповідних їм структур даних, а також структурної та об'єктно-орієнтованої методологій програмування.

Цикл лабораторних робіт складається з семи робіт і призначений для покриття практичної частини другого кредитного модуля дисципліни «Структури даних та алгоритми».

Роботи дозволяють отримати практичні навички складання складних комбінованих нерекурсивних і рекурсивних алгоритмів, а також використання складних динамічних структур даних, що використовуються в програмуванні.

Завдання до кожної роботи включають:

- постановку завдання та вимоги до алгоритмів та програм, що повинні розробити студенти;
- вказівки до оформлення звіту та тестування розробленого алгоритму і відповідної йому програми;
- контрольні питання для самоконтролю та підготовки до захисту лабораторної роботи;
- варіанти індивідуальних завдань.

1. ЛАБОРАТОРНА РОБОТА №2.1. АЛГОРИТМИ ДВІЙКОВОГО ПОШУКУ

Мета лабораторної роботи

Метою лабораторної роботи №2.1. «Алгоритми двійкового пошуку» є засвоєння теоретичного матеріалу та набуття практикних навичок рішення задачі пошуку заданої категорії елементів за допомогою різних алгоритмів методу двійкового пошуку у двовимірних масивах.

Постановка задачі

1. Написати програму розв'язання задачі пошуку (за варіантом) у двовимірному масиві (матриці) методом двійкового пошуку. Алгоритм двійкового пошуку задається варіантом завдання.
2. Розміри матриці ***m*** та ***n*** взяти самостійно у межах від 7 до 10.
3. При тестуванні програми необхідно підбирати такі вхідні набори початкових значеннь матриці, щоб можна було легко відстежити коректність виконання пошуку і ця коректність була б протестована для всіх можливих випадків. З метою тестування дозволяється використовувати матриці меншого розміру.

Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варіанту.
2. Текст програми, вхідні дані.

3. Тести для налагодження програми і результати, отримані для них на комп'ютері.

Контрольні питання

1. При якій умові можливе використання двійкового пошуку?
2. Написати алгоритм №1 двійкового пошуку в одномірному масиві (векторі)?
3. Написати алгоритм №2 двійкового пошуку в одномірному масиві (векторі)?
4. Чим відрізняється загальна поведінка алгоритмів двійкового пошуку №1 та №2?
5. Чим відрізняється текст (код) алгоритмів двійкового пошуку №1 та №2?
6. Які саме дії алгоритму двійкового пошуку №2 призводять до того, що він знаходить найлівіший елемент масиву, що співпадає з шуканим елементом?
7. Які саме дії алгоритму двійкового пошуку №2 і як треба змінити, щоб він знаходив найправіший елемент масиву, що співпадає з шуканим елементом, замість найлівішого?

Варіанти завдань

*Розміри матриці **m** і **n** взяти самостійно у межах від 7 до 10.*

Варіант № 1

Задано матрицю дійсних чисел ***A***[***m***,***n***]. Окремо у кожному рядку матриці визначити присутність заданого дійсного числа ***X*** і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи кожного рядка окремо впорядковані за незбільшенням.

Варіант № 2

Задано матрицю дійсних чисел ***A***[***m***,***n***]. Окремо у кожному стовпчику матриці визначити присутність заданого дійсного числа ***X*** і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи кожного стовпчика окремо впорядковані за незбільшенням.

Варіант № 3

Задано матрицю дійсних чисел ***A***[***m***,***n***]. Визначити присутність серед усіх елементів матриці заданого дійсного числа ***X*** і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи кожного стовпчика окремо впорядковані за незбільшенням.

Варіант № 4

Задано квадратну матрицю дійсних чисел $A[n,n]$. Визначити присутність у головній діагоналі матриці заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи цієї діагоналі впорядковані за незбільшенням..

Варіант № 5

Задано квадратну матрицю дійсних чисел $A[n,n]$. Визначити присутність у побічній діагоналі матриці заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи цієї діагоналі впорядковані за незбільшенням..

Варіант № 6

Задано матрицю дійсних чисел $A[m,n]$. Окремо у першому рядку і останньому стовпчику матриці визначити присутність заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи цього рядка і стовпчика впорядковані за незбільшенням.

Варіант № 7

Задано матрицю дійсних чисел $A[m,n]$. Окремо у останньому рядку і першому стовпчику матриці визначити присутність заданого дійсного числа X і його місцезнаходження (координати)

методом двійкового пошуку (Алгоритм №1), якщо елементи цього рядка і стовпчика впорядковані за незбільшенням.

Варіант № 8

Задано матрицю дійсних чисел $A[m,n]$. Окремо у кожному рядку матриці визначити присутність заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи кожного рядка окремо впорядковані за незменшенням.

Варіант № 9

Задано матрицю дійсних чисел $A[m,n]$. Окремо у кожному стовпчику матриці визначити присутність заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи кожного стовпчика окремо впорядковані за незменшенням.

Варіант № 10

Задано матрицю дійсних чисел $A[m,n]$. Визначити присутність серед усіх елементів матриці заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи кожного рядка окремо впорядковані за незменшенням.

Варіант № 11

Задано матрицю дійсних чисел $A[m,n]$. Визначити присутність серед усіх елементів матриці заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи кожного стовпчика окремо впорядковані за незменшенням.

Варіант № 12

Задано квадратну матрицю дійсних чисел $A[n,n]$. Визначити присутність у побічній діагоналі матриці заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи цієї діагоналі впорядковані за незменшенням.

Варіант № 13

Задано квадратну матрицю дійсних чисел $A[n,n]$. Визначити присутність у головній діагоналі матриці заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи цієї діагоналі впорядковані за незменшенням.

Варіант № 14

Задано матрицю дійсних чисел $A[m,n]$. Окремо у першому рядку і останньому стовпчику визначити присутність заданого дійсного числа X і його місцезнаходження (координати) методом

двійкового пошуку (Алгоритм №2), якщо елементи цього рядка і стовпчика впорядковані за незменшенням.

Варіант № 15

Задано матрицю дійсних чисел $A[m,n]$. Окремо у останньому рядку і першому стовпчику визначити присутність заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи цього рядка і стовпчика впорядковані за незменшенням.

Варіант № 16

Задано матрицю дійсних чисел $A[m,n]$. Окремо у кожному рядку матриці визначити присутність будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи кожного рядка окремо впорядковані за незбільшенням.

Варіант № 17

Задано матрицю дійсних чисел $A[m,n]$. Окремо у кожному стовпчику матриці визначити присутність будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи кожного стовпчика окремо впорядковані за незбільшенням.

Варіант № 18

Задано матрицю дійсних чисел $A[m,n]$. Визначити присутність серед усіх елементів матриці будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи кожного рядка окремо впорядковані за незбільшенням.

Варіант № 19

Задано матрицю дійсних чисел $A[m,n]$. Визначити присутність серед усіх елементів матриці будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи кожного стовпчика окремо впорядковані за незбільшенням.

Варіант № 20

Задано квадратну матрицю дійсних чисел $A[n,n]$. Визначити присутність у головній діагоналі матриці будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи цієї діагоналі впорядковані за незбільшенням.

Варіант № 21

Задано квадратну матрицю дійсних чисел $A[n,n]$. Визначити присутність у побічній діагоналі матриці будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом

двійкового пошуку (Алгоритм №1), якщо елементи цієї діагоналі впорядковані за незбільшенням.

Варіант № 22

Задано матрицю дійсних чисел $A[m,n]$. Окремо у першому рядку і останньому стовпчику визначити присутність будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи цього рядка і стовпчика впорядковані за незбільшенням.

Варіант № 23

Задано матрицю дійсних чисел $A[m,n]$. Окремо у останньому рядку і першому стовпчику визначити присутність будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи цього рядка і стовпчика впорядковані за незбільшенням.

Варіант № 24

Задано матрицю дійсних чисел $A[m,n]$. Окремо у кожному рядку матриці визначити присутність будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи кожного рядка окремо впорядковані за незменшенням.

Варіант № 25

Задано матрицю дійсних чисел ***A***[***m***,***n***]. Окремо у кожному стовпчику матриці визначити присутність будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи кожного стовпчика окремо впорядковані за незменшенням.

Варіант № 26

Задано матрицю дійсних чисел ***A***[***m***,***n***]. Визначити присутність серед усіх елементів матриці будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи кожного рядка окремо впорядковані за незменшенням.

Варіант № 27

Задано матрицю дійсних чисел ***A***[***m***,***n***]. Визначити присутність серед усіх елементів матриці будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи кожного стовпчика окремо впорядковані за незменшенням.

Варіант № 28

Задано квадратну матрицю дійсних чисел ***A***[***n***,***n***]. Визначити присутність у головній діагоналі матриці будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом

двійкового пошуку (Алгоритм №2), якщо елементи цієї діагоналі впорядковані за незменшенням.

Варіант № 29

Задано квадратну матрицю дійсних чисел $A[n,n]$. Визначити присутність у побічній діагоналі матриці будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи цієї діагоналі впорядковані за незменшенням.

Варіант № 30

Задано матрицю дійсних чисел $A[m,n]$. Окремо у останньому рядку і першому стовпчику визначити присутність будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи цього рядка і стовпчика впорядковані за незменшенням.

2. ЛАБОРАТОРНА РОБОТА №2.2. АЛГОРИТМИ СОРТУВАННЯ

Мета лабораторної роботи

Метою лабораторної роботи №2.2. є засвоєння теоретичного матеріалу та набуття практичних навичок рішення задачі сортування заданої категорії елементів за допомогою різних алгоритмів сортування у двовимірних масивах.

Постановка задачі

1. Задано двовимірний масив (матрицю) цілих чисел ***A***[***m***,***n***] або ***A***[***n***,***n***], де ***m*** та ***n***. – натуральні числа (константи), що визначають розміри двовимірного масиву. Виконати сортування цього масиву або заданої за варіантом його частини у заданому порядку заданим алгоритмом (методом).

Сортування повинно бути виконано безпосередньо у двовимірному масиві «на тому ж місці», тобто без перезаписування масиву та/або його будь-якої частини до інших одно- або двовимірних масивів, а також без використання спискових структур даних.

2. Розміри матриці ***m*** та ***n*** взяти самостійно у межах від 7 до 10.

3. При тестуванні програми необхідно підбирати такі вхідні набори початкових значеннь матриці, щоб можна було легко відстежити коректність виконання сортування і ця коректність була

б протестована для всіх можливих випадків. З метою тестування дозволяється використовувати матриці меншого розміру.

Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варіанту.
2. Текст програми.
3. Тестування програми. При тестуванні програми необхідно перевірити коректність її роботи для трьох випадків початкового стану масиву або тієї його частини, яка сортується:
   - вже відсортований початковий стан масиву;
   - невідсортований початковий стан масиву (масив заповнений випадковими числами);
   - обернено відсортований (до заданого за завданням) початковий стан масиву.
4. В якості результату роздрукувати дані тестування для всіх трьох випадків початкового стану масиву або тієї його частини, яка сортується.

Контрольні питання

1. Класифікація методів та алгоритмів сортування.
2. Принцип та схема алгоритму №1 методу вставки (з лінійним пошуком зліва).
3. Принцип та схема алгоритму №2 методу вставки (з лінійним пошуком справа).

4. Принцип та схема алгоритму №3 методу вставки (з лінійним пошуком справа з використанням бар'єру).

5. Принцип та схема алгоритму №4 методу вставки (з двійковим пошуком).

6. Принцип та схема методу вибору.

7. Принцип та схема алгоритму №1 методу обмінів («бульбашкове сортування»).

8. Принцип та схема алгоритму №2 методу обмінів («бульбашкове сортування» з використанням «прапорця»).

9. Принцип та схема алгоритму №3 методу обмінів («бульбашкове сортування» із запам'ятовуванням місця останньої перестановки).

10. Принцип та схема алгоритму №4 методу обмінів («шейкерне сортування»).

11. Принцип та схема методу Шелла.

12. Принцип та схема методу швидкого сортування (методом Хоара)

Варіанти індивідуальних завдань

Варіант № 1

Задано двовимірний масив (матрицю) цілих чисел $A[m,n]$. Відсортувати елементи першого рядка масиву, що стоять на непарних позиціях, алгоритмом №1 методу вставки (з лінійним пошуком зліва) за незменшенням.

Варіант № 2

Задано двовимірний масив (матрицю) цілих чисел $A[m,n]$. Відсортувати окремо кожен рядок масиву алгоритмом №2 методу вставки (з лінійним пошуком справа) за незбільшенням.

Варіант № 3

Задано двовимірний масив (матрицю) цілих чисел $A[m,n]$. Відсортувати окремо кожен стовпчик масиву алгоритмом №3 методу вставки (з лінійним пошуком справа з використанням бар'єру) за незменшенням.

Варіант № 4

Задано квадратну двовимірний масив (матрицю) цілих чисел $A[n,n]$. Відсортувати головну діагональ масиву алгоритмом №4 методу вставки (з двійковим пошуком) за незбільшенням.

Варіант № 5

Задано квадратну двовимірний масив (матрицю) цілих чисел $A[n,n]$. Відсортувати побічну діагональ масиву алгоритмом №3 методу обмінів («бульбашкове сортування» із запам'ятовуванням місця останньої перестановки) за незменшенням.

Варіант № 6

Задано двовимірний масив (матрицю) цілих чисел $A[m,n]$. Відсортувати елементи першого стовпчика масиву, що стоять на непарних позиціях, методом вибору за незбільшенням.

Варіант № 7

Задано двовимірний масив (матрицю) цілих чисел $A[m,n]$. Відсортувати окремо кожен рядок масиву методом швидкого сортування (методом Хоара) за незменшенням.

Варіант № 8

Задано двовимірний масив (матрицю) цілих чисел $A[m,n]$. Відсортувати окремо кожен стовпчик масиву алгоритмом №4 методу обмінів («шейкерне сортування») за незбільшенням.

Варіант № 9

Задано квадратну двовимірний масив (матрицю) цілих чисел $A[n,n]$. Відсортувати головну діагональ масиву методом Шелла за незменшенням.

Варіант № 10

Задано квадратну двовимірний масив (матрицю) цілих чисел ***A***[***n***,***n***]. Відсортувати побічну діагональ масиву алгоритмом №2 методу обмінів («бульбашкове сортування» з використанням «прапорця») за незбільшенням.

Варіант № 11

Задано двовимірний масив (матрицю) цілих чисел ***A***[***m***,***n***]. Відсортувати елементи останнього рядка масиву, що стоять на парних позиціях, методом вибору за незменшенням.

Варіант № 12

Задано двовимірний масив (матрицю) цілих чисел ***A***[***m***,***n***]. Відсортувати окремо кожен рядок масиву алгоритмом №3 методу вставки (з лінійним пошуком справа з використанням бар'єру) за незбільшенням.

Варіант № 13

Задано двовимірний масив (матрицю) цілих чисел ***A***[***m***,***n***]. Відсортувати окремо кожен стовпчик масиву алгоритмом №4 методу вставки (з двійковим пошуком) за незменшенням.

Варіант № 14

Задано квадратну двовимірний масив (матрицю) цілих чисел ***A***[***n***,***n***]. Відсортувати головну діагональ масиву алгоритмом №4 методу обмінів («шейкерне сортування») за незбільшенням.

Варіант № 15

Задано квадратну двовимірний масив (матрицю) цілих чисел ***A***[***n***,***n***]. Відсортувати побічну діагональ масиву алгоритмом №1 методу вставки (з лінійним пошуком зліва) за незменшенням.

Варіант № 16

Задано двовимірний масив (матрицю) цілих чисел ***A***[***m***,***n***]. Відсортувати елементи останнього стовпчика масиву, що стоять на непарних позиціях, алгоритмом №2 методу обмінів («бульбашкове сортування» з використанням «прапорця») за незбільшенням.

