МІНІСТЕРСТВО МОЛОДІ ТА СПОРТУ УКРАЇНИ

Запорізький національний технічний університет

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт з дисципліни:

“Алгоритми та структури даних”

для студентів спеціальності 122 “Комп’ютерні науки”
(всіх форм навчання)

2017

Методичні вказівки до лабораторних робіт з дисципліни “Алгоритми та структури даних” для студентів спеціальності 122 “Комп’ютерні науки” (усіх форм навчання)/Уклад.: Рисіков В.П., – Запоріжжя: ЗНТУ, 2017. – 66с.

Укладачі: В.П. Рисіков, к.т.н., доцент,

Рецензент: С.К. Корнієнко, к.т.н., доцент

Відповідальний
за випуск: С.О. Субботін, д.т.н., професор

Затверджено
на засіданні кафедри
"Програмні засоби"

Протокол № від р.

ЗМІСТ

І. ЗАГАЛЬНІ ВІДОМОСТІ ПРО СПОСОБИ ПОДАННЯ СТРУКТУРНОЇ ІНФОРМАЦІЇ В ЕОМ

В цьому розділі викладені концепції подання структурної інформації в ОП ЕОМ.

Логічна структура даних не несе у собі інформації про те, яким чином дані та зв'язки між ними подаються в пам'яті ЕОМ.

Звідси, абстрактна структура (логічна) не може безпосередньо використовуватись при обробці даних на ЕОМ. Між тим така структура зручна і для неї нема альтернативи при розробці немашинних алгоритмів рішення практичних задач обробки даних. Тому, при перетворенні немашинних алгоритмів у машині з'являється необхідність перетворення абстрактної структури даних у так звану структуру зберігання у пам'яті ЕОМ, яка не тільки зберігає саму абстрактну структуру, але й доповнює її відомості про адресу кожного даного та зв'язки адрес у відповідності зі зв'язками абстрактної структури.

Проблема відображення абстрактних структур зберігання полягає у знаходженні ефективних методів, тобто таких, котрим відповідають наступні якості:

- легка формалізуємість з можливістю подання результата формалізації у вигляді адресної функції;
- можливість реалізації адресної функції простим алгоритмом з мінімальним обчисленням та виконання відображення в автоматичному режимі;
- адресація елементів даних.

Сукупність цих якостей забезпечує можливість автоматичного формування структур зберігання на основі опису абстрактних структур, без котрого неможлива побудова трансляторів, які дозволяють обробляти дані, подані у різних абстрактних структурах.

При реалізації адресних функцій слід пам'ятати адресну структуру ОП ЕОМ, у яких пам'ять подана як єдиний вектор з елементами - байтами, проідентифікованими адресами та впорядкованими по збільшенню адрес від 0 до найбільшої. Ця

структура відображає найбільш відповідну структуру даних для зберігання в ОП - кінцеву послідовність, яку називають списком. Список - лінійна структура. Багато типів абстрактних структур без особливих перешкод можуть бути перебудовані у список. Це дуже важливо, тому що для відображення списка у структуру зберігання імені застосовується тип адресних функцій: послідовне або зв'язане розподілення пам'яті.

При послідовному розподіленні пам'яті елементи списка розташовуються у послідовних елементах пам'яті.

При зв'язаному розподіленні пам'яті необхідно вказати за допомогою зв'язків (вказівників) відношення слідування та передування елементів списка. Вказівниками є адреси, які зберігаються біля кожного елемента списка. Кожний елемент зберігає вказівник на слідуючий (попередній) або на обидва елемента відразу. Наявність адрес зв'язку дозволяє розташовувати елементи списка вільно у будь-якій вільній ділянці пам'яті. При цьому лінійна структура списку забезпечується вказівниками.

Зв'язане подання списка називається зв'язанним списком. Зв'язані списки - зручна форма подання в пам'яті ЕОМ абстрактних лінійних структур, що диннамічно змінюються. Вказані типи адресних функцій застосовують тільки для списків. Велику кількість типів абстрактних структур не можливо перебудувати до списку.

Звичайно, при цьому стає проблема вибору адресних функцій для відображення абстрактних структур до структур зберігання. Для цих нелінійних абстрактних структур розроблена єдина методика відображення до структур зберігання, суть якої у наступному.

Нелінійна абстрактна структура моделюється з багатозв'язаним списком. До багатозв'язаного списку застосовується адресна функція зв'язного розподілення пам'яті. Багатозв'язний список відрізняється від списка наявністю явно заданих зв'язків між собою.

2 ЛАБОРАТОРНА РОБОТА №1

ОПИС МАСИВІВ, ДІЇ НАД НИМИ

Мета роботи: Вивчення логічної структури масивів різної розмірності та її реалізації у відповідну фізичну структуру. Вивчення способів доступу до будь-якого елемента масива та обчислення адреси будь-якого елемента.

2.1 ЗАВДАННЯ

2.1.1 Отримати елементи квадратичної матриці $A=(A_{ij})$ $(i,j=1,...n)$, де $A_{ij}=f(ij)$, $f(ij)$ – задана функція.

2.1.2 З матриці А отримати компоненти вектора $X=(X_i)$ $(i=1,...n)$.

2.1.3 Обчислити значення $U=g(x)$, де $g(x)$ – задана функція.

Як результат видати на роздрук вихідні рядки матриці А, перебудувати марицю А, елементи вектора Х та значення U.

Значення P_n, функції $f(ij)$ і $g(x)$, а також правила отримання вектора Х зумовлюються варіантом завдання.

2.2 ПРАВИЛА ОТРИМАННЯ ВЕКТОРА Х ПО МАТРИЦІ А

2.2.1 X_i - скалярний добуток і-того рядка матриці А на стовпець, який містить перший по порядку найбільший елемент даного рядка.

2.2.2 За вектор Х прийняти головну діагональ матриці А, перебудованої таким чином:

на початку кожного рядка зібрати невід'ємні елементи, а в кінці від'ємні.

2.2.3 У матриці А знайти перший по черзі рядок з найбільшою сумою його елементів, в якості компонент вектора прийняти впорядковані по незменшуванню елементи цього рядка.

2.2.4 Рядки матриці A впорядкувати по незменшенню елементів його першого стовпчика і за вектор X, прийняти головну діагональ перебудованої матриці.

2.2.5 За Xi прийняти

$$i=1,2...n \left| \max_{1<j<n}(a_{ij}) - \min_{1<j<n}(a_{ij}) \right|$$

2.2.6 В матриці A знайти перший по черзі рядок з максимальною сумою його елементів. Вектор X отримати із знайденого рядка циклічним зрушенням його елементів на дві позиції ліворуч.

2.2.7 Рядки матриці відсортувати за зростанням тільки додатніх елементів і за вектор взяти елементи побічної діагоналі матриці A.

2.2.8 В матриці знайти найбільший по модулю елемент, та за (n-1)-ий елемент вектора прийняти (n-1) елемент того стовпця матриці, в якому знаходиться необхідний елемент. За Xn прийняти елемент A11.

2.2.9 За перші (n-1)-х елементів вектора X прийняти мінімальні елементи перших (n-1) стовпців матриці A. Елементи Xn покласти рівними сумі елементів останнього стовпця матриці.

2.2.10 Серед стовпців матриці A знайти стовпець, який містить мінімальний добуток своїх елементів і прийняти цей стовпець за вектор X.

2.2.11 Кожний рядок матриці відсортувати за зростанням його елементів, і за елемент X прийняти стовпець, який містить найбільший елемент.

2.2.12 Перший елемент вектора X покласти рівним сумі елементів матриці A, розташованих нижче головної діагоналі, за винятком Ann.

2.2.13 Останній елемент вектора X покласти рівним сумі елементів матриці A, розташованих над головною діагоналлю, за перші елементи прийняти елементи побічної діагоналі матриці, за винятком A1n.

2.2.14 За останній елемент вектора X взяти добуток елементів того рядка і того стовпця матриці, на перехресті яких знаходиться перший вид'ємний елемент матриці. Усі попередні елементи заповнити елементами стовпця, відкинувши останній.

2.2.15 За елементи вектора X прийняти елементи стовпця матриці, які мають мінімальну суму елементів та відсортованого у порядку зростання.

2.2.16 У кожному стовпці матриці знайти мінімальний та наступний за ним елемент прийняти за елемент вектора X. У випадку, якщо мінімальний елемент знаходиться в кінці стовпця, його прийняти за елемент вектора X.

2.2.17 Відсортувати стовпці матриці A за зменшенням та прийняти за вектор X строку з мінімальними елементами в стовпцях.

2.2.18 Відсортувати матрицю A, переставляючи стовпці так, щоб елементи першого рядка зменшувались. За вектор X прийняти елементи цього рядка у зворотньому по номеру стовпця порядку.

2.2.19 Вектор X заповнити сумами елементів кожного рядка матриці A, розміщених після мінімального елементу у відповідальному рядку. В отриманому векторі змінити знаки елементів на протилежні.

2.2.20 За елементи вектора X взяти попарні добутки елементів головної діагоналі на наступний елемент у рядку та відсортовані по зростанню. За останній елемент вектора X взяти елемент Ann.

2.2.21 Транспонувати матрицю A та її третій рядок взяти за вектор X.

2.2.22 У кожному стовпці матриці A поставити спочатку від'ємні елементи, а потім додатні. Елементами останнього рядка заповнити вектор X.

2.2.23 Компоненти вектора X – впорядковані за незростанням середніх арифметчних значень елементів рядків матриці A.

2.2.24 Транспонувати матрицю A, та заповнити елементами побічної діагоналі вектор X.

2.2.25 Елементи вектора X прийняти рівними максимальному елементу серед елементів матриці A, розміщених не нижче i-того рядка.

2.2.26 За елементи вектора X прийняти кількість додатніх елементів в i-тому стовпці матриці.

2.2.27 За елементи вектора X прийняти n/2 найбільших та n/2 найменших елементів матриці A.

2.2.28 За елементи вектора X прийняти n найменших елементів матриці A.

2.2.29 Знайти мінімальний елемент головної діагоналі матриці; елементи стовпця, в якому він знаходиться, відсортувати за незменшенням та прийняти за елементи вектора X.

2.2.30 Знайти максимальний елемент побічної діагоналі матриці; елементи рядка, в якому він знаходиться, відсортувати за незростанням і прийняти за елементи вектора X.

2.3 ВКАЗІВКИ ДО ВИКОНАННЯ ЗАВДАННЯ

Для відлагодження програм необхідно підготувати відповідний текст, тобто, вирішити задачу вручну, взяв таке значення n, яке не потребує великого об'єму обчислень. При цьому потрібно при рішенні на машині тестового варіанта задачі у програмі виконати усі передбачені в ній вітки обчислювального процесу. З цією метою для відлагодження частин програми можливо прийдеться задати інші правила обчислення матриці A, або значення компонент вектора X.

2.4 ЗМІСТ ЗВІТУ

2.4.1 Постанова задачі
2.4.2 Текст програми, вихідні дані.
2.4.3 Тести та результати наладок.
2.4.4 Результат рішення на ЕОМ.

Варіанти завдання отримати у викладача.

2.5 КОНТРОЛЬНІ ЗАПИТАННЯ

2.5.1 У чому різниця фізичного і логічного подання даних у багатомірному масиві?

2.5.2 Як здійснюється у машині лінеаризація багатомірного масиву (якими засобами)?

2.5.3 Для чого використовується функція впорядкування?

2.5.4 Що є аргументом функції впорядкування?

2.5.5 Як обчислити адресу похідного елементу багатомірного масиву?

2.5.6 Що представляє собою розріджений масив?

3 ЛАБОРАТОРНА РОБОТА №2

ЗАПИСИ, МАСИВИ ЗАПИСІВ

Мета роботи: вивчення структури даних-запису як кінцевої впорядкованої кількості елементів, що характеризуються різними типами даних. Вивчення ієрархічної структури записів, її подання на логічному та машинному рівнях.

3.1 ЗАВДАННЯ

Описати масив записів, що містять в собі відомості про кожного студента вашої групи.

3.2 ВКАЗІВКИ ДО ВИКОНАННЯ ЗАВДАННЯ

Вихідні дані треба виводити з текстового файла, за допомогою текстового редактора, в програмі організувати ввод з цього файла, а не з екрана.