Варіант № 17

Задано двовимірний масив (матрицю) цілих чисел ***A***[***m***,***n***]. Відсортувати окремо кожен рядок масиву методом Шелла за незменшенням.

Варіант № 18

Задано двовимірний масив (матрицю) цілих чисел ***A***[***m***,***n***]. Відсортувати окремо кожен стовпчик масиву методом швидкого сортування (методом Хоара) за незбільшенням.

Варіант № 19

Задано квадратну двовимірний масив (матрицю) цілих чисел ***A***[***n***,***n***]. Відсортувати головну діагональ масиву алгоритмом №2 методу вставки (з лінійним пошуком справа) за незменшенням.

Варіант № 20

Задано квадратну двовимірний масив (матрицю) цілих чисел ***A***[***n***,***n***]. Відсортувати побічну діагональ масиву методом вибору за незбільшенням.

Варіант № 21

Задано двовимірний масив (матрицю) цілих чисел ***A***[***m***,***n***]. Відсортувати елементи першого рядка масиву, що стоять на парних позиціях, алгоритмом №2 методу обмінів («бульбашкове сортування» з використанням «прапорця») за незменшенням.

Варіант № 22

Задано двовимірний масив (матрицю) цілих чисел ***A***[***m***,***n***]. Відсортувати окремо кожен рядок масиву алгоритмом №4 методу вставки (з двійковим пошуком) за незбільшенням.

Варіант № 23

Задано двовимірний масив (матрицю) цілих чисел ***A***[***m***,***n***]. Відсортувати окремо кожен стовпчик масиву алгоритмом №1 методу вставки (з лінійним пошуком зліва) за незменшенням.

Варіант № 24

Задано квадратну двовимірний масив (матрицю) цілих чисел ***A***[***n***,***n***]. Відсортувати головну діагональ масиву алгоритмом №3 методу вставки (з лінійним пошуком справа з використанням бар’єру) за незбільшенням.

Варіант № 25

Задано квадратну двовимірний масив (матрицю) цілих чисел $A[n,n]$. Відсортувати побічну діагональ масиву алгоритмом №3 методу обмінів («бульбашкове сортування» із запам'ятовуванням місця останньої перестановки) за незменшенням.

Варіант № 26

Задано двовимірний масив (матрицю) цілих чисел $A[m,n]$. Відсортувати елементи останнього стовпчика масиву, що стоять на парних позиціях, алгоритмом №3 методу обмінів («бульбашкове сортування» із запам'ятовуванням місця останньої перестановки) за незбільшенням.

Варіант № 27

Задано двовимірний масив (матрицю) цілих чисел $A[m,n]$. Відсортувати окремо кожен рядок масиву алгоритмом №4 методу обмінів («шейкерне сортування») за незменшенням.

Варіант № 28

Задано двовимірний масив (матрицю) цілих чисел $A[m,n]$. Відсортувати окремо кожен стовпчик масиву алгоритмом №2 методу вставки (з лінійним пошуком справа) за незбільшенням.

Варіант № 29

Задано квадратну двовимірний масив (матрицю) цілих чисел ***A***[***n***,***n***]. Відсортувати побічну діагональ масиву методом Шелла за незбільшенням.

Варіант № 30

Задано квадратну двовимірний масив (матрицю) цілих чисел ***A***[***n***,***n***]. Відсортувати головну діагональ масиву методом швидкого сортування (методом Хоара) за незменшенням.

3. ЛАБОРАТОРНА РОБОТА №2.3. РЕКУРСИВНІ АЛГОРИТМИ

Мета лабораторної роботи

Метою лабораторної роботи №2.3. є засвоєння теоретичного матеріалу та набуття практичного досвіду створення рекурсивних алгоритмів та написання відповідних їм програм.

Постановка задачі

Дано натуральне число ***n***. Знайти суму перших ***n*** членів ряду чисел, заданого рекурентною формулою.

Розв'язати задачу трьома способами (написати три програми):

1) в програмі використати рекурсивну процедуру або функцію, яка виконує обчислення і членів ряду, і суми на рекурсивному спуску;
2) в програмі використати рекурсивну процедуру або функцію, яка виконує обчислення і членів ряду, і суми на рекурсивному поверненні;
3) в програмі використати рекурсивну процедуру або функцію, яка виконує обчислення членів ряду на рекурсивному спуску, а обчислення суми на рекурсивному поверненні.

Програми повинні працювати коректно для довільного натурального ***n*** включно з ***n*** = 1.

Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варіанту.
2. Текст усіх трьох програм.
3. Тестування програм. З метою тестування потрібно написати циклічний варіант рішення задачі, а також виконати обчислення заданої формули на калькуляторі.
4. В якості результату роздрукувати дані тестування (для n=5) як циклічною програмою, так і обчислення на калькуляторі, та розв'язку задачі усіма трьома рекурсивними програмами. Результати обчислень усіма способами повинні співпадати.

Контрольні питання

1. Визначення рекурсивного об'єкту.
2. Визначення глибини та поточного рівня рекурсії.
3. Форма виконання рекурсивних дій на рекурсивному спуску.
4. Форма виконання рекурсивних дій на рекурсивному поверненні.
5. Форма виконання рекурсивних дій на як рекурсивному спуску, так і на рекурсивному поверненні.

Варіанти індивідуальних завдань

Варіант № 1

$F_1 = 1,$

для $\forall i > 1$ $F_i = \cos(F_{i-1}) \cdot \sqrt{i} / 5$

Варіант № 2

$F_1 = 1,$

для $\forall i > 1$ $F_i = F_{i-1} + 2 \cdot \sin(i)$

Варіант № 3

$F_1 = 1,$

для $\forall i > 1$ $F_i = \sqrt{F_{i-1}} \cdot \ln(i) / 7$

Варіант № 4

$F_1 = 1,$

для $\forall i > 1$ $F_i = 2 \cdot F_{i-1} - \cos(i)$

Варіант № 5

$F_1 = 1,$

для $\forall i > 1$ $F_i = \sqrt{F_{i-1} \cdot i} / 4$

Варіант № 6

$F_1 = 1,$

для $\forall i > 1$ $F_i = 2 \cdot \cos(F_{i-1}) + i$

Варіант № 7

$F_1 = 1,$

для $\forall i > 1$ $F_i = F_{i-1} \cdot \ln(i) / 8$

Варіант № 8

$F_1 = 1,$
для $\forall i > 1$ $F_i = \sin(F_{i-1} + \cos(i))$

Варіант № 9

$F_1 = 3,$
для $\forall i > 1$ $F_i = \ln(F_{i-1}) + i / 3$

Варіант № 10

$F_1 = 1,$
для $\forall i > 1$ $F_i = F_{i-1} + e^i / 100$

Варіант № 11

$F_1 = 1,$
для $\forall i > 1$ $F_i = \ln(F_{i-1} \cdot i) + 9$

Варіант № 12

$F_1 = 1,$
для $\forall i > 1$ $F_i = \ln(F_{i-1}+3) + \sqrt{i}$

Варіант № 13

$F_1 = 1,$
для $\forall i > 1$ $F_i = \cos(F_{i-1}) \cdot 100 / i$

Варіант № 14

$F_1 = 1,$
для $\forall i > 1$ $F_i = 3 \cdot \sin(F_{i-1}) + \ln(i)$

Варіант № 15

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = \sqrt{F_{i-1} \ln(i+1)}$$

Варіант № 16

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = 2 \cdot \sqrt{F_{i-1}} + i$$

Варіант № 17

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = F_{i-1} \cdot \sin(\sqrt{i / 2})$$

Варіант № 18

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = F_{i-1} + \sqrt{5 \cdot |\sin(i)|}$$

Варіант № 19

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = 10 / i \cdot \sqrt{3 \cdot F_{i-1}}$$

Варіант № 20

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = \cos(i) + 2 \cdot \sqrt{|F_{i-1}|}$$

Варіант № 21

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = \sqrt{F_{i-1} \cdot i} \ / \ln(i+1)$$

Варіант № 22

$F_1 = 1,$

для $\forall i > 1$ $F_i = \sin(i) - 2 \cdot \cos(F_{i-1})$

Варіант № 23

$F_1 = 1,$

для $\forall i > 1$ $F_i = 4 \cdot \sin(F_{i-1} \cdot \ln(i))$

Варіант № 24

$F_1 = 1,$

для $\forall i > 1$ $F_i = i - \sin(F_{i-1} + i)$

Варіант № 25

$F_1 = 1,$

для $\forall i > 1$ $F_i = \cos(|F_{i-1}| + i^2)$

Варіант № 26

$F_1 = 1,$

для $\forall i > 1$ $F_i = F_{i-1} + \cos (e^i - i)$

Варіант № 27

$F_1 = 1,$

для $\forall i > 1$ $F_i = \ln(|F_{i-1}| \cdot i) + \sin(i)$

Варіант № 28

$F_1 = 1,$

для $\forall i > 1$ $F_i = \cos(i \cdot F_{i-1}) + \sqrt{i}$

Варіант № 29

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = \sqrt{128 \cdot F_{i-1} - \ln(i)}$$

Варіант № 30

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = \ln(i) \cdot \sqrt{5 + F_{i-1}}$$

Варіант № 31

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = \sin\left(\sqrt{|F_{i-1}|}\right) + 9 \cdot i$$

Варіант № 32

$$F_1 = 1,$$
$$\text{для } \forall i > 1 \quad F_i = \sqrt{3 \cdot F_{i-1} + \ln(i)}$$

4. ЛАБОРАТОРНА РОБОТА №2.4.
МОДУЛІ

Мета лабораторної роботи

Метою лабораторної роботи №2.4. є засвоєння теоретичного матеріалу та набуття практичного досвіду використання конструкції «модуль» при створенні багатомодульних програм.

Постановка задачі

1. Згідно заданої схеми взаємозв'язків модулів (**M_i**, **MErr**) і процедур (**S_i** , **Err**) намалювати повну схему взаємозв'язків модулів і процедур, на якій різними типами стрілок позначити напрями імпортування модулів та виклики процедур.

2. Згідно отриманої повної схеми взаємозв'язків модулів і процедур написати програмний код цих модулів та процедур на рівні "заглушок".

3. Процедура **Err** повинна викликатися з усіх інших процедур та функцій.

4. В усіх модулях програми повинні бути доступними спільні (глобальні) структури даних, задані згідно варіанту.

5. Виконати тестування та налагодження програми на комп'ютері.

6. Протокол повідомлень про початок та закінчення роботи процедур та функцій, а також про виконання ініціалізацій них розділів модулів, вивести у текстовий файл.

Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варіанту.

2. Повна схема взаємозв'язків модулів і процедур, на якій різними типами стрілок позначені напрями імпортування модулів та виклики процедур.

3. Текст програми, вхідні дані.

4. Протокол повідомлень про початок та закінчення роботи процедур та функцій, а також про виконання ініціалізацій них розділів модулів.

Контрольні питання

1. Які проблеми виникають при створенні великих програмних систем?

2. Які цілі структурної методології програмування?

3. Які основні принципи структурної методології програмування?

4. У чому полягає суть принципу абстракції?

5. У чому полягає суть принципу формальності?

6. У чому полягає суть принципу «поділяй та володарюй»?

7. У чому полягає суть принципу ієрархічного впорядкування?

8. Що розуміють в програмуванні під терміном «модуль»?

9. Сформулюйте принцип приховування інформації. Хто його автор?

10.Що забезпечує принцип приховування інформації при створенні програмних систем?

11.В чому полягає відмінність області дії змінних у модулі області дії змінних у процедурах?

12.З яких розділів складається модуль?

13.Що і як може бути описане в розділі інтерфейсу?

14.Що і як може бути описане в розділі реалізації?

15.Що і як може бути описане в розділі ініціалізації?

16.Що і як може бути описане в розділі закінчення?

Варіанти індивідуальних завдань

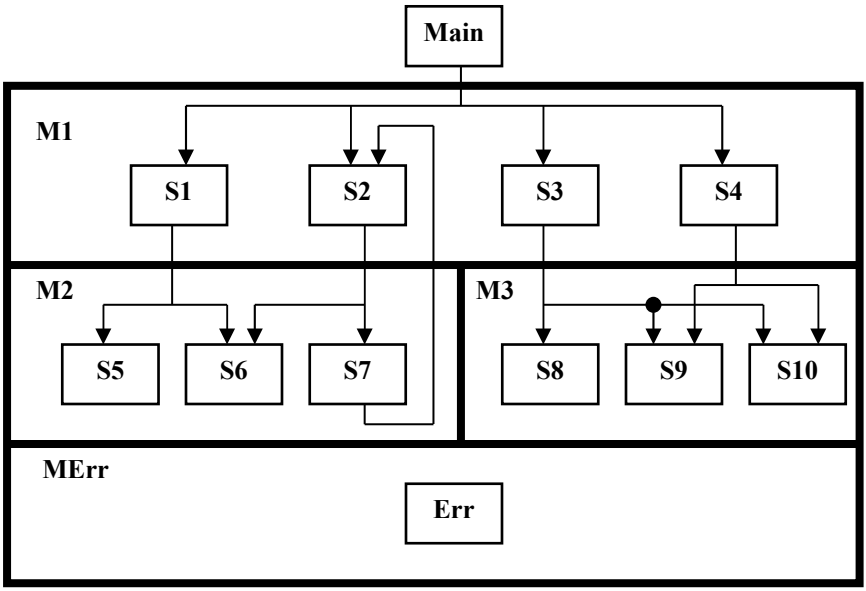
Варіант 1

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg2.

Константи: Cg1 – Cg5.

Змінні: Vg1 – Vg5.



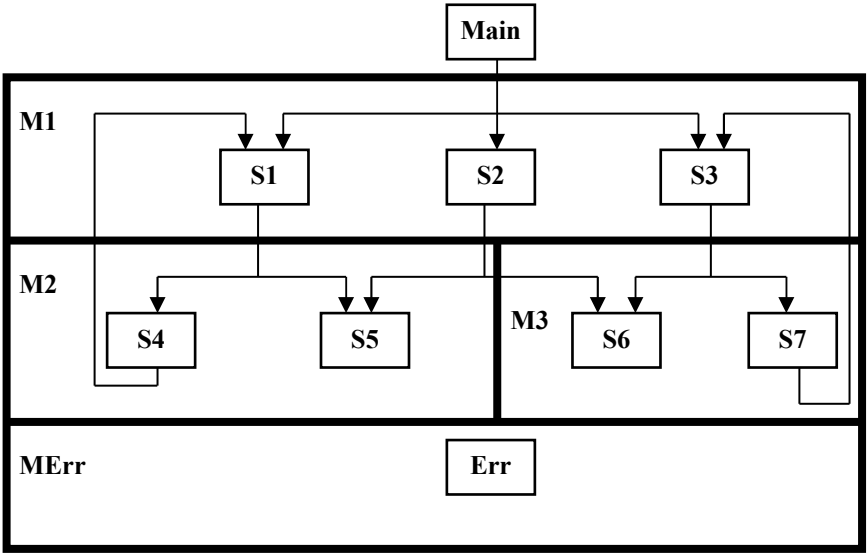
Варіант 2

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg3.

Константи: Cg1 – Cg5.

Змінні: Vg1 – Vg2.