Для відкриття текстового файла LAB2.DAT в програмі необхідно:

1) Описати файлову змінну типа TEXT,
   VAR F:TEXT;
2) Назначити файловій змінній ім'я файлу
   ASSIGN (F,'LAB2.DAT);
3) Відкрити файл
   RESET(F);

При читанні змінних типу "запис" можна читати тільки значення молодших полів запису.

3.3 ВАРІАНТИ ЗАВДАННЯ

3.3.1 Проаналізувавши підсумки сесії у своїй групі, вирахувати середній бал.

3.3.2 Надрукувати список студентів, що здали сесію на "5".

3.3.3 Надрукувати список студентів групи, що отримали одну оцінку "4", а решта "5".

3.3.4 Надрукуйте список групи студентів, що здали сесії без "3".

3.3.5 По якому предмету у минулу сесію був самий високий середній бал, і по якому самий низький.

3.3.6 Упорядкувати список студентів за датою народження, з вказанням напроти кожного прізвища повної дати народження.

3.3.7 Видати список студентів, день народження у яких в травні та червні.

3.3.8 Написати список відмінників по математиці.

3.3.9 Надрукувати список хлопців і дівчат з своєї групи окремо.

3.3.10 Впорядкувати список студентів за датами народження.

3.3.11 Впорядкувати список іноземних студентів за датами народження.

3.3.12 Видати список студентів із зазначенням отриманої ними стипендії (125%, 115%, 100% ,"не отримують"). В графі поставити знак "+".

3.3.13 Видати списки студентів-іменинників за місяцями. У разі збігу дат народження перевагу нумерації віддати жінці, або особі з більш коротким прізвищем.

3.3.14 Видати списки студентів-земляків з указанням Ф.І.Б. та місця народження.

3.3.15 Видати списки студентів, які мають відмінні оцінки по кожному з предметів окремо.

3.3.16 Вирахувати підсумкову стипендію групи за поточний семестр.

3.3.17 Видати список студентів, які мають домашні телефони з указанням їх номерів.

3.3.18 Видати список студентів, що проживають у гуртожитку, з указанням номерів їх кімнат.

3.3.19 Впорядкувати список студентів за іменами в лексикографічному порядку. В разі збігу імен, перевагу віддати імені з більш коротким прізвищем.

3.3.20 Видати списки студентів, що проживають в містах та селах, окремо.

3.3.21 Видати прізвища п'яти студентів, що отримали в минулій сесії найбільшу кількість балів в порядку збільшення.

3.3.22 Видати список трьох студентів, що отримали в минулій сесії найменшу кількість балів, з указанням кількості балів проти кожного прізвища.

3.3.23 Вказати прізвища хлопців, зріст яких вище 170 см.

3.3.24 Вказати прізвища дівчат, зріст яких нижче 155 см.

3.3.25 Видати список студентів, впорядкованих за зростом.

3.3.26 Впорядкувати списки студентів по середньому балу у порядку збільшення.

3.3.27 Визначити, чи є в групі студенти з однаковим прізвищем.

3.3.28 Видати списки студентів у разі збігу імені.

3.3.29 Видати списки студентів, імена яких не повторюються.

3.3.30 Видати списки студентів, що мають однакові імена по батькові.

3.4 ЗМІСТ ЗВІТУ

3.4.1 Постанова задачі.

3.4.2 Программа та ісходні дані.

3.4.3 Результати рішення на ЕОМ.

3.5 КОНТРОЛЬНІ ЗАПИТАННЯ

3.5.1 Що називається записом?

3.5.2 Що з’являється елементом запису?

3.5.3 Яка ієрархічна структура запису?

3.5.4 Дати характеристику багаторівневому запису.

3.5.5 Що з’являється таблицею? Які формули організайії пам'яті?

3.5.6 Які існують методи доступу до кожного елементу таблиці?

3.5.7 Як здійснюється пошук заданого елементу, виключення заданого елементу з таблиці?

4 ЛАБОРОТОРНА РОБОТА №3

ТИПІЗОВАНІ ПОСЛІДОВНІ ФАЙЛИ

Мета роботи: вивчення способів організації файлів, як важливої структури даних, операції над файлами. Вивчення відношень компонентів ієрархічної структури файла в основній і зовнішній пам'яті обчислювальної системи.

4.1 ЗАВДАННЯ

4.1.1 Сформувати типізований послідовний файл f, компоненти якого обчислюються по формулі.

4.2.2 Вказані по варіантам дії над компонентами файла виконати в рамках процедури або функції, яка отримує файл в якості параметра.

4.2.3 Роздрукувати значення результуючих змінних та компонент всіх сформованих файлів.

Примітка:

Кількість компонент файла вибирають в розмірах від 10 до 50. Забороняється використовувати проміжні масиви, додактові файли та здійснювати прямий доступ до компонентів файлів.

Формули для формування файлів отримати у викладача.

4.2 ВАРІАНТИ ЗАВДАННЯ

4.2.1 Знайти найменше із значень компонент з парними номерами.

4.2.2 Знайти найбільше із значень компонент по модулю з непарними номерами.

4.2.3 Знайти кількість парних чисел серед компонент файла типа INTEGER.

4.2.4 Знайти кількість подвоєних непарних чисел серед компонент файла INTEGER.

4.2.5 Знайти кількість квадратів непарних чисел серед компонент файла INTEGER.

4.2.6 Записати файл із збереженням порядку слідування компонент файла f, значення яких не перевищують 1000.

4.2.7 Зформувати файли f1 і f2. Формулу для f2 взяти із слідуючого варіанту. Переписати із збереженням порядка слідування компонентів f1 і f2, і навпаки, використовуючи проміжний файл h.

4.2.8 Отримати у файлі A компоненти файла B цілого типу, які є цілими числами.

4.2.9 Отримати у файлі A компоненти файла B цілого типа, які діляться на 3 та не діляться на 7.

4.2.10 Отримати в файлі A компоненти файла B цілого типу, які є точними квадратами.

4.2.11 Записати в файл A всі парні числа файла B, а в файл C усі непарні.

4.2.12 Зформувати файли A і B. Формулу для формування A взяти із слюдуючого варіанту завдання. Записати в файл C спочатку компоненти файла B, а потім компоненти - A із збереженням порядку.

4.2.13 Додатково дано число P. Переписати послідовно компоненти файла B у файл A, зупинившись після першої компоненти, меншого P.

4.2.14 Записати у файл A із збереженням порядку слідування ті компоненти файла B цілого типу, слід за якими розміщаються компоненти, кратні 5-ти.

4.2.15 Записати в файл A компоненти файла B перед якими розміщуються компоненти кратні 3.

4.2.16 Записати всі компоненти файла B, які є кратними числами, в файл A, а останні в файл C.

4.2.17 Перевірити, чи впорядковані компоненти файла B за збільшенням.

4.2.18 Дано два файли A і B. Формулу для формування файла A взяти із слідуючого варіанту. Перевірити, чи еквівалентні ці файли?

4.2.19 Знайти значення передостанньої компоненти файла.

4.2.20 Знайти значення третьої від кінця компоненти файла.

4.2.21 Визначити суму компонент.

4.2.22 Знайти добуток компонент.

4.2.23 Визначити суму квадратів компонент.

4.2.24 Визначити модуль суми та квадрат добутку компонент.

4.2.25 Знайти найбільше із значень компонент.

4.2.26 Знайти суму найбільшого та найменшого з значень компонент.

4.2.27 Знайти суму від'ємних компонент.

4.2.28 Отримати копію файла B в файлі A.

4.2.29 Визначити добуток компонент файла, розміщених на позиціях з непарними номерами.

4.2.30 Знайти мінімальне значення серед компонет файлу, розміщених на позиціях з парними номерами.

4.2.31 Число компонент файла B повинно бути кратним 10-ти. Записати у файл A спочатку найбільше значення перших 10-ти компонент файлу B, потім слідуючих 10-ти і т.д.

4.2.32 Сформувати файли B1, B2, B3, B4. Формули для формування цих файлів взяти з інших варіантів. Організувати обмін компонентами між файлами відповідно зі схемою:

B1--->B3
B2--->B4
B3--->B2
B4--->B1

дозволяється використовувати тільки один допоміжний файл.

4.3 ЗМІСТ ЗВІТУ

4.3.1 Постановка задачі.

4.3.2 Роздрук текста програми, ісходних та результуючих даних.

4.4 КОНТРОЛЬНІ ЗАПИТАННЯ

4.3.1 Що є файлом як структури даних?

4.3.2 Відношення компонентів ієрархічної структури файла до основної та зовнішньої пам'яті ОС.

4.3.3 Поняття зовнішніх та внутрішніх сортувань.

4.3.4 У чому полягає ефективність сортування?

4.3.5 Методи сортування.

4.3.6 Способи визначення часових затрат сортування.

4.3.7 Як здійснюється пошук заданого елементу, виключення заданого елементу з таблиці?

5 ЛАБОРАТОРНА РОБОТА №4

ОРГАНІЗАЦІЯ ЧЕРГИ ЗА ДОПОМОГОЮ СПИСКУ

Мета роботи: метою роботи є вивчення основних операцій над списковою структурою-чергою: включення нового елементу, виключення елементу, перевірка поточної довжини та очищення.

5.1 ЗАВДАННЯ

Модифікувати програму таким чином, щоб дані які зберігаються у черзі, були типа запису (RECORD), який складається з двох полів. Вивід полів запису повинен виконуватись в один рядок.