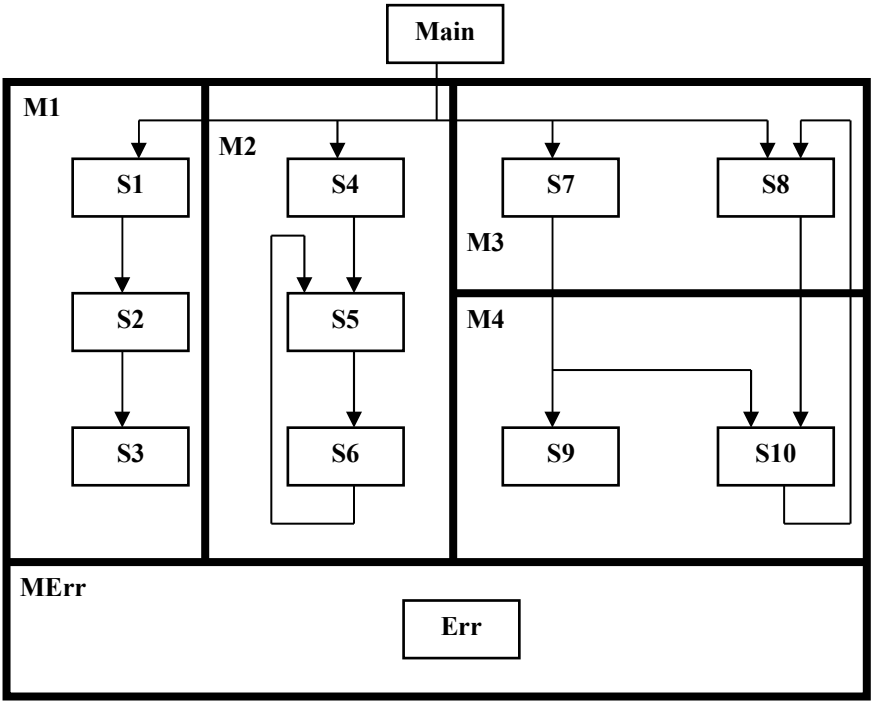
Варіант 3

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg5.

Константи: Cg1 – Cg2.

Змінні: Vg1 – Vg2.



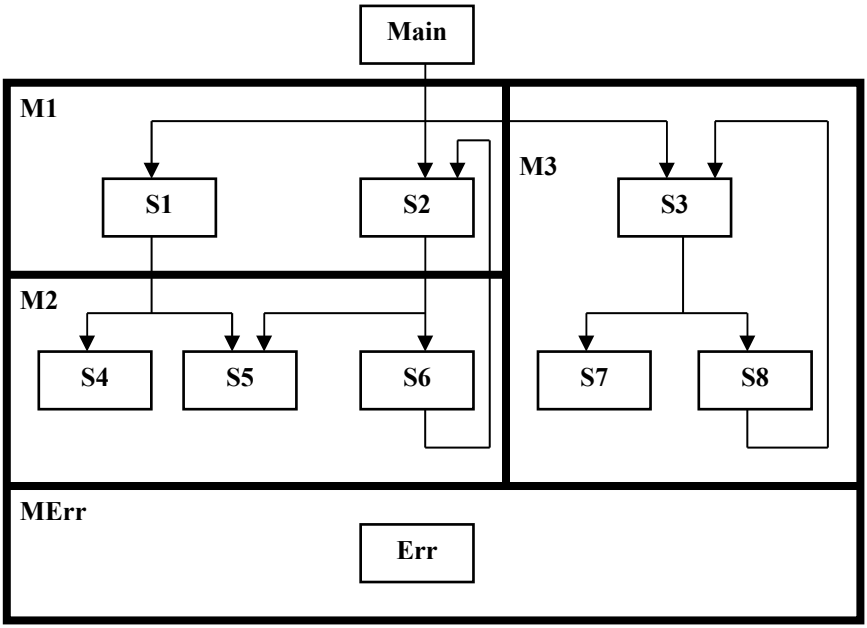
Варіант 4

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg4.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg5.



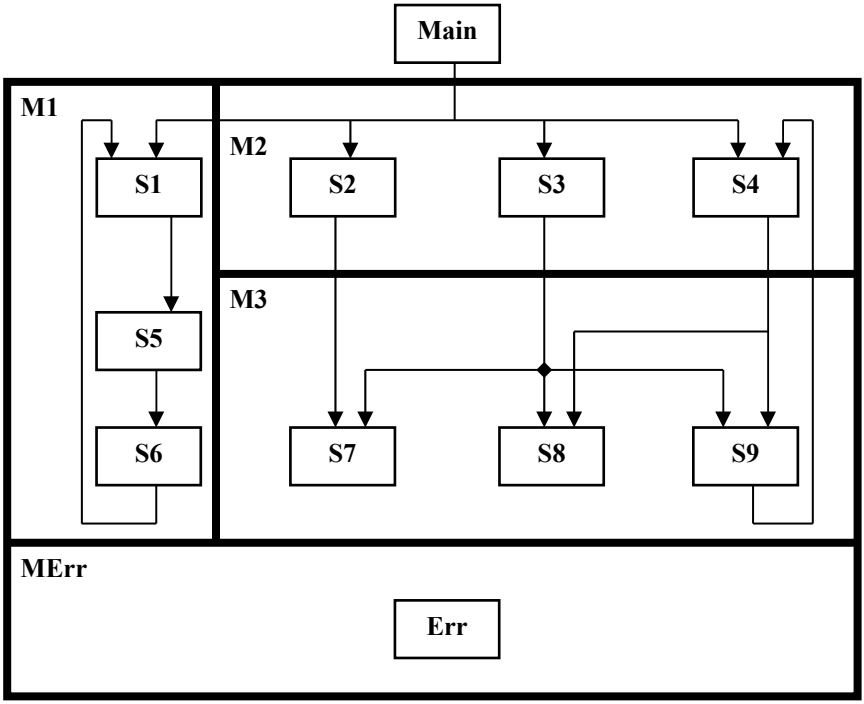
Варіант 5

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg4.

Константи: Cg1 – Cg2.

Змінні: Vg1 – Vg2.



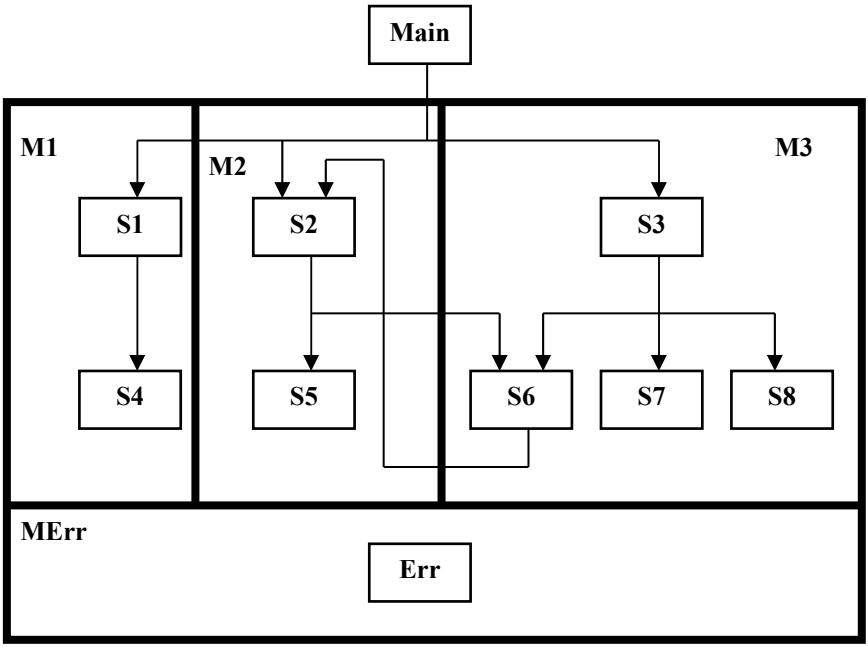
Варіант 6

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg5.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg4.



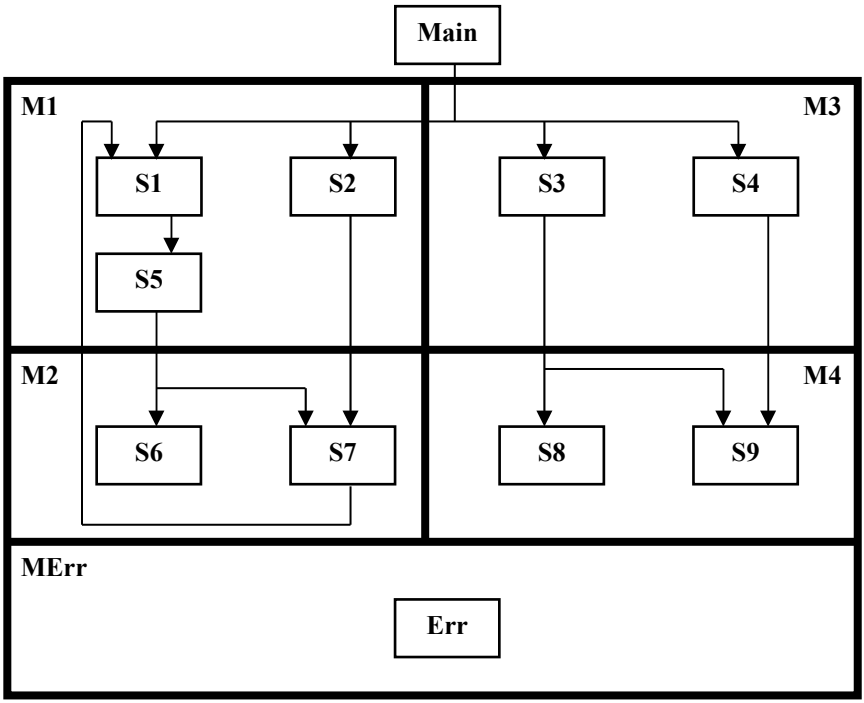
Варіант 7

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg2.

Константи: Cg1 – Cg4.

Змінні: Vg1 – Vg4.



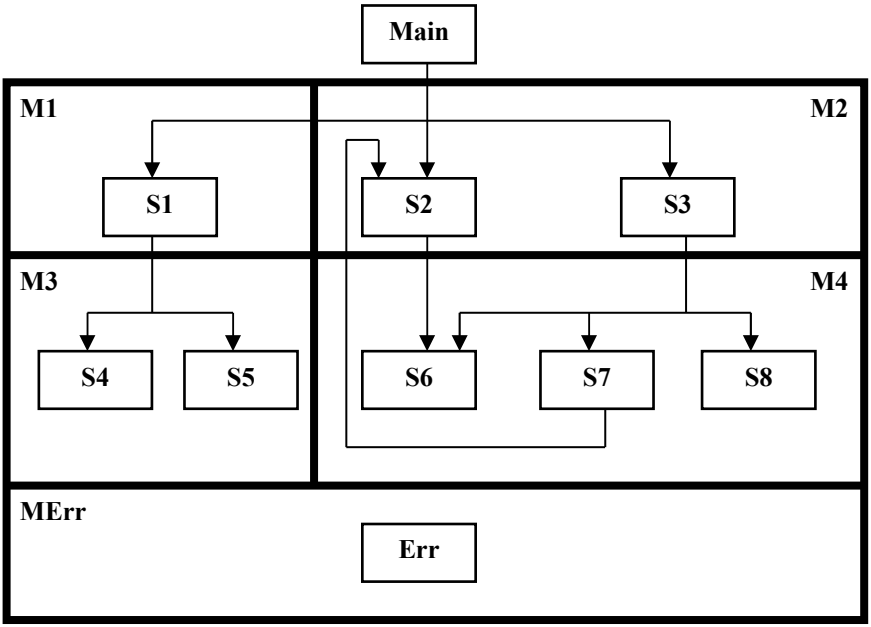
Варіант 8

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg4.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg5.



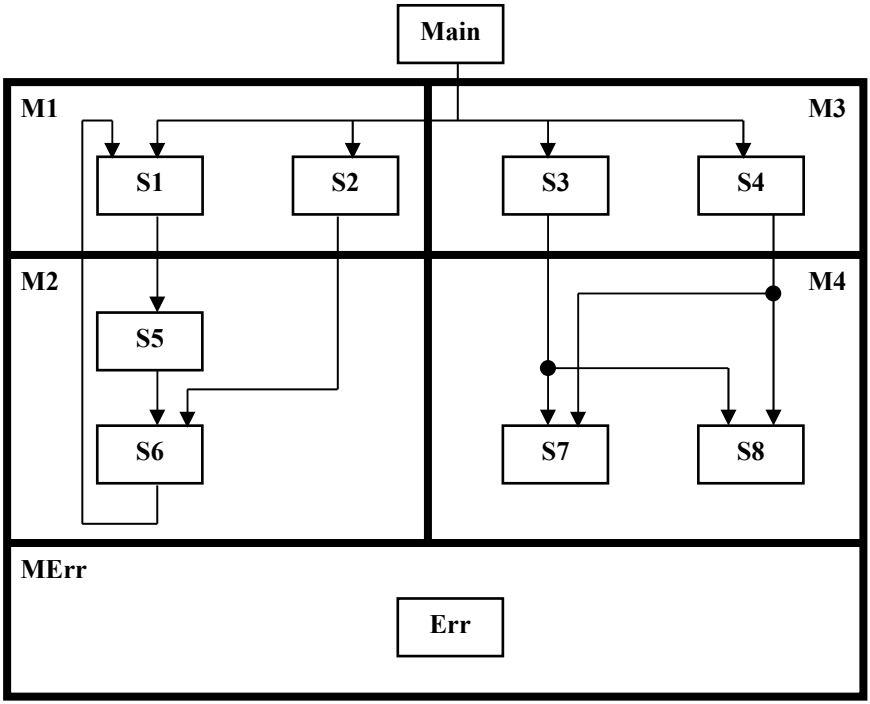
Варіант 9

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg5.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg3.



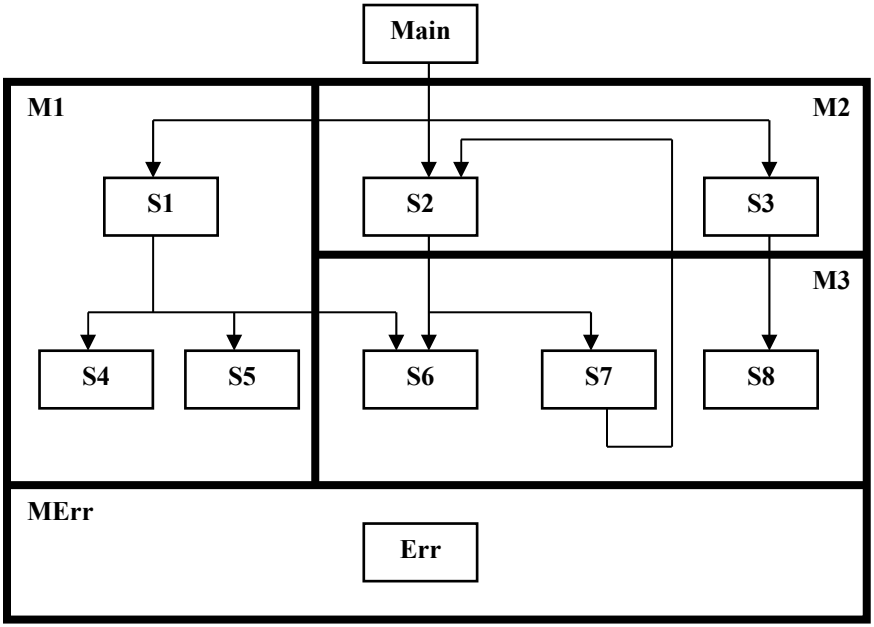
Варіант 10

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg3.

Константи: Cg1 – Cg4.

Змінні: Vg1 – Vg2.