Тип даних узгодити з викладачем.

```
Program siod 1.1;
  Uses Crt;
  Type
       el=^spis;                          {тип вказівника на елемент черги}
        spis=record                                  {тип элемента черги}
          inform:string;                          {рядок даних  элемента}
          next :el;                        {вказівник на наступний элемент}
     end:
   Var
       key:char;          {символ, введений з клавіатури за допомогою}
                                                    {небуферованого вводу}
        per :el;                                  {вказівник початку черги}
        pos :el;                                    {вказівник кінця черги}
        num:integer;                             {кількість елементів у черзі}

Procedure Clear_m(n:integer);
      {процедура очищення частини екрана}
      {n=1- очищується  половина ліворуч екрана (стовпчики 1-40)}
```

```
      {n=2- очищується права половина екрана (стовпчики 41-80)}
     Var
         i :integer;                          {координата курсора по вертикалі}
     Begin
        n:=(n-1*39+1);   {вказівник координати курсора по горизонталі}
       for i:=to 23 do                        {цикл очищення частини екрана}
          begin
            gotoXY(n,i);
            writeln('                                                    ');
          end;
   End;

Procedure Insoch;                             {процедура додання в чергу}
   Var
      n :el;                                  {елемент черги, що додається}
   Begin
       Clear_m(1);
          new(n);                                    {створення елементу черги}
        pos^.next:=n;      {додання створеного елементу в кінець черги}
         pos:=n;                  {встановлення вказівника кінця черги на}
       gotoXY(1,3);
       writeln('Введіть новий елемент:');
         readln(pos^.inform);    {ввод рядка даних останнього елементу}
       num:=num+1; {визначити  поточну кількість елементів у черзі}
     End;

  Procedure Outoch;                           {процедура вилучення з черги}
      Var
         dele: el;{вказівник на перший  елемент черги (що вилучається)}
          och: el;                          {вказівник на другий елемент черги}
      Begin
        if num>0 then
          begin
            Clear_m(1);
            gotoXY(1,3);
```

```
                dele:=per^.next;          {встановлення dele на перший}
                                                  {елемент черги}
              writeln('Узятий елемент:');
              writeln(dele^.inform);          {виведення рядка даних}
                                                  {першого елемента}
              och:=dele^.next;    {встановлення och на другий елемент}
                                                  {черги}
              per^.next:=och;         {встановлення вказівника початку}
                                                  {черги на другий елемент}
                Dispose(dele);            {вилучення першого елемента}
              num:=num-1;             {знаходження поточної кількості}
                                                  {елементів у черзі}
              if num = 0 then     {встановлення вказівника кінця черги}
                 pos:=per;            {на вказівник початку черги, якщо}
                                                  {в черзі немає елементів}
          end;
      End;

Procedure Print_och;                      {процедура друку змісту черги}
      Var
           och :el;                             {поточний елемент черги}
          i : integer;                  {координата курсора по вертикалі}
          j : integer;                        {лічильник елементів черги}
      Begin
        i:=4;
       Clear_m(2);
       gotoXY(40,3);
       writeln('Зміст черги :');
        {перевірка кількості елементів у черзі,  виведення елементів,}
                                          {якщо кількість не більше 17}
        if num<=17 then
          begin
            och:=per^.next;           {призначення першого елемента}
                                                  {черги поточним}
            for i:=1 to num do            {цикл друку елементів черги}
```

```
                begin
                   goto XY(40,i+3);
                      writln(och^ .inform);      {вивід рядка даних поточ-}
                                                          {ного елементу}
                      ocb:=och^.next:            {призначння наступного}
                                                          {елементу поточним}
                  end;
              end

           else                                  {друк елементів черги, якщо}
               begin                             {їх кількість перевищує 17}
                  och:=per^.next;       {призначення першого елемента}
                                                          {черги поточним}
                   for i:= 4 to 13 do               {цикл друку перших}
                        begin                              {10 елементів}
                      gotoXY(40.i);
                         write(och ^.inform);    {виведення рядка даних}
                                                          {поточного елементу}
                         och :=och^ .next;       {призначення наступного}
                                                          {елемента поточним}
             end;
        gotoXY(40,14);
        write('     . . .
         for j:=l to num-5 do     {пропуск наступних елементів черги,}
           begin                                    {крім останніх 5}
              och:=och^.next;
                 for i:= 15 to 20 do            {цикл друку останніх}
              begin                                    {5 елементів}
                 gotoXY(40,i);
                 write(och^.inform);
                 och:=och^.next;
              end;
           end;
End;
```

```
BEGIN
      TextBackGround(l);              {встановлення кольору фону екрана}
      TextColor(14);          {встановлення кольору виводимих символів}
    CIrScr;
     writeln(' ОЧЕРЕДЬ "+" -ДОБАВИТЬ "-" ИЗВЛЕЧЬ');
     TextColor(ll);
     new(per);                          {створення вказівника початку черги}
      per^.inform:='**********';              ініціалізація рядка даних}
                                                {вказівника початку черги}
      pos:=per;                    {встановлення вказівника кінця черги  на}
                                                      {вказівник  початку}
        num:=0;                                 {початкова довжина черги}
     key:=' ';
        while key <> chr(27) do                {цикл, що здійснює роботу з}
         begin        {чергою, вихід з циклу - натиснути клавишу <ESC>}
              while not KeyPressed do   {очікування   натиснення клавиши}
               key :=ReadKey;
               if key = chr(0) then                {ввод символа з клавіатури}
               key:=ReadKey;
                  if key <> chr(27) then                {робота з чергою, якщо}
                     begin                      {не натиснута клавиша <ESC>}
                        case key of                          {вибір виду роботи}
                       '+':insoch;
                        '-':outoch;
                      else
                        begin
                           Sound(3000);                  {звуковий  сигнал,}
                           Delay(128);{якщо натиснута будь-яка клавіша,}
                           NoSound;              {окрім '+', '-' або <ESC>}
                        end;
                     end;
               Print_och;                              {друк елементів черги}
            end;
         end;
END.
```

5.2 ЗМІСТ ЗВІТУ

5.2.1 Мета роботи.
5.2.2 Блок-схема алгоритма з пояснювальними коментаріями.
5.2.3 Схема фізичної структури черги.

5.3 КОНТРОЛЬНІ ЗАПИТАННЯ

5.3.1 Що називається списком ?
5.3.2 Що називається послідовним списком ?
5.3.3 Що називається чергою ?
5.3.4 Що означає переповнення черги ?
5.5.5 Які основні операції над чергою ?

6 ЛАБОРАТОРНА РОБОТА №5

ОРГАНІЗАЦІЯ СТЕКА ЗА ДОПОМОГОЮ СПИСКУ

Мета роботи: Метою роботи є вивчення логічної структури стеку, важливих операцій доступу до стека, очищення стека та перевірки стека.

6.1 ЗАВДАННЯ

Модифікувати програму таким чином, щоб дані, які зберігаються в стеку, були типу запису (RECORD), складеного з двох полів. Вивід полів запису повинен здійснюватися в один рядок.

Тип даних узгодити з викладачем.

```
 Program siod_1_1;                    {тип вказівника на елемент стеку}
    Uses Crt;                                    {тип елементу стеку}
 Type                                          {рядок даних  елементу}
       spis = "el;                    {вказівник на наступний елемент}
    el = record
      inf: string;
      next: spis;
    end;
 Var
      key: char;          {символ, введенний з клавіатури за допомогою}
                                                  {небуферованого вводу}
       tek: spis;                                 {вказівник вершини стеку}
      och : spis;  {вказівник на доданий або виключений елемент стеку}

Procedure clear_m(n: integer):      {процедура очищення частини экрану}
                  {n=1 очищується половина екрану ліворуч (стовбці 1-40)}
                   {n=2 очищується права половина екрану (стовбці 41-80)}
 Var
     i: integer;                          {координата курсора по вертикалі}
```

```
Begin
    n:=(n-1)*39+1;     {визначення координати курсора по горизонталі}
     for i:=3 to 23 do                    {цикл очищення частини екрану}
      begin
        gotoXY(n,i);
        writeln.('                                                    ');
       end;
 end;

   Procedure Instek;                  {процедура додання елементу в стек}
Begin
  clear_m(l);
  new(och);                                        {утворення елементу}
    och^.next:=tek;                  {додання утвореного елементу в стек}
    tek:=och;               {встановлення вказівника вершини стеку на}
                                                    {доданий елемент}
   gotoxy(l,3);
   writeln('Введите новий елемент:');
     readln(tek^.inf);                          {ввод рядка даних елементу}
End;
```

6.2 ЗМІСТ ЗВІТУ

6.2.1 Мета роботи.
6.2.2 Блок – схема алгоритму з пояснювальними коментаріями.
6.2.3 Схема фізичної структури стеку.

6.3 КОНТРОЛЬНІ ЗАПИТАННЯ

6.3.1 Що називається стеком?
6.3.2 Які основні операції над стеком?
6.3.3 Що називається вершиною стеку? Як вона адресується?
6.3.4 Як зберігається стек в пам’яті ЕОМ ?