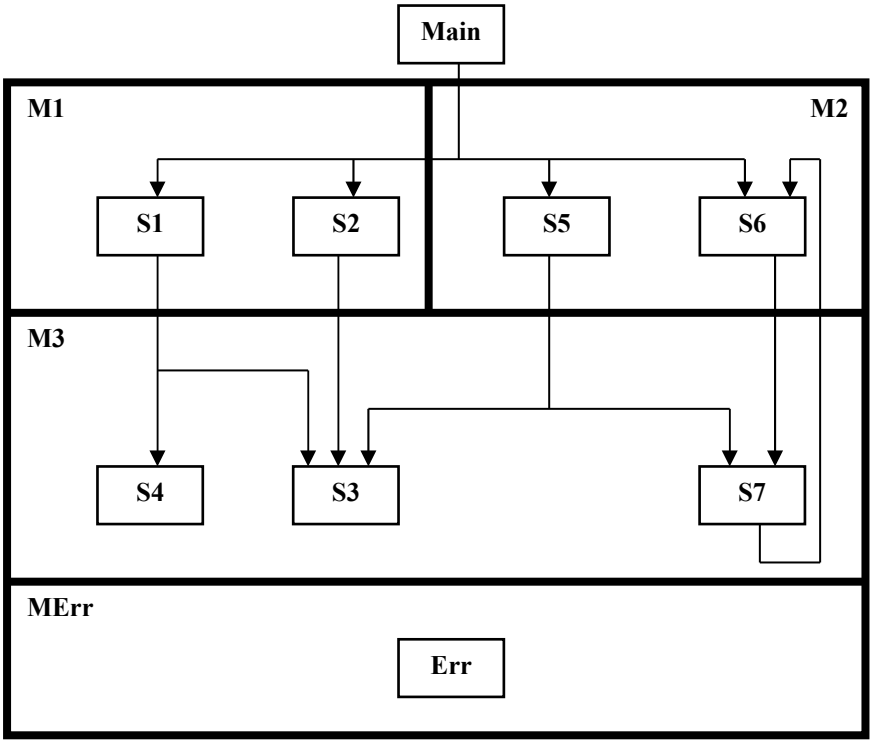
Варіант 11

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg3.

Константи: Cg1 – Cg2.

Змінні: Vg1 – Vg4.



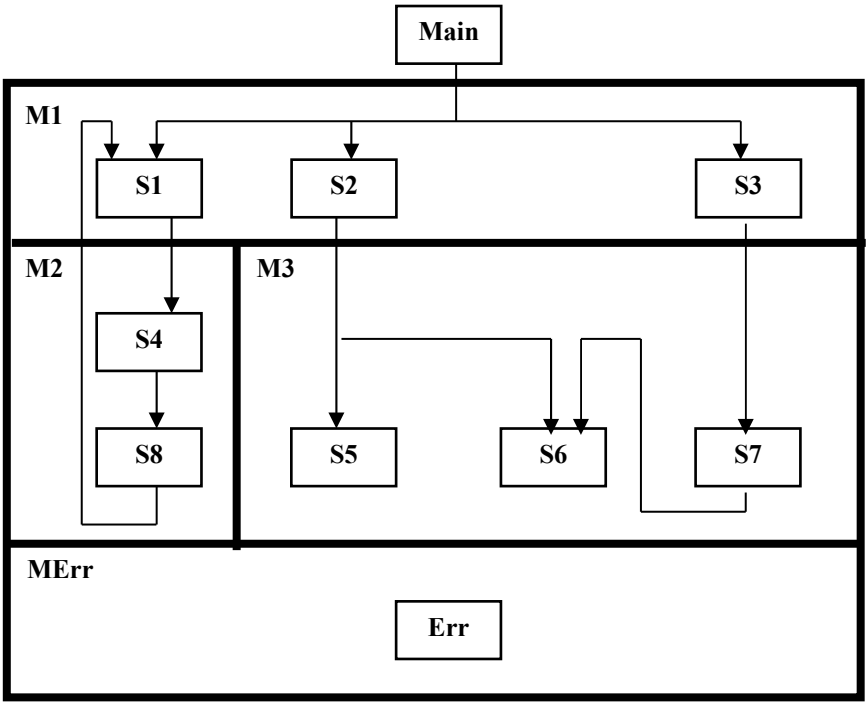
Варіант 12

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg5.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg2.



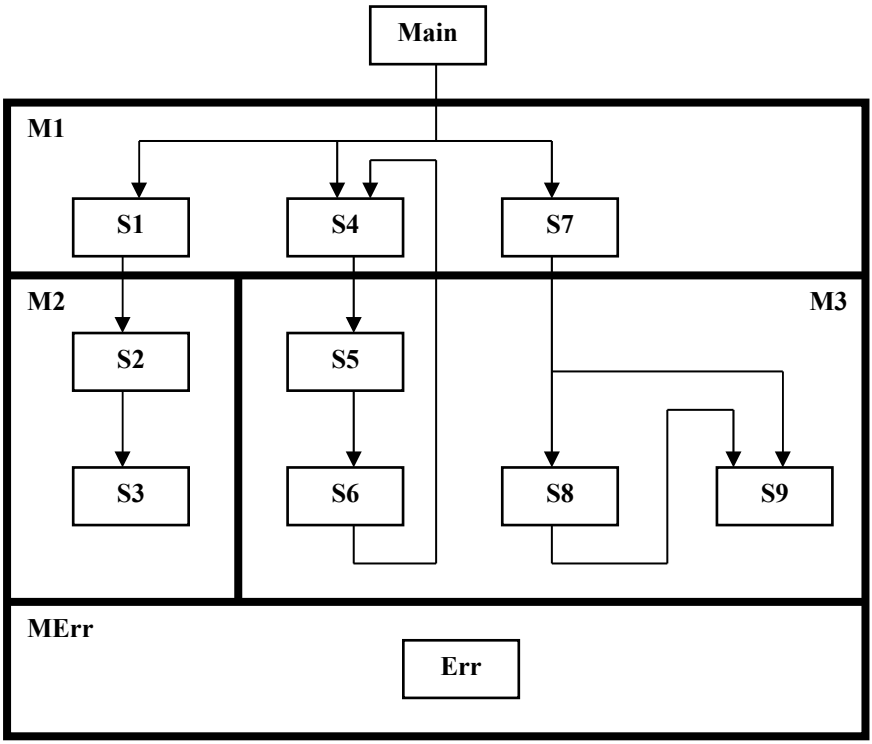
Варіант 13

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg3.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg3.



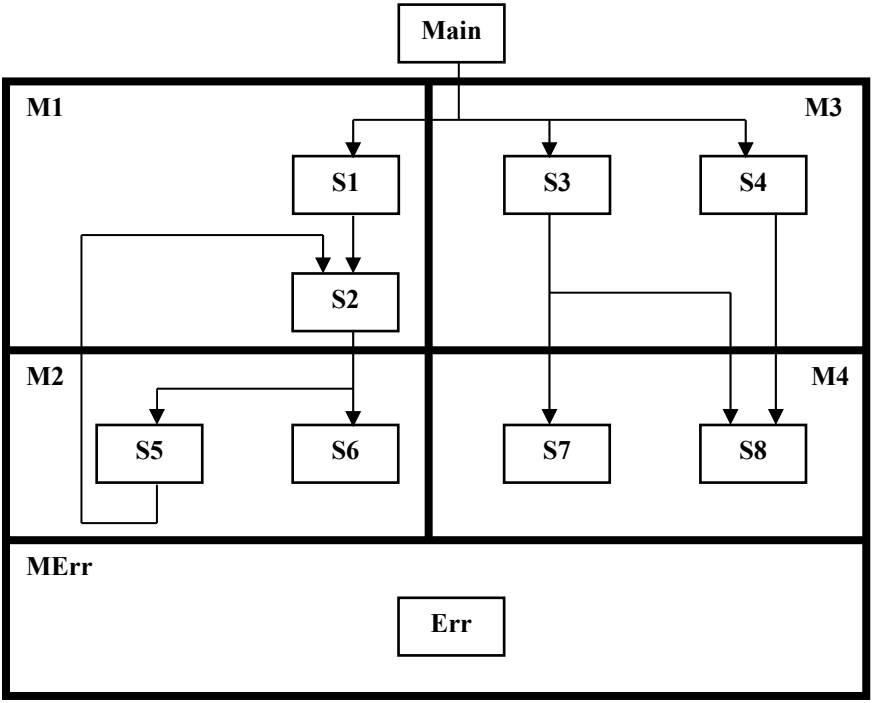
Варіант 14

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg3.

Константи: Cg1 – Cg4.

Змінні: Vg1 – Vg5.



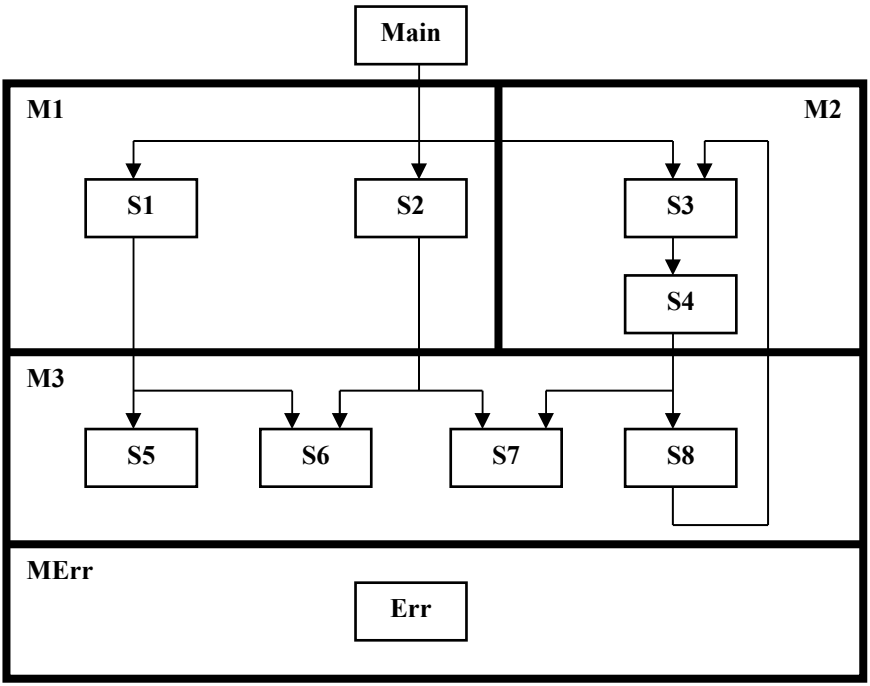
Варіант 15

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg4.

Константи: Cg1 – Cg2.

Змінні: Vg1 – Vg5.



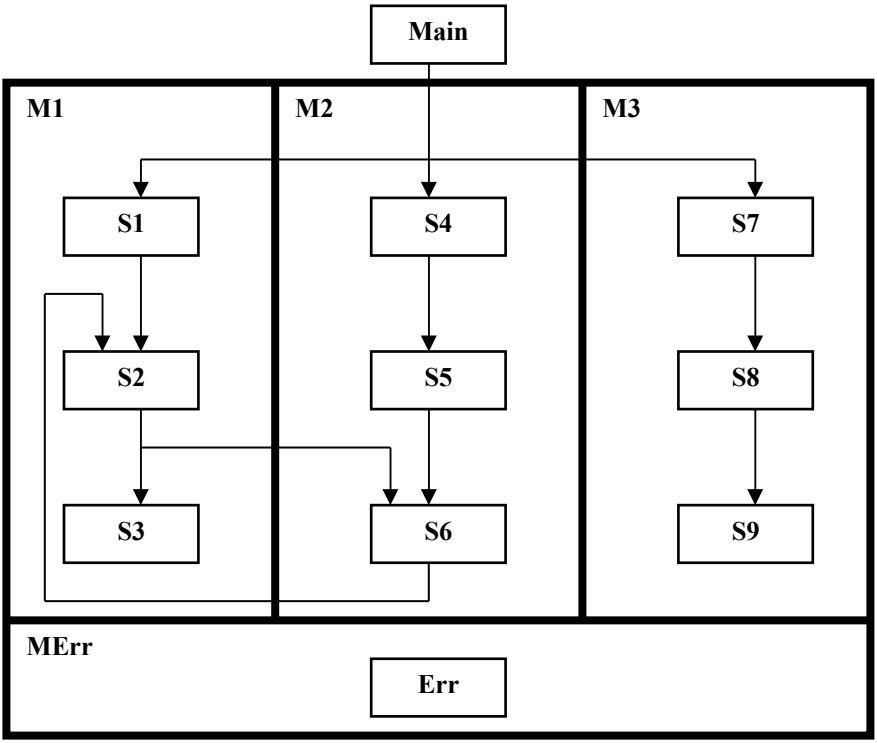
Варіант 16

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg4.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg2.



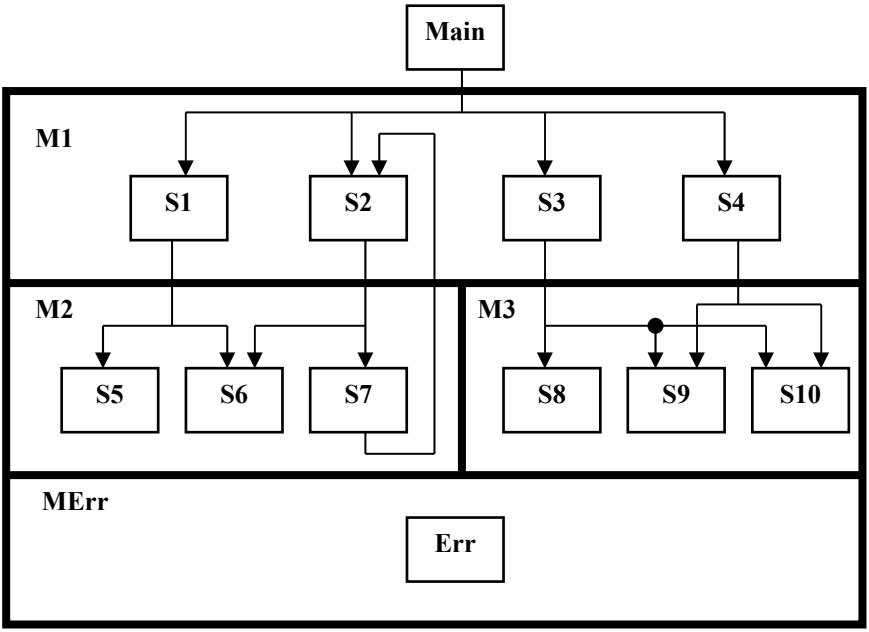
Варіант 17

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg2.

Константи: Cg1 – Cg5.

Змінні: Vg1 – Vg5.



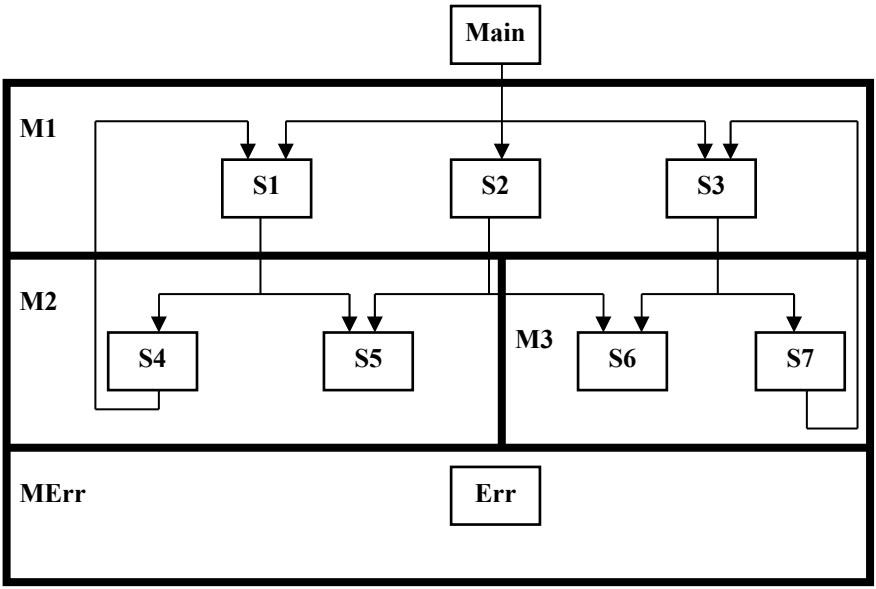
Варіант 18

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg3.

Константи: Cg1 – Cg5.

Змінні: Vg1 – Vg2.



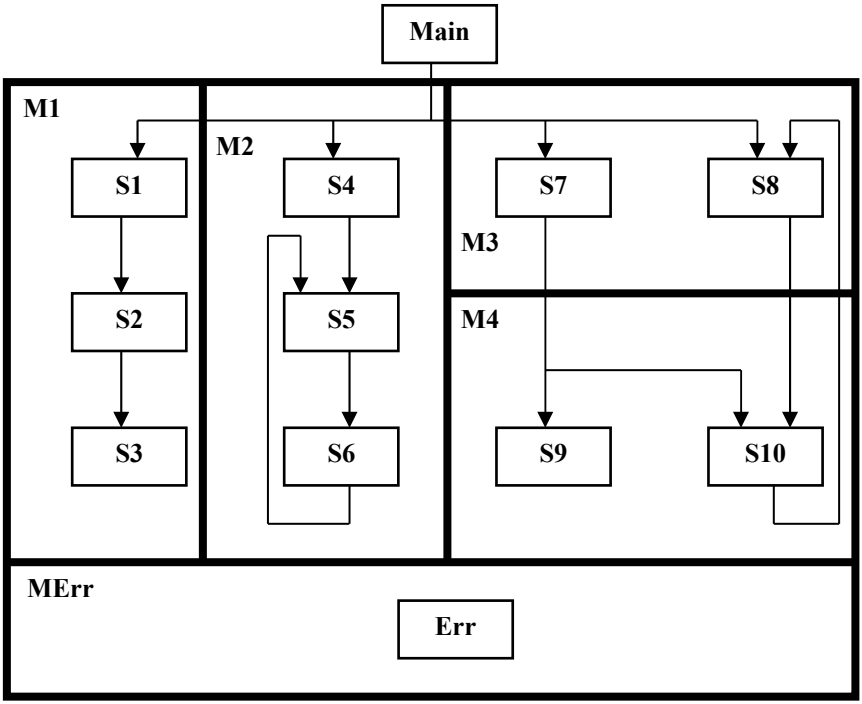
Варіант 19

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg5.

Константи: Cg1 – Cg2.

Змінні: Vg1 – Vg2.



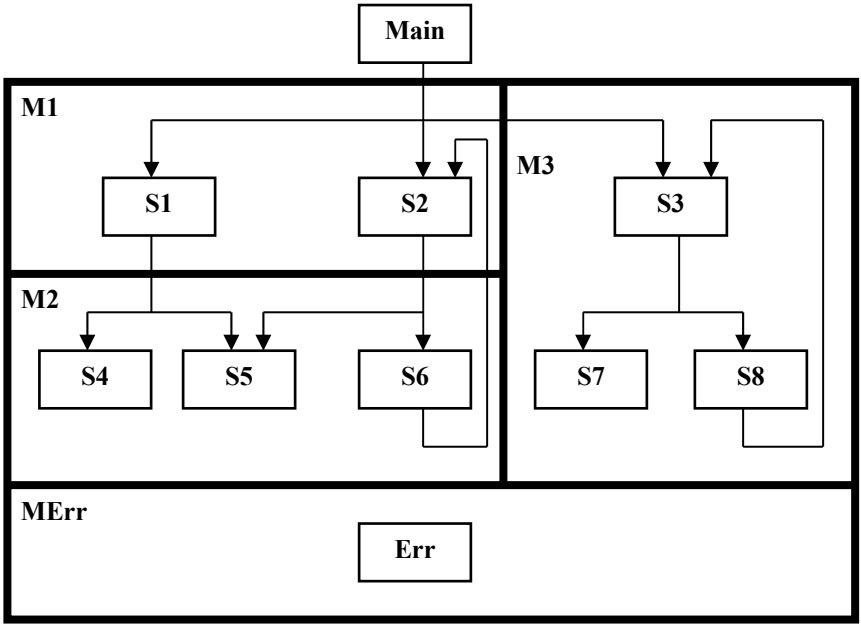
Варіант 20

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg4.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg5.



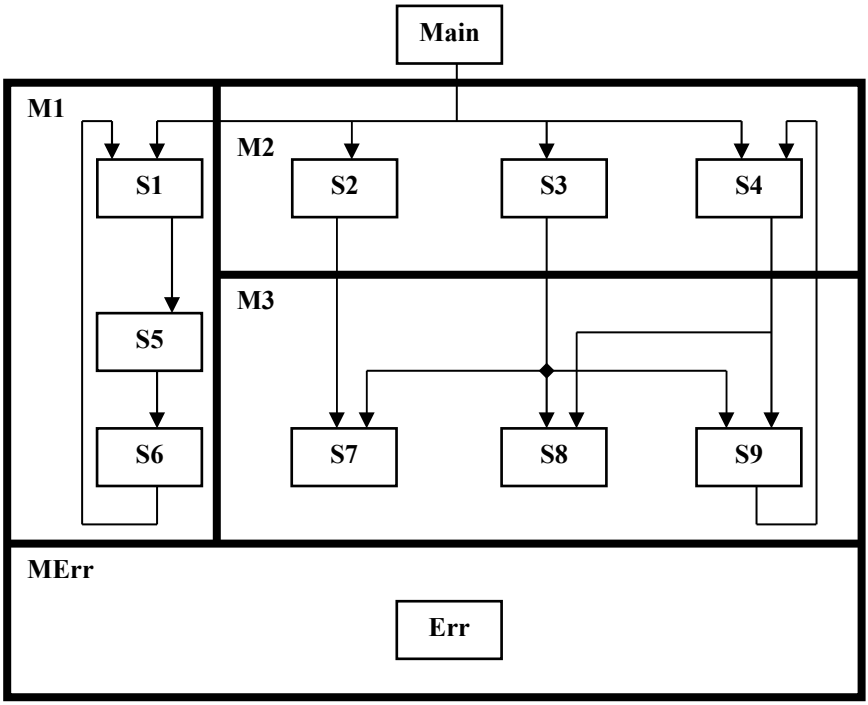
Варіант 21

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg4.

Константи: Cg1 – Cg2.

Змінні: Vg1 – Vg2.



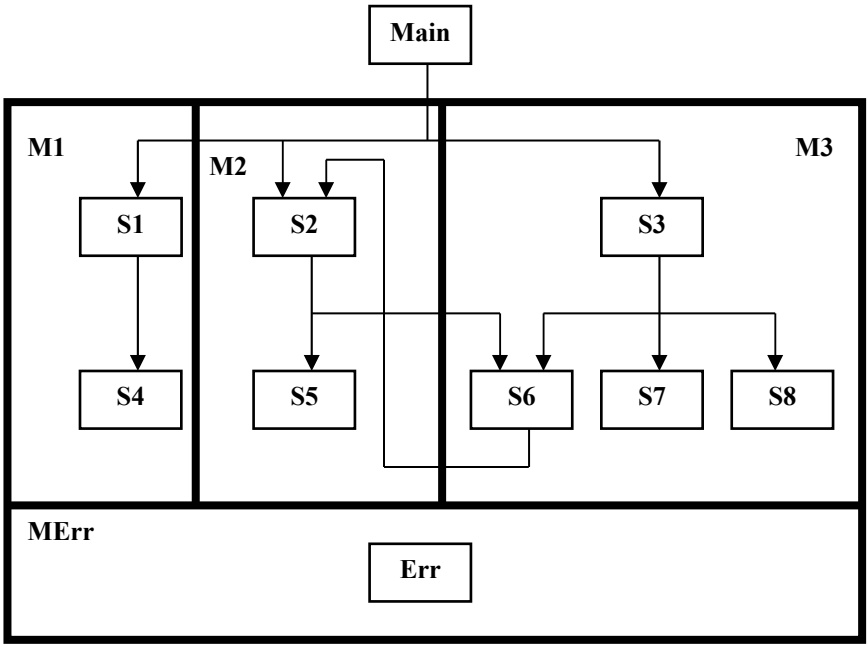
Варіант 22

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg5.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg4.



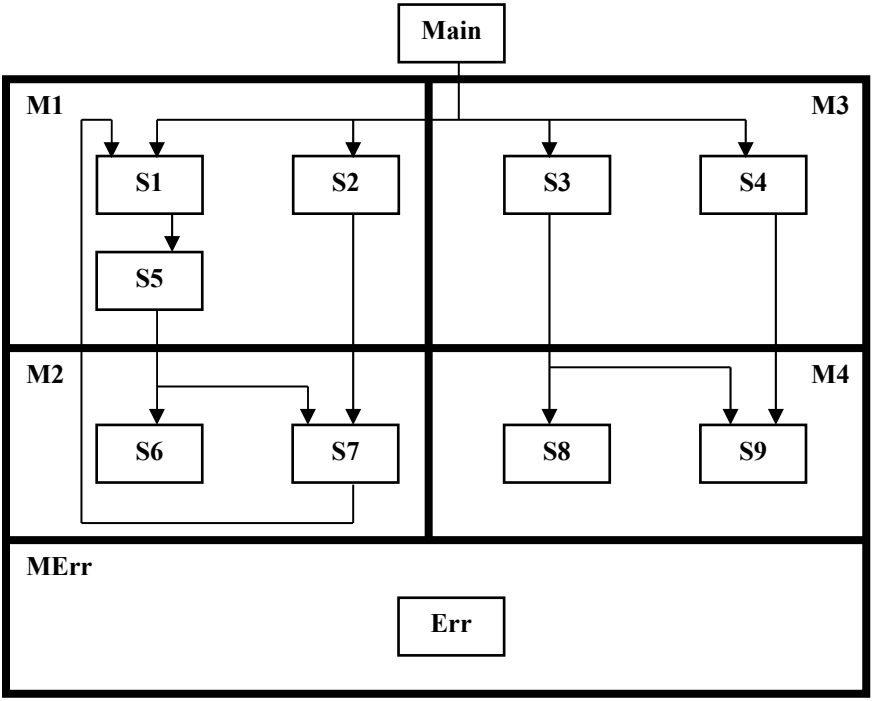
Варіант 23

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg2.

Константи: Cg1 – Cg4.

Змінні: Vg1 – Vg4.



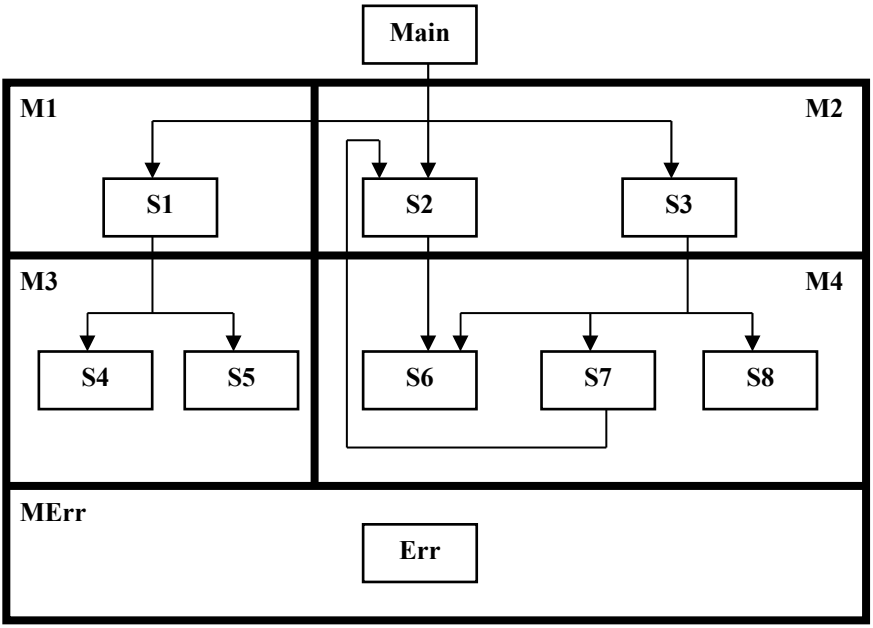
Варіант 24

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg4.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg5.



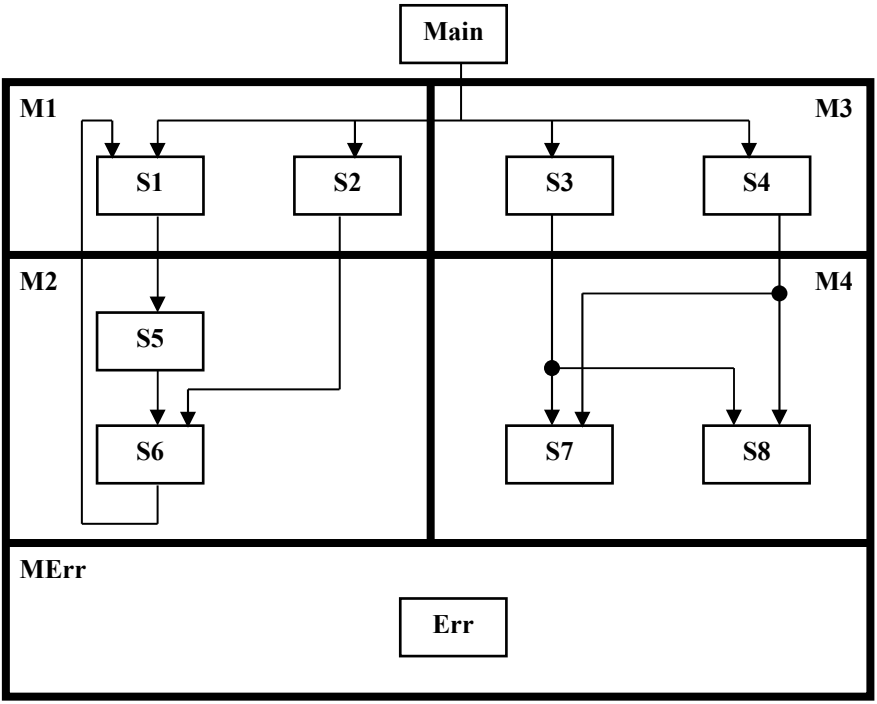
Варіант 25

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg5.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg3.



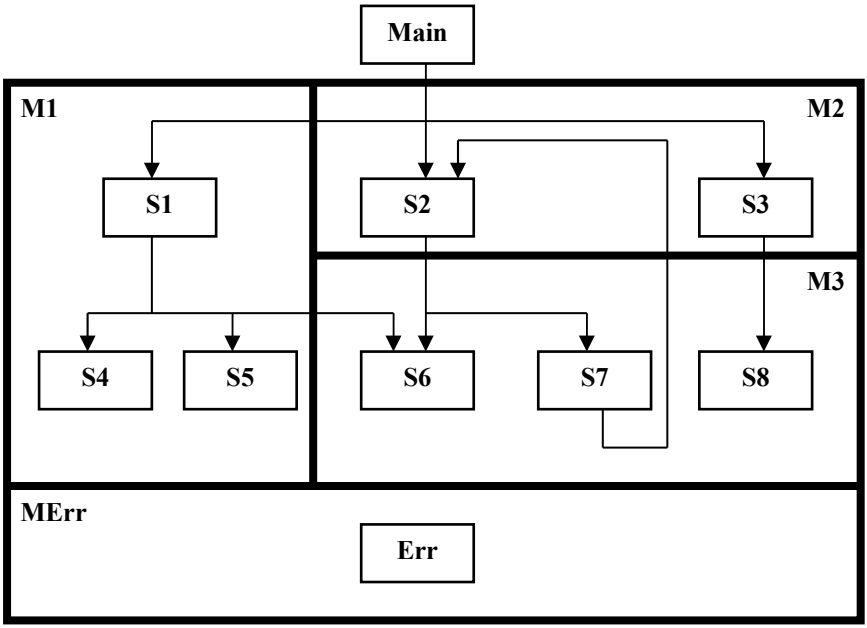
Варіант 26

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg3.