7 ЛАБОРОТОРНА РОБОТА №6

РЕАЛІЗАЦІЯ ДВУНАПРАВЛЕНОГО ЦИКЛІЧНОГО СПИСКУ

Мета роботи: Метою роботи є вивчення основних операцій над списковою структурою - двунаправленим кільцевим списком: включенря нового елементу, виключення елементу, переміщення по списку.

Завдання до лабораторної роботи одержати у викладача.

7.1 ЗАВДАННЯ

```
Program spis_2c;
  Uses Crt,Graph;
  Const.
     Pi=3.1415826535;
   Type
         el = ^spis;                          {вказівник на елемент списку}
          spis = record                                {тип елементу списку}
          x,y: integer;{координати центру кола, відповідні елем-ту списку}
           inform: string;                  {рядок даних  елементу  списку}
                                                          {(суть елементу)}
            pred: el;                 вказівник на попередній елемент списку}
           posi: el;                 {вказівник на наступний елемент списку}
       end;
    Var
      Driver_Mode: integer;   {переміщення, використані для ініціалізації}
                                                    {графічного режиму дисплея}
       main: el;
       numb: integer;                              {кількість елементів списку}
       key: char;
         st: integer;                {символ, відповідний натиснутій клавіші}

Procedure Beep;
```

```
    Begin                          {процедура подачи звукового сигналу}
       Sound( 1000);                  {включення звукового сигналу}
        delay(128);                                    {затримка}
       Nosound;                       {виключення звукового сигналу}
  End;

Procedure Clr(xl,yl,x2,y2:integer);
                  {процедура очищення частини екрану,  заданої у вигляді}
                                                       {прямокутника}
              {x1 ,y1 - координати  верхнього кута прямокутника ліворуч}
               {x2,y2 - координати правого нижнього кута прямокутника}
Begin
   SetFillStyle(0,l);
   Bar(x1,y1,x2,y2);
End;

Procedure Setscr; {процедура вивода інструкції по роботі з програмою}
Begin
   Cleardevice;                                    {очищення екрана}
    OutTextXY(10,10,' ПРОГРАМА РЕАЛІЗУЄ ДВУНАПРАВЛЕНИЙ
                        ЦИКЛІЧНИЙ СПИСОК');
     OutTextXY(20,20,'i-додати елемент d-знищити елемент 1-перейти на
                                                  елемент по ч.с.');
 OutTextXY(20,30,' г - перейти на елемент проти ч.с. Enter - ввести суть
                        елемента');
    OutTextXY(410.60,'Суть елемента:');
End:

Procedure Inp_inf;                 {процедура ввода суті елементу}
Begin
     OutTextXY(410,120,'Введіть суть елемента:');     {вивод запрошення}
      GotoXY(55,17);                          {позиціонування курсору}
      readln(main^.inform);                          {ввод суті елементу}
     clr(410.120,640,135);          {витирання запрошення і введеної суті}
                                                       {елементу}
```

```
End;

Procedure Out_inf;                          {процедура виводу суті елементу}
Begin
      clr(410,70,640.80);                        {підготовка экрану до виводу}
      OutTxtXY(410,70,main^. inform);                 {вивод суті  елементу}
End;

Procedure Ch_inf; {процедура ввода суті елементу и виводу її на екран}
Begin
   inp_uif;
   out_mf;
End;

Procedure Mal_sp;
     {Процедура виводить на екран геометричного уявлення списку }
     {Список поданий у вигляді кільця, елементи списку - у вигляді }
        {кіл, які лежать на кільці. Поточний елемент позначається }
                {символом '*' - міткою поточного елементу}
Var
       tek : el;                      {вказівник на поточний елемент списку}
      st : real;                       {величина кута між сусідніми колами,}
                                          {якими позначені елементи списку}
       i : integer;                                  {керуюча змінна циклу}
Begin
    clr( 1,60,400,200);
     Circle(150,130,118);                           {вичерчування кільця}
    Circle(150,130,142);
    st:=2*Pi/numb;
      tek:=maili;{встановлення вказівника tek на поточний елем-т списку}
     for i:=l to numb do      {цикл вичерчування кола - елементів списку}
        begin
           tek^.x:=Trunc(130*cos((i-l)*st)+150);
                  {обчислювання координат по горизонталі  та  вертикалі}
           tek^:y:=Trunc(54*sin((i-l)*st)+130);
```

```
            Circlc(tclt^.x,tek^.y,10);          {вичерчування  кола елементу}
             tck:=tck^.posl;                    {перехід до наступного елементу}
        end;
      OutTextXY(main^.x-3,main^.y-3,'*');   {виведення мітки поточного}
End;                                                          {елементу}

 Procedure Left_right(napr:char);      {процедура переміщення за списком}
Begin
      SetColor(0);                          {зтирання мітки поточного елементу}
    OutTextXY(main^Jt-3,main^.y-3 ,'*');
    SetColor(l);
       if napr=T                              {вибір напрямку руху за  списком}
       then
            main:=main^ .posi;                {перехід до наступного елементу}
       else
          main:=main^ .pred;              {перехід до попереднього елементу}
OutTextXY(main^ .x-3,main^ -y-3 ,'*');
                                          {виведення мітки поточного елементу}
     cild;

Procedure Plus_l ;
   {процедура включення нового елемента між поточним та наступним}
Var
        r:el;                          {вказівник на наступний елемент списку}
         n:el;                             {вказівник на новий елемент списку}
Begin
      r:=main^ .posl;    {встановлення вказівника r на наступний елемент}
                                                                  {списку}
        new(n);                           {утворення нового елементу списку}
     main^ .posl:=n;    {формування зв'язку поточного елементу з новим}
    n^.pred:=main;
     n^.posi:=r;         {формування зв'язку нового елементу з наступним}
    r^.pred:=n;
    n^.inform :='';      {ініціалізація рядка даних нового елементу списку}
      main:=n.;    {призначення включеного елементу списку наступним}
```

```
        mal_sp;                  {виведення на екран поточного стану списку}
End;

Procedure Minus_l;
                          {процедура знищення поточного елементу з списка}
Var
        l: el;                   {вказівник на попередній елемент списку}
        r: el;                   {вказівник на наступний  елемент списку}
Begin
      l:=main^.pred;   {встановлення вказівника 1 на попередній елемент}
                                                              {списку}
     r:=main^ .posi;    {встановлення вказівника r на наступний елемент}
                                                              {списку}
       Dispose (main);                    {знищення поточного елементу}
    l^.posl:=r;          {формування зв'язку між попереднім та наступним}
      r^.pred:=l;                                    { елементами списку}
     main:=l;              {призначення  попереднього елемента поточним}
     mal_sp;                 {виведення на екран поточного стану списку}
End;

BEGIN
   Driver:=Dctect ;
    Initgraph(Driver_Mode,' ');          {ініціалізація графічного режиму}
                                                        {роботи екрану}
   setscr;
     numb:=l;                          {похідна кількість елементів списка}
      new(main);                      {утворення першого елементу списка}
     main^.posl:=main;                      {утворення циклічного списку}
    main^ .pred:=main;
     main^.inform:=' ';                  {ініціалізація рядка даних елементу}
      mal_sp;                            {виведення поточного стану списку}
   key:=' ':
    while key<>chr(27) do       {цикл обробки натиснутої клавіші, вихід}
        begin                  {з циклу при натисненні клавіші <КЛЮЧ>}
         out_inf;
```

```
        while not KeyPressed do        {очікування натиснення клавіші}
         key:=ReadKey;
           if key=chr(0) then              {ввод символа,  відповідного}
            begin
                 key:=ReadKey;                          {натиснутій клавіші}
                  case key of                           {вибір виду обробки}
                  'r': left_right('r'); {переміщення за списком проти г.с.}
                  'l': left_right(T);      {переміщення за списком за г.с.}
                    #13 : ch_inf;                       {ввод суті елементу}
                  'd': if numb<>l          {перевірка кількості елементів}
                       then
                            begin {якщо кількість елементів більше 1 ,то}
                             numb:=numb-l;
                                   {зменшення кількості елементів на 1}
                               minus^l;   {усування поточного елементу}
                           end
                          else {якщо список складається з 1 елементу, то}
                           beep;
                                {подання звукового сигналу про помилку}
                      'i': begin                        {додаток елементу}
                          numb:=numb+l;
                                {збільшення кількості  елементів на 1}
                           plus_l;          {включенння нового елементу}
                        end;
                    #27:;
                 else
                      beep;         {подання звукового сигналу у випадку}
                  end;                  {натиснення невикористаної клавіші}
          end;
     CloseGraph:          {знищення графічного режиму роботи екрану}
End.
```