Константи: Cg1 – Cg4.

Змінні: Vg1 – Vg2.



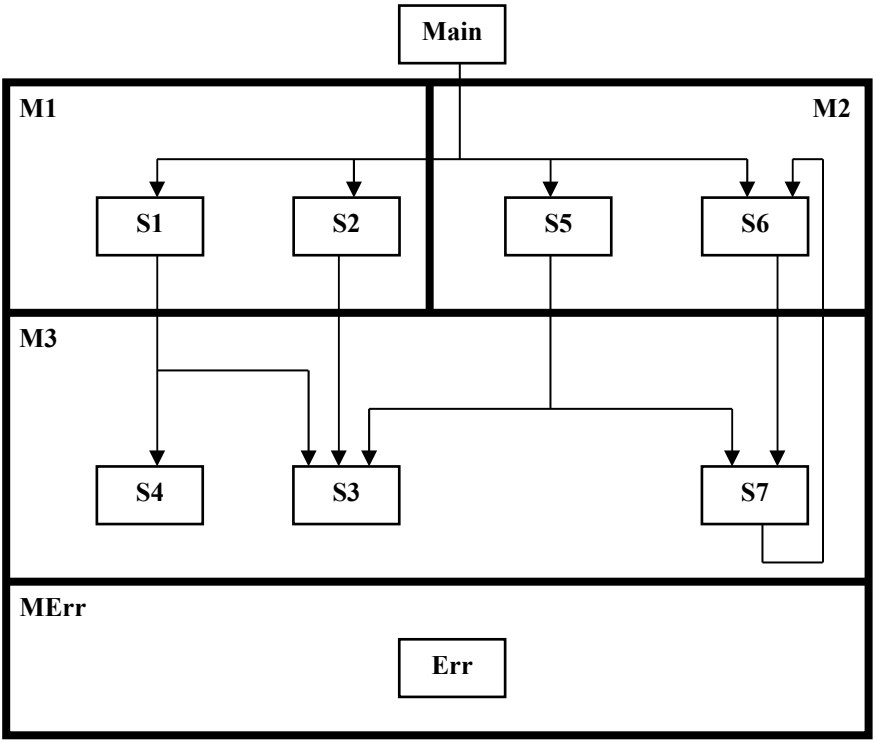
Варіант 27

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg3.

Константи: Cg1 – Cg2.

Змінні: Vg1 – Vg4.



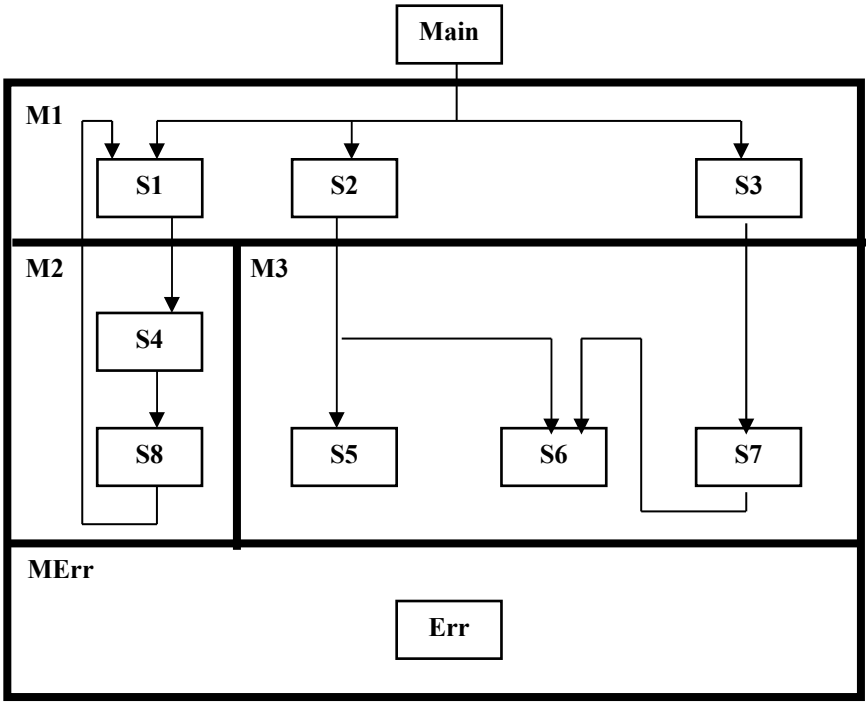
Варіант 28

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg5.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg2.



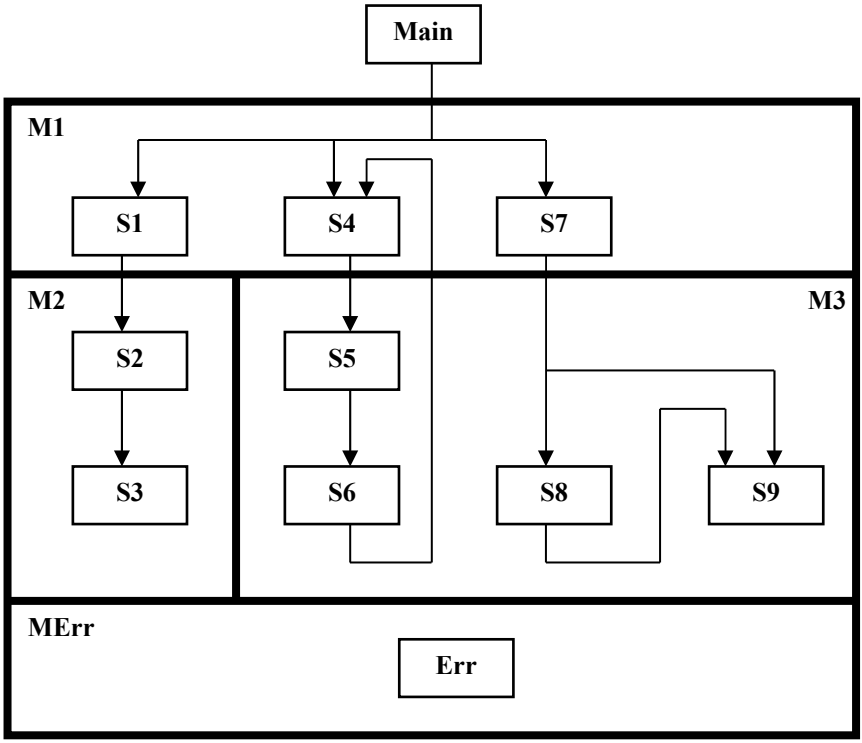
Варіант 29

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg3.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg3.



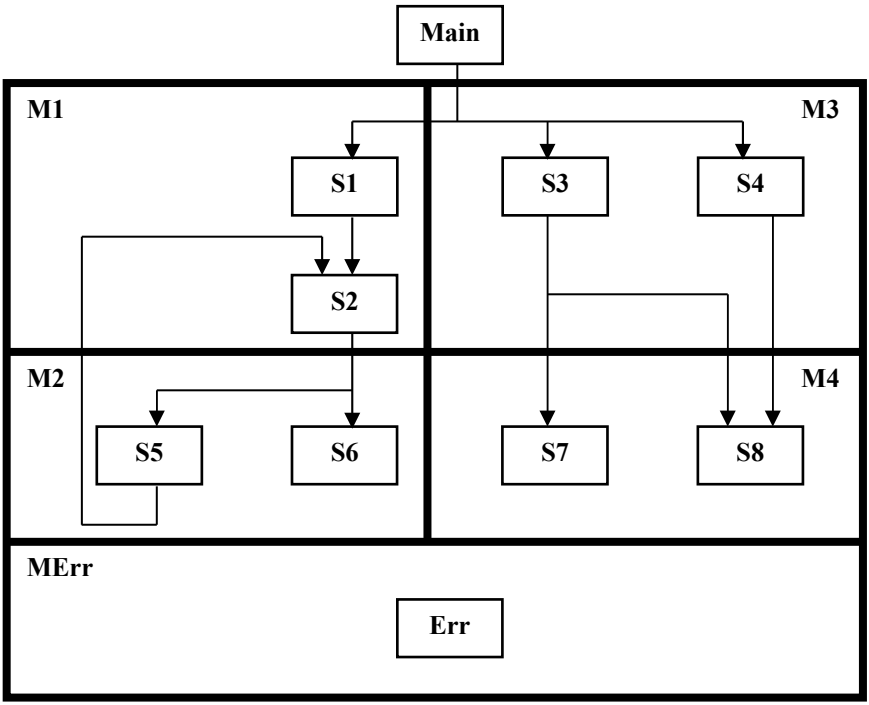
Варіант 30

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg3.

Константи: Cg1 – Cg4.

Змінні: Vg1 – Vg5.



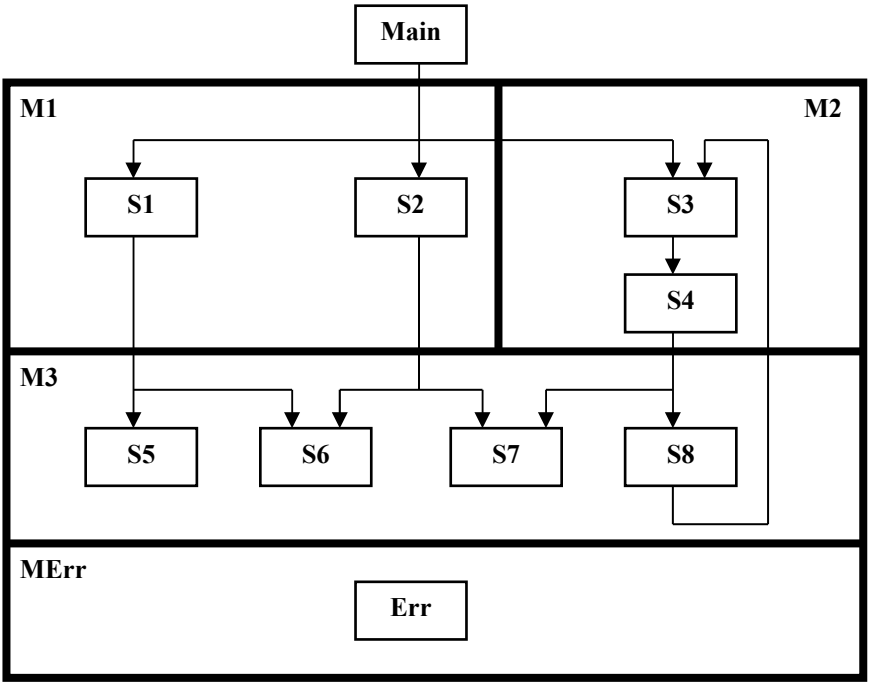
Варіант 31

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg4.

Константи: Cg1 – Cg2.

Змінні: Vg1 – Vg5.



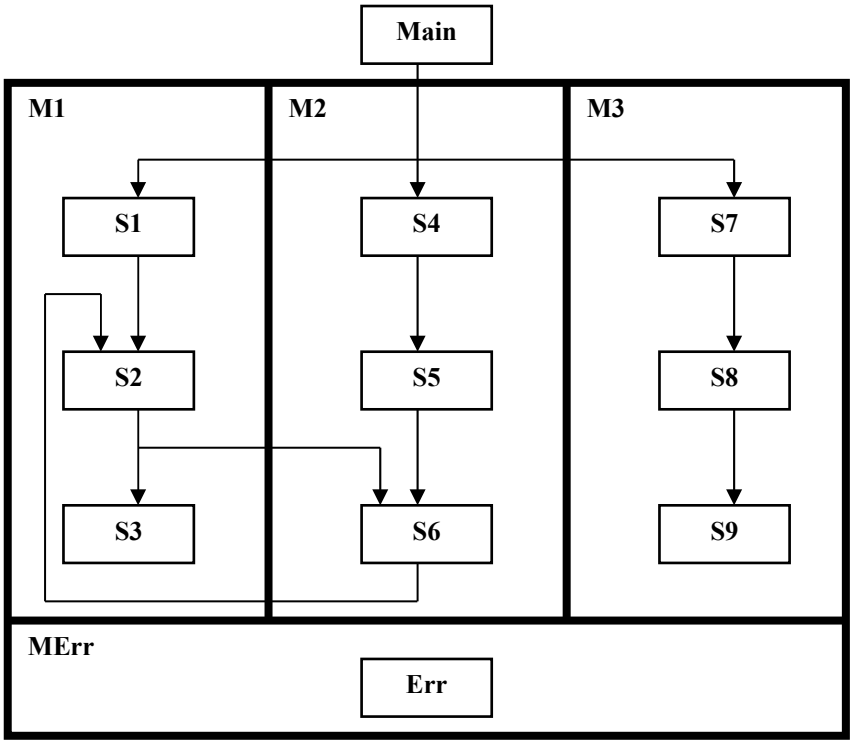
Варіант 32

В усіх модулях програми повинні бути доступними наступні спільні (глобальні) структури даних:

Типи :Tg1 – Tg4.

Константи: Cg1 – Cg3.

Змінні: Vg1 – Vg2.



5. ЛАБОРАТОРНА РОБОТА №2.5. НЕЗВ'ЯЗАНІ ДИНАМІЧНІ СТРУКТУРИ ДАНИХ

Мета лабораторної роботи

Метою лабораторної роботи №2.5. є засвоєння теоретичного матеріалу та набуття практичного досвіду використання незв'язаних динамічних структур даних при складанні різних алгоритмів.

Постановка задачі

1. Написати програму рішення задачі згідно варіанту.
2. При написанні програми всі задані масиви необхідно реалізувати як незв'язані динамічні дані.
3. На заданих масивах малого розміру змоделювати ситуацію роботи з масивами дуже великого розміру, коли розміщення у пам'яті усіх одразу масивів неможливе або недоцільне. Тобто пам'ять для кожного з масивів необхідно виділяти безпосередньо перед першим його використанням, а звільняти пам'ять відразу після останнього використання.
4. Для кожного із заданих масивів необхідно виділяти рівно стільки елементів, скільки задано у завданні. Тобто потрібно продемонструвати вміння працювати з виділенням пам'яті економно.

5. Повторювані частини алгоритму необхідно оформити у вигляді процедур або функцій (для введення, виведення та обробки масивів) з передачею масиву за допомогою параметра, що є безтиповим вказівником (*pointer*) або відкритим масивом (*array of* тип).

Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варіанту.
2. Текст програми.
3. Тестування програми, тобто початкові тестові дані та відповідні їм результати.
4. В якості результату роздрукувати дані тестування та розв'язку задачі на комп'ютері.

Контрольні питання

1. Яких типів допускаються динамічні дані?
2. В чому полягає відмінність оголошення звичайних змінних і незв'язаних динамічних змінних такого самого типу?
3. Що виконує операція розйменування динамічної змінної?
4. За допомогою яких процедур чи функцій виконується виділення пам'яті для динамічних даних?
5. В яких станах може знаходитись динамічна змінна?
6. В яких випадках динамічна змінна може знаходитись у невизначеному стані?

7. В чому полягає відмінність стану динамічної змінної nil та її невизначеного стану?

8. Які значення будуть порівнюватися, якщо виконується операція порівняння над ідентифікаторами динамічних змінних?