7.2 ЗМІСТ ЗВІТУ

7.2.1 Мета роботи.

7.2.2 Блок-схема алгоритму з коментаріями

7.2.3 Схема фізичної структури двунаправленого кільцевого списку.

7.3 КОНТРОЛЬНІ ЗАПИТАННЯ

7.3.1 Що називається списком ?

7.3.2 Що називається двунаправленим кільцевим списком ?

7.3.3 Як включається новий елемент до списку ?

7.3.4 Як виключається елемент зі списку ?

8 ЛАБОРАТОРНА РОБОТА №7

ЗАСТОСУВАННЯ БІНАРНИХ ДЕРЕВ

Мета роботи: метою роботи є вивчення методів знаходження одного рішення з двох можливих із застосуванням бінарних дерев.

8.1 ЗАВДАННЯ

8.1.1 Вивчити операції над бінарними деревами.

8.1.2 Розробить програму знаходження усіх дублікатів у списку чисел.

8.1.3 Розробить програму знаходження бінарних дерев.

8.1.4 Розробить програму подання виразу, утримуючого операнди та бінарні оператори у виді строго бінарного дерева.

Варіанти завдання отримати у викладача.

8.2 ОПИСАННЯ МЕТОДИКИ ЗАСТОСУВАННЯ БІНАРНИХ ДЕРЕВ

8.2.1 Алгоритми і процеси, що використовують бінарні дерева, роспадаються на дві фази. У першій фазі будується бінарне дерево, а у другій воно проходиться.

8.2.2 Вхідний файл, містить список чисел, який необхідно надрукувати у возрастающому порядку.

8.2.3 Знаходження дублікатів застосовується шляхом використання бінарного дерева: зчитується перше число та поміщується у вузол, який стає корнем бінарного дерева. Якщо значення співпадають, тоді це повторюється для пів-дерева ліворуч,а якщо більше, тоді для правого продовжується , поки не зустрінеться дублікат, або пока ми не достигнем пустого дерева в останньому.

Приклад**:**

Дана послідовність чисел:
14 15 4 9 7 18 3 5 16 4 20 17 9 14 5

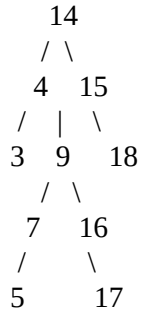

Рис. 8.2.1 Бінарне дерево, побудувано для знаходження дублікатів

8.3 ЗМІСТ ЗВІТУ

8.3.1 Мета роботи.
8.3.2 Описання методів примінення бінарних дерев.
8.3.3 Алгоритм.
8.3.4 Текст програми.

8.4 КОНТРОЛЬНІ ПИТАННЯ

8.4.1 Яке дерево є бінарним ?
8.4.2 Класіфікация бінарних дерев.
8.4.3 Операції над бінарними деревами.
8.4.4 Примінення бінарних дерев.
8.4.5 Методи проходження бінарних дерев.
8.4.6 Подання списків у вигляді бінарних дерев.
8.4.7 Як застосовується пошук заданого елементу, знищення заданого елементу з таблиці ?

9 ЛАБОРАТОРНА РОБОТА № 8

ВИВЧЕННЯ ЗАСОБІВ ФОРМУВАННЯ ВИХІДНИХ ДАНИХ ДЛЯ ПОБУДОВИ МАТЕМАТИЧНИХ МОДЕЛЕЙ

Мета роботи: вивчити засоби отримання вихідних даних для складання ймовірносних моделей та для складання програм формування файлів вихідних даних для них.

9.1 ЗАВДАННЯ

9.1.1 Вивчити методи вводу результатів спостережень.

9.1.2 Розробити програму вводу результатів спостережень.

9.1.3 Розробити процедури коригування файлів результатами спостережень.