9. В чому полягають динамічні властивості незв'язаних динамічних даних?

10. В яких випадках доцільно використовувати незв'язані динамічні дані?

Методичні вказівки

1. Приклад передавання вказівника на масив за допомогою параметра, що є безтиповим вказівником (*pointer*).

```
program NonLnkDS_Matr;

const
    m1=3; n1=5;
    m2=4; n2=3;

type
    TMatr1 = array [1..m1, 1..n1] of integer;
    TMatr2 = array [1..m2, 1..n2] of integer;
    P_TMatr1 = ^TMatr1;
    P_TMatr2 = ^TMatr2;

var
    A : P_TMatr1;
    B : P_TMatr2;
```

```
procedure Inp (P : pointer; m,n : byte; ch : char);
var
    i, j : byte;

begin
     writeln('Input values of the array ', ch);
     case ch of
     'A': begin
               for i := 1 to m do
               begin
                    for j := 1 to n do
                        read(P_TMatr1(P)^[i,j]);
                    readln;
               end;
          end;
     'B': begin
               for i := 1 to m do
               begin
                    for j := 1 to n do
                        read(P_TMatr2(P)^[i,j]);
                    readln;
               end;
          end;
     end; { case }
end; { Inp }

. . .

BEGIN

     . . .

     Inp(A, m1, n1, 'A');

     . . .

     Inp(B, m2, n2, 'B');

     . . .

END.
```

2. Приклад передавання масиву за допомогою параметра, що відкритим масивом (*array of* тип).

```
program NonLnkDS_Vect;

const n1=5;
      n2=3;

type
    TVect1 = array [1..n1] of integer;
    TVect2 = array [1..n2] of integer;
    P_TVect1 = ^TVect1;
    P_TVect2 = ^TVect2;

var
   A : P_TVect1;
   B : P_TVect2;

procedure Inp (var V : array of integer; ch : char);
var
   i : byte;
begin
     writeln('Input values of the array ', ch);
     for i := 0 to High(V) do
         read(V[i]);
     readln;
end; { Inp }

. . .

BEGIN
. . .
     Inp(A^, 'A');

. . .

     Inp(B^, 'B');

. . .

END.
```

Варіанти завдань

Варіант 1

Дано три масиви ***A***[***10***], ***B***[***12***], ***C***[***8***] цілого типу. При послідовному перегляді знайти в кожному масиві добуток стількох елементів, сума яких не перевищує заданого цілого ***M***.

Варіант 2

Дано три масиви ***A***[***11***], ***B***[***9***], ***C***[***12***] дійсного типу. Виконати в кожному з них циклічний зсув елементів вліво на 6 розрядів.

Варіант 3

Дано три масиви ***A***[***8***], ***B***[***13***], ***C***[***10***] довгого дійсного типу. Нормувати елемент кожного масиву по максимальному (тобто розділити всі елементи масиву на його максимальний елемент).

Варіант 4

Дано чотири масиви: ***A***[***10***], ***B***[***10***], ***C***[***14***], D[***14***] довгого цілого типу. Обчислити скалярні добутки векторів | ***AB*** | і | ***CD*** |:

$$| XY | = \sum_{i=1}^{n} X_i Y_i$$

Варіант 5

Дано дві матриці ***A***[***6***,***6***], ***B***[***8***,***8***] довгого цілого типу. В кожній з матриць знайти суму елементів над головною діагоналлю і добуток елементів під нею.

Варіант 6

Дано дві матриці ***A***[***4***,***7***], ***B***[***9***,***3***] байтового типу. Повернути кожну із них навколо вертикальної осі (тобто поміняти перший стовпець з останнім, другий – з передостаннім і т.д.).

Варіант 7

Дано дві матриці ***A***[***5***,***5***], ***B***[***7***,***7***] натурального типу. Присвоїти змінній PA значення 1, якщо елементи матриці ***A*** симетричні відносно головної діагоналі, і значення 0 – в іншому випадку. Аналогічно присвоїти значення змінній PB для матриці ***B***.

Варіант 8

Дано дві матриці ***A***[***5***,***6***], ***B***[***7***,***4***] цілого типу. Для кожної з матриць підрахувати кількість елементів, які більші за задане ціле число ***k*** і мають непарну суму координат.

Варіант 9

Дано три масиви ***A***[***9***], ***B***[***11***], ***C***[***14***] дійсного типу. В кожному з них знайти суму елементів, розміщених після першого від'ємного елемента, і суму елементів, розміщених до нього.

Варіант 10

Дано три матриці ***A***[***5***,***5***], ***B***[***4***,**6**], ***C***[***7***,***3***] натурального типу. В кожній з них знайти всі максимальні елементи і замінити їх нулями.

Варіант 11

Дано три матриці ***A***[***7***,***4***], ***B***[3,***5***], ***C***[***5***,***5***] довгого цілого типу. Обчислити добуток елементів в кожному стовпчику заданих матриць.

Варіант 12

Дано три матриці ***A***[***3***,***9***], ***B***[***6***,***5***], ***C***[***5***,***7***] байтового типу. Обчислити суму елементів в кожному рядку заданих матриць.

Варіант 13

Дано три масиви ***A***[***11***], ***B***[***10***], ***C***[***14***] довгого цілого типу. Якщо перший елемент масиву від'ємний, то обчислити суму елементів, які знаходяться на непарних місцях в кожному із даних масивів, в іншому випадку – суму всіх елементів.

Варіант 14

Дано дві матриці ***A***[***5***,***5***], ***B***[***6***,***6***] натурального типу. Транспонувати кожну із них відносно головної діагоналі (без збереження початкової матриці).

Варіант 15

Дано три масиви ***X***[***10***], ***Y***[***15***], ***Z***[***12***] дійсного типу. В кожному з цих масивів елементи з номерами, кратними трьом, замінити напівсумою двох попередніх елементів, якщо їх добуток менше нуля, і залишити без змін в іншому випадку.

Варіант 16

Дано три масиви ***A***[***11***], ***B***[***10***], ***C***[***14***] довгого цілого типу і натуральне число ***k***. Обчислити в кожному масиві суму елементів, які стоять до елемента з номером ***k***, і добуток елементів, які стоять після елемента з номером ***k***.

Варіант 17

Дано три масиви ***A***[***19***], ***B***[***17***], ***C***[***23***] цілого типу. Для кожного масиву визначити кількість додатних елементів на парних позиціях і суму елементів на непарних позиціях.

Варіант 18

Дано дві матриці ***A***[***3***,***8***], ***B***[***7***,***4***] байтового типу. Повернути кожну із них навколо горизонтальної осі (тобто поміняти перший рядок з останнім, другий – з передостаннім і т.д.).

Варіант 19

Дано три масиви ***A***[***12***], ***B***[***15***], ***C***[***20***] натурального типу. Для кожного масиву визначити добуток елементів на непарних позиціях та знайти перший максимальний елемент масиву.

Варіант 20

Дано дві матриці ***A***[*7*,*7*], ***B***[*5*,*5*] дійсного типу. В кожній з матриць знайти суму елементів над побічною діагоналлю і добуток елементів під нею.

Варіант 21

Дано дві матриці ***A***[*4*,*5*], ***B***[*5*,*8*] цілого типу. При обході по рядках знайти останній мінімальний елемент кожної матриці, а також суму елементів матриці.

Варіант 22

Дано дві матриці ***A***[*4*,*9*], ***B***[*7*,*7*] байтового типу. В кожній матриці при обході по стовпчиках знайти перший максимальний і останній мінімальний елемент і поміняти їх місцями.

Варіант 23

Дано дві матриці ***A***[*4*,*4*], ***B***[*6*,*6*] натурального типу. В кожній з матриць знайти довільний максимальний елемент над побічною діагоналлю і замінити його нулем.

Варіант 24

Дано три масиви ***A***[*13*], ***B***[*17*], ***C***[*11*] цілого типу. В кожному з масивів підрахувати кількість пар поряд розміщених рівних між собою елементів. Кожен елемент може належати тільки одній парі.

Варіант 25

Дано три масиви ***A***[***11***], ***B***[***14***], ***C***[***12***] дійсного типу. Для кожного масиву зробити перевірку, чи впорядкований він за незменшенням.

Варіант 26

Дано три масиви ***A***[***10***], ***B***[***7***], ***C***[***13***] байтового типу і натуральне число ***k***. Виконати в кожному з них циклічний зсув елементів вправо на ***k*** позицій.

Варіант 27

Дано три матриці ***A***[***4***,***4***], ***B***[*7*,*7*], ***C***[***6***,***6***] натурального типу. Транспонувати кожну із них відносно побічної діагоналі (без збереження початкової матриці).

Варіант 28

Дано три масиви ***A***[***17***], ***B***[***10***], ***C***[***11***] цілого типу. Поміняти в кожному з них порядок слідування елементів на протилежний (виконати “дзеркальне відображення” елементів масивів).

Варіант 29

Дано три масиви ***A***[***9***], ***B***[***5***], ***C***[***12***] натурального типу. В кожному із них визначити кількість елементів, які мають парні порядкові номери і є непарними числами.

Варіант 30

Дано дві матриці ***A***[***4***,***6***], ***B***[***8***,***5***] дійсного типу. В кожній із них поміняти місцями підряд розміщенні рядки з непарними і парними номерами (тобто перший рядок з другим, третій з четвертим і т.д.).

6. ЛАБОРАТОРНА РОБОТА №2.6. ЗВ'ЯЗАНІ ДИНАМІЧНІ СТРУКТУРИ ДАНИХ. СПИСКИ

Мета лабораторної роботи

Метою лабораторної роботи №2.6 є засвоєння теоретичного матеріалу та набуття практичного досвіду використання зв'язаних динамічних структур даних у вигляді одно- та двозв'язаних списків при складанні різних алгоритмів.

Постановка задачі

1. Створити список з ***n*** (***n***>0) елементів (***n*** вводиться з клавіатури), якщо інша кількість елементів не вказана у конкретному завданні,

2. Тип ключів (інформаційних полів) задано за варіантом.

3. Значення елементів списку взяти самостійно такими, щоб можна було продемонструвати коректність роботи алгоритму програми. Введення значень елементів списку можна виконати довільним способом (випадкові числа, формування значень за формулою, введення з файлу чи з клавіатури).

4. Вид списку (черга, стек, дек, прямий однозв'язний лінійний список, обернений однозв'язний лінійний список, двозв'язний лінійний список, однозв'язний кільцевий список, двозв'язний

кільцевий список,) вибрати самостійно з метою найбільш доцільного рішення поставленої за варіантом задачі.

5. Виконати над створеним списком дії, вказані за варіантом, та коректне звільнення пам'яті списку.

6. **При виконанні заданих дій, виводі значень елементів та звільненні пам'яті списку вважати, що довжина списку (кількість елементів *n* чи *2n*) *невідома* на момент виконання цих дій.**

7. Повторювані частини алгоритму необхідно оформити у вигляді процедур або функцій (для створення, обробки, виведення та звільнення пам'яті списків) з передачею списку за допомогою параметра(ів).

Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варіанту.

2. Текст програми.

3. Тестування програми, тобто початкові дані та відповідні їм результати.

4. В якості результату роздрукувати дані тестування та розв'язку задачі на комп'ютері.

Контрольні питання

1. Чи існують обмеження на кількість елементів у списку, якщо так, то які?

2. Які є види спискових структур даних з точки зору їх логічного використання (стек, черга, тощо). Поясніть їх особливості та відмінності між ними.

3. Які є вимоги до структури елемента зв'язних динамічних даних?

4. В чому полягає особливість опису типів для створення зв'язних динамічних даних?

5. Якою повинна бути структура елементу лінійного двозв'язного списку?

6. Скільки вказівників і якого призначення необхідно для роботи з чергою?

7. Скільки вказівників і якого призначення необхідно для роботи з стеком?

8. Скільки вказівників і якого призначення необхідно для роботи з деком?

9. Скільки вказівників і якого призначення необхідно для роботи з лінійними однозв'язними списками?

10. Скільки вказівників і якого призначення необхідно для роботи з лінійними двозв'язними списками?

Варіанти завдань

Варіант 1

Ключами елементів списку є рядки довжиною не більше 10-ти символів, що складаються з латинських літер. Відсортувати елементи списку у лексикографічному порядку, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»), методом вибору.

Варіант 2

Ключами елементів списку є дійсні числа. Кількість елементів списку повинна дорівнювати ***2n***. Обчислити значення виразу: $(a_1 - a_{2n})(a_3 - a_{2n-2})\ldots(a_{2n-1} - a_2)$, де a_i – ***i***-й елемент списку.

Варіант 3

Ключами елементів списку є символи з множини латинських літер та цифр. Перекомпонувати список таким чином, щоб усі цифри стояли на початку списку, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 4

Заданий список є *чергою*. Ключами елементів списку є символи. Переписати елементи цієї черги до іншої черги у оберненому порядку, *не підраховуючи кількості елементів у черзі*.

Варіант 5

Ключами елементів списку є дійсні числа. Виконати циклічний зсув елементів списку на ***k*** позицій вліво (***k*** – натуральне і не перевищує кількості елементів списку). При необхідності дозволяється використати ще один список, інші структури даних, крім простих змінних, використовувати не дозволяється.

Варіант 6

Ключами елементів списку є дійсні числа. Відсортувати елементи списку за незбільшенням, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»), методом обміну (“бульбашки”) з використанням «прапорця».

Варіант 7

Ключами елементів списку є цілі числа. Кількість елементів списку повинна дорівнювати ***2n***. Обчислити значення виразу: $a_1 a_{2n} + a_2 a_{2n-1} + \ldots + a_n a_{n+1}$, де a_i – ***i***-й елемент списку.

Варіант 8

Ключами елементів списку є цілі ненульові числа, причому кількість від’ємних чисел дорівнює кількості додатних. Перекомпонувати список так, щоб отримати послідовність чисел із чергуванням знаків, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 9

Ключами елементів списку є різні дійсні числа. Знайти максимальний та мінімальний елементи списку. Вставити до списку два нових елементи: після елементу з максимальним значенням вставити елемент з мінімальним значенням, а після елементу з мінімальним значенням вставити елемент з максимальним значенням.

Варіант 10

Ключами елементів списку є цілі числа. Визначити кількість елементів списку, значення яких більше за задане ціле число ***M***, та вставити нових елементів після ***k***-го (***k*** – натуральне і не перевищує кількості елементів списку) елементу списку.

Варіант 11

Ключами елементів списку є латинські літери. Відсортувати елементи списку у лексикографічному порядку, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»), методом вставки.

Варіант 12

Ключами елементів списку є цілі числа. Обчислити значення виразу: $(a_1 + a_2 + 2a_n)(a_2 + a_3 + 2a_{n-1})\ldots(a_{n-1} + a_n + 2a_2)$, де a_i – ***i***-й елемент списку.

Варіант 13

Ключами елементів списку є цілі числа. Переставити елементи списку так, щоб спочатку розташовувались додатні, потім нульові, а за ними від'ємні елементи, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 14

Ключами елементів списку є рядки довжиною не більше 5-ти символів. Перекомпонувати список так, щоб елементи списку були розташовані у оберненому порядку (виконати «дзеркальне відображення» списку), не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 15

Ключами елементів списку є дійсні числа. Перекомпонувати елементи списку таким чином, щоб його елементи розташовувались у такому порядку: $a_1, a_n, a_2, a_{n-1}, \ldots, a_{\left[\frac{n+1}{2}\right]}$, де a_i – i-й елемент списку. При необхідності дозволяється використати ще один список, інші структури даних, крім простих змінних, використовувати не дозволяється.

Варіант 16

Ключами елементів списку є дійсні числа. Розширити список, дописавши в його кінець свої ж елементи, але у оберненому порядку, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 17

Ключами елементів списку є дійсні числа. Обчислити значення виразу: $a_1 a_n + a_2 a_{n-1} + \ldots a_n a_1$, де a_i – ***i***-й елемент списку.

Варіант 18

Ключами елементів списку є цілі ненульові числа, які розташовуються у наступному порядку: 10 додатних, 10 від'ємних і т.д. Кількість елементів списку ***n*** повинна бути кратною 20. Перекомпонувати список так, щоб розташування елементів було наступним: 5 додатних, 5 від'ємних і т.д., не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 19

Задано два списки, список ***S1*** довжиною ***2n*** елементів і список ***S2*** довжиною ***n*** елементів. Ключами елементів обох списків є натуральні числа. Вставити список ***S2*** у середину списку ***S1***, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 20

Ключами елементів списку є цілі числа. Кількість елементів списку повинна дорівнювати ***2n***. Перекомпонувати елементи списку так, розташування елементів було наступним: $a_1, a_{n+1}, a_2, a_{n+2}, a_3, \ldots, a_n, a_{2n}$, де a_i – ***i***-й компонент файлу, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 21

Ключами елементів списку є дійсні числа. Виконати наступні дії: якщо елементи списку впорядковані за незбільшенням, то залишити його без змін, інакше перезаписати елементи списку у оберненому порядку. При необхідності дозволяється використати ще один список, інші структури даних, крім простих змінних, використовувати не дозволяється.

Варіант 22

Ключами елементів списку є цілі ненульові числа, які розташовуються в наступному порядку: 5 від'ємних, 5 додатних і т.д. Кількість елементів списку ***n*** повинна бути кратною 20-ти. Перекомпонувати елементи списку так, щоб розташування елементів було наступним: 10 від'ємних, 10 додатних і т.д., не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 23

Ключами елементів списку є цілі числа. Виконати циклічний зсув елементів списку на ***k*** позицій вправо (***k*** – натуральне і не перевищує кількості елементів списку). При необхідності дозволяється використати ще один список, інші структури даних, крім простих змінних, використовувати не дозволяється.

Варіант 24

Ключами елементів списку є цілі ненульові числа. Кількість елементів списку ***n*** повинна бути кратною 4-ом, причому кількість від'ємних чисел дорівнює кількості додатних. Перекомпонувати елементи списку так, розташування елементів було наступним: два додатних, два від'ємних і т.д., не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 25

Ключами елементів списку є латинські літери. Перекомпонувати список так, щоб спочатку розташовувались елементи з голосними латинськими літерами, а потім елементи з приголосними латинськими літерами, не змінюючи початкового взаємного розташування літер. Наприклад:

початковий список: university

результат: uieiynvrst

При необхідності дозволяється використати ще один список, інші структури даних, крім простих змінних, використовувати не дозволяється.

Варіант 26

Ключами елементів списку є натуральні числа. Перекомпонувати список так, щоб спочатку йшли елементи з ключами, що діляться на 3 без залишку, потім елементи з ключами, що діляться на 3 із залишком 1, і нарешті ті, що діляться на 3 із залишком 2, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 27

Ключами елементів списку є цілі числа. Перекомпонувати список так, щоб спочатку розташовувались додатні, потім нульові, а за ними від'ємні елементи, не змінюючи початкового взаємного розташування елементів. Наприклад:

початковий файл: -1 0 5 -9 8 -3 5 0 -7 4

результат: 5 8 5 4 0 0 -1 -9 -3 -7.

При необхідності дозволяється використати ще один список, інші структури даних, крім простих змінних, використовувати не дозволяється.

Варіант 28

Ключами елементів списку є цілі ненульові числа. Кількість елементів списку ***n*** повинна бути кратною 10-ти, а елементи у початковому списку розташовуватись із чергуванням знаків. Перекомпонувати список, змінюючи порядок чисел всередині кожного десятка елементів так, щоб спочатку йшли від'ємні числа цього десятка елементів, а за ними – додатні, не використовуючи

додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Варіант 29

Ключами елементів списку є рядки довжиною не більше 25-ти символів. Кількість елементів списку ***n*** повинна бути кратною 20-ти. Перекомпонувати список всередині кожних 20-ти елементів, розташувавши їх у наступному порядку: $a_1, a_{11}, a_2, a_{12}, \ldots, a_{21}, a_{31}, a_{22}, a_{32}, \ldots$, де a_i – ***i***-й елемент списку. При необхідності дозволяється використати ще один список, інші структури даних, крім простих змінних, використовувати не дозволяється.

Варіант 30

Ключами елементів списку є цілі числа. Відсортувати елементи списку за незменшенням, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»), методом шейкерного сортування.

7. ЛАБОРАТОРНА РОБОТА №2.7. ЗВ'ЯЗАНІ ДИНАМІЧНІ СТРУКТУРИ ДАНИХ. ДЕРЕВА

Мета лабораторної роботи

Метою лабораторної роботи №2.7 є засвоєння теоретичного матеріалу та набуття практичного досвіду використання зв'язаних динамічних структур даних у вигляді дерев при складанні різних алгоритмів.

Постановка задачі

1. Задано **n** цілих чисел ($15 \leq \mathbf{n} \leq 31$). Побудувати динамічну структуру даних у вигляді двійкового дерева заданої за варіантом форми, значеннями вершин якого є задані числа.
2. Виконати умову задачі за варіантом, використавши заданий спосіб обходу дерева.
3. Надрукувати всю зібрану у вершинах дерева інформацію в порядку, згідно заданого способу обходу дерева.
4. Виконати налагодження написаної програми.

Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варіанту.
2. Текст програми.
3. В якості результату нарисувати дерева та надрукувати відповідну їм результуючу інформацію щонайменше для трьох вхідних послідовностей чисел з різним значенням **n** ($15 \leq \mathbf{n} \leq 31$).

Контрольні питання

1. Визначення дерева як динамічної структури даних. Визначення та приклади степені дерева. Визначення та приклади висоти (глибини) дерева.
2. Поняття довжини внутрішнього шляху дерева. Приклад.
3. Поняття довжини зовнішнього шляху дерева. Приклад.
4. Поняття впорядкованого дерева. Приклад.
5. Поняття та види піраміди. Приклади.
6. Поняття ідеально-збалансованого дерева. Приклади.
7. Поняття дерева пошуку. Приклад.
8. Рекурсивне визначення дерева.
9. Структура вершини дерева та її опис на мові програмування.
10. Способи обходу двійкового дерева.

Варіанти завдань

Таблиця 1

Варіант	Задача	Форма дерева	Спосіб обходу
1.	1	Ідеально-збалансоване дерево	7
2.	1	Ідеально-збалансоване дерево	4
3.	2	Ідеально-збалансоване дерево	8
4.	2	Ідеально-збалансоване дерево	5
5.	1	АВЛ-дерево (збалансоване дерево)	9
6.	1	АВЛ-дерево (збалансоване дерево)	6
7.	2	АВЛ-дерево (збалансоване дерево)	10
8.	2	АВЛ-дерево (збалансоване дерево)	1
9.	1	Ідеально-збалансоване дерево	11
10.	1	Ідеально-збалансоване дерево	2
11.	2	Ідеально-збалансоване дерево	12
12.	2	Ідеально-збалансоване дерево	3
13.	2	АВЛ-дерево (збалансоване дерево)	7
14.	2	АВЛ-дерево (збалансоване дерево)	4
15.	1	АВЛ-дерево (збалансоване дерево)	8
16.	1	АВЛ-дерево (збалансоване дерево)	5
17.	2	Ідеально-збалансоване дерево	9
18.	2	Ідеально-збалансоване дерево	6
19.	1	Ідеально-збалансоване дерево	10
20.	1	Ідеально-збалансоване дерево	1
21.	2	АВЛ-дерево (збалансоване дерево)	11
22.	2	АВЛ-дерево (збалансоване дерево)	2
23.	1	АВЛ-дерево (збалансоване дерево)	12
24.	1	АВЛ-дерево (збалансоване дерево)	3
25	1	Ідеально-збалансоване дерево	8
26	2	Ідеально-збалансоване дерево	7
27	1	Ідеально-збалансоване дерево	5
28	2	АВЛ-дерево (збалансоване дерево)	9
29	1	АВЛ-дерево (збалансоване дерево)	4
30	2	АВЛ-дерево (збалансоване дерево)	6

Задачі

1. а) Знайти перше максимальне і останнє мінімальне значення ключів вершин дерева, використовуючи заданий спосіб обходу, та поставити на ці вершини окремі вказівники.

б) Промаркірувати окремо всі вершини, значення ключа яких дорівнює максимальному значенню та мінімальному значенню, двома різними ознаками, використавши додаткові інформаційні поля у вершинах дерева.

2. а) Знайти першу і останню вершини дерева, значення ключів яких співпадає із заданим цілим числом X, використовуючи заданий спосіб обходу, та поставити на ці вершини окремі вказівники.

б) Промаркірувати окремо всі вершини, що розташовані нижче знайдених вершин, двома різними ознаками, використавши додаткові інформаційні поля у вузлах дерева.

Способи обходу дерева

1. Виписування значень ключів вершин дерева або спуск по гілках дерева виконати в такій послідовності:

а) спуск по під-дереву нащадка з меншим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною;

б) виписування значення ключа поточної вершини;

в) спуск по під-дереву нащадка з більшим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною.

У випадку рівних значень ключів нащадків спочатку виписувати значення ключа лівого нащадка або спускатися по його під-дереву.

2. Виписування значень ключів вершин дерева або спуск по гілках дерева виконати в такій послідовності:

а) спуск по під-дереву нащадка з більшим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною;

б) виписування значення ключа поточної вершини;

в) спуск по під-дереву нащадка з меншим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною.

У випадку рівних значень ключів нащадків спочатку виписувати значення ключа правого нащадка або спускатися по його під-дереву.

3. Виписування значень ключів вершин дерева або спуск по гілках дерева виконати в такій послідовності:

а) виписування значення ключа поточної вершини;

б) спуск по під-дереву нащадка з меншим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною;

в) спуск по під-дереву нащадка з більшим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною.

У випадку рівних значень ключів нащадків спочатку виписувати значення ключа лівого нащадка або спускатися по його під-дереву.

4. Виписування значень ключів вершин дерева або спуск по гілках дерева виконати в такій послідовності:

а) виписування значення ключа поточної вершини;

б) спуск по під-дереву нащадка з більшим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною;

в) спуск по під-дереву нащадка з меншим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною.

У випадку рівних значень ключів нащадків спочатку виписувати значення ключа правого нащадка або спускатися по його під-дереву.

5. Виписування значень ключів вершин дерева або спуск по гілках дерева виконати в такій послідовності:

а) спуск по під-дереву нащадка з меншим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною;

б) спуск по під-дереву нащадка з більшим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною;

в) виписування значення ключа поточної вершини.

У випадку рівних значень ключів нащадків спочатку виписувати значення ключа лівого нащадка або спускатися по його під-дереву.

6. Виписування значень ключів вершин дерева або спуск по гілках дерева виконати в такій послідовності:

а) спуск по під-дереву нащадка з більшим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною;

б) спуск по під-дереву нащадка з меншим значенням ключа або виписування значення ключа цього нащадка, якщо його вершина є термінальною;

в) виписування значення ключа поточної вершини.

У випадку рівних значень ключів нащадків спочатку виписувати значення ключа правого нащадка або спускатися по його під-дереву.

7. Виписування значення ключа поточної вершини, або спуск по під-дереву одного з двох нащадків, або виписування значення

ключа цього нащадка (якщо його вершина є термінальною), виконати в такій послідовності: мінімальне значення, середнє значення, максимальне значення.

У випадку рівних значень ключів виконувати обхід згідно схеми Left-Root-Right.

8. Виписування значення ключа поточної вершини, або спуск по під-дереву одного з двох нащадків, або виписування значення ключа цього нащадка (якщо його вершина є термінальною), виконати в такій послідовності: максимальне значення, середнє значення, мінімальне значення.

У випадку рівних значень ключів виконувати обхід згідно схеми Root-Left-Right.

9. Виписування значення ключа поточної вершини, або спуск по під-дереву одного з двох нащадків, або виписування значення ключа цього нащадка (якщо його вершина є термінальною), виконати в такій послідовності: середнє значення, мінімальне значення, максимальне значення.

У випадку рівних значень ключів виконувати обхід згідно схеми Right-Root-Left.

10. Виписування значення ключа поточної вершини, або спуск по під-дереву одного з двох нащадків, або виписування значення ключа цього нащадка (якщо його вершина є

термінальною), виконати в такій послідовності: середнє значення, максимальне значення, мінімальне значення.

У випадку рівних значень ключів виконувати обхід згідно схеми Left-Right-Root.

11. Виписування значення ключа поточної вершини, або спуск по під-дереву одного з двох нащадків, або виписування значення ключа цього нащадка (якщо його вершина є термінальною), виконати в такій послідовності: мінімальне значення, максимальне значення, середнє значення.

У випадку рівних значень ключів виконувати обхід згідно схеми Root-Right-Left.

12. Виписування значення ключа поточної вершини, або спуск по під-дереву одного з двох нащадків, або виписування значення ключа цього нащадка (якщо його вершина є термінальною), виконати в такій послідовності: максимальне значення, мінімальне значення, середнє значення.

У випадку рівних значень ключів виконувати обхід згідно схеми Right-Root-Left.

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

Основна література

1. Вирт Н. Алгоритмы и структуры данных. М.: Мир, 1989.

2. Марченко А.И., Марченко Л.А. Программирование в среде Turbo Pascal 7.0. – 9-е изд. – К.: Век+, Спб.: КОРОНА-Век, 2007. – 464 с., ил.

3. Т.Кормен, Ч.Лейзерзон, Р.Ривест. Алгоритмы: построение и анализ. М.: МЦНМО, 2000.

4. Кнут Д. Искусство программирования для ЭВМ. т.1. Основные алгоритмы., М.: Мир, 1976.

5. Кнут Д. Искусство программирования для ЭВМ. т.2. Получисленные алгоритмы., М.: Мир, 1977.

6. Кнут Д. Искусство программирования для ЭВМ. т.3. Сортировка и поиск., М.: Мир, 1978.

Додаткова література

7. Вирт Н. Алгоритмы + структуры данных = программы. М.:Мир, 1985.

8. Г.Буч. Объектно-ориентированное проектирование с примерами применения: Пер. с англ. – М.: Конкорд, 1992. – 519 с., ил.

9. Методичні вказівки для виконання лабораторних робіт з дисципліни „Програмування” для студентів спеціальностей „Комп’ютерні системи та мережі”, „Спеціалізовані комп’ютерні системи”, „Системне програмування” , ч. 1,2 (Укл. Р.Ф. Колінько, О.І. Марченко). Київ НТУУ „КПІ” – 2004, видавництво „Ювета”.

10. Абрамов С.А., Гнездилова Г.Г., Капустина Е.Н., Селюн, М.И. Задачи по программированию., М.: наука, 1988.

11. Martin J., McClure C. Structured techniques : The basis for CASE., 1988.

12. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов., М.: Мир, 1979.

13. Гудман С., Хидетниеми С. Введение в разработку и анализ алгоритмов., М.: Мир, 1981.

14. Дал У., Дейкстра., Хоор К. Структурное программирование., М.: Мир, 1975.

15. Дейкстра Э. Дисциплина программирования. М.: Мир, 1978.

16. Вирт Н. Систематическое программирование. Введение., М.: Мир,1977.

17. Лингер Р., Миллс Х., Уитт Б. Теория и практика структурного программирования., М.: Мир, 1982.

18. Брукс Ф.П. Как проектируются и создаются программные комплексы., М.: Наука, 1979.