9.1.4 Розробити програми по п. 2.2, 2.3.

9.1.5 Розробити програму роздруку файлів з результатами спостережень.

9.1.6 Перевірити роботу програми на конкретних даних , наданих викладачем.

9.2 ОСНОВНІ ТЕОРЕТИЧНІ ПОЛОЖЕННЯ.

9.2.1 Постановка задачи статистичного моделювання (на прикладі технологічних процесів).

Математичному моделюванню передує статистичний аналіз.

Статистичний аналіз технологічного процесу – це визначення статистичними методами характеристик, закономірностей протікання в часі і причин змінення параметра процеса.

Технологічний процес, як об’єкт досліджування, відноситься до складних стохастичних систем. Стохастичні властивості технологічних процесів доволі різноманітні і з них найбільш важливе значення мають ті властивості, які визначають якість виробляємої продукції. До таких властивостей, перед усього, треба віднести

стабільність розподілень параметрів виготовляємих виробів. Один з методів забезпечення стабільності розподілень – це впровадження нових технологічних рішень, змінення режимів роботи обладнання, підвищення кваліфікації робітників. Для оцінки ефективності внесення змінень в технологічний процес, для виснування о причинах погіршення або покращення якості прдукції головне застосування знайшли слідуючі методи математичної статистики:

перевірка статичних гіпотез;
кореляційний аналіз;
регресійний аналіз;
дисперсійний аналіз.

Використання цих методів дозволяє вирішувати низку задач статистичного аналізу, а саме:

оцінка узгодження виробничих розподілень з теоретичними;

визначення наявності суттєвих розбіжностей між виробничими розподіленнями;

оцінка однородності послідовності вибіркових середніх
(стабільності технологічного процесу);

оцінка впливу відмінностей в обладнанні, кваліфікації робітників і вихідних матеріалів розкид параметрів виготовляємих виробів;

визначення відмінностей і сили кореляційних зв'язків між технологічними факторами та параметрами виготовляємих виробів;

побудова регрессійних моделей технологічних процессів.

9.2.2 Форми та характер представлення даних при статистичному моделюванні.

Результати спостережень, в залежності від розв'язуємої задачі, подаються у вигляді одно – , двух - або трьохмерним масивом і записуються окремим файлом на магнитних диску або стрічці. Необхідні для розв'язання кожної задачі вихідні дані, включаючи і назву файла з результатами спостережень, вводятся з відеотерміналу під час диалога ЕОМ з користувачем.

Змінні, характерізуючі технологічний процес і виготовляєму продукцію відносятся до випадкових величін.

Випадковой величиною зветься змінна, яка може приймати ті або інші числові значення у залежності від різноманітних випадкових обставин. Результати спостережень за параметрами технологічного процесу і виготовляємой продукцією утворюють вибірку, яка характерізується центром групування, розсіювання і розподілення.

9.3 ОПИС МЕТОДІКИ І АЛГОРИТМІВ ФОРМИРУВАННЯ ВИХІДНИХ ДАНИХ

9.3.1 Файли вихідних даних формірюються з допомогою програми VVOD, пертворюються і коректуються програмой KRF і розпечатуються програмой RSP.

9.3.2 Опис логічної структури, вхідні та вихідні дані.

ПРОГРАМА VVOD.

Алгоритм програми складається з слідуючих дій:

1. Вводится вид масива як перемінна NR, визначаюча розмір масива.

2.Для тривимірового масива вводится число рівней NK, для двовимірового – число стілбців NJ.

3. Вводится число рівней NI для масива будь-який розмірності.

4. Вихідні дані заносяться як елементи масива A(N), де

N=I + NI*(J – 1) + NI*NJ*(K – 1); I = 1, NI; J = 1,NJ; K = 1,NK

5.При необхідності масив A(N) роздруковується на АЦПУ, при знаходженні помилок вона виправляється.

6.Вводиться назва для файлу з масивом A(N).

Програма KRF.

Алгоритм програми складається зі слідуючих дій:

1.Вказується назва корегуємого файлу та вигляд масиву.

2.Вводиться кількість рівней NK для трьох мірних масивів, кількість стовбців для двох та трьох мірних масивів і кількість нагляданнь у стовбці NI.

3.При необхідності виправляється будь-який елемент масива, виключаються та довводяться наглядання.

9.4 МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ.

9.4.1 Для виконання пп. 2.1, 2.2 необхідно ознайомитися із конспектом лекцій, уяснити значення принципів вводу вихідних даних.

9.4.2 Для виконання пп. 2.3, 2.4 необхідно старанно вивчити методіку вказаних процедур.

9.4.3 Для виконання пп. 2.5 отримати у викладача вихідні дані, після розробки програми отримати роздрук файлу із результатами спостереженнь.

9.5 ЗМІСТ ЗВІТУ.

9.5.1 Ціль роботи.
9.5.2 Опис логічної структури процедури: Вхідні дані.
9.5.3 Блок-схема алгоритму.
9.5.4 Тексти програм вводу, корегування, роздруку.

9.6 КОНТРОЛЬНІ ПИТАННЯ.

9.6.1 Поняття математичної моделі.
9.6.2 Види математичних моделей.
9.6.3 Яка модель виявляється ймовірносною?
9.6.4 Яка модель виявляється детермінованою?
9.6.5 Поняття випадкової величини.
9.6.6 Що таке залежна та незалежна випадкова величина?
9.6.7 Поняття статистичної ймовірності.
9.6.8Дискретні та неперервні випадкові величини.

10 ЛАБОРАТОРНА РОБОТА № 9

ВИВЧЕННЯ МОДЕЛЕЙ РОЗПОДІЛЕНЬ. ОБЧИСЛЕННЯ ПАРАМЕТРІВ РОЗПОДІЛЕНЬ.

Мета роботи: метою цієї роботи є вивчення методів оцінки узгодження виробничіх розподілень з теоретичними, визначення наявності суттєвих різниць між виробничими розподіленнями, побудова програм за вказаними методами.

10.1 ЗАВДАННЯ

10.1.1 Вивчити методи побудови виробничих розподілень та оцінки їх з теоретичними;

10.1.2 Розробити блок-схему алгоритму розрахунку параметрів розподілень;

10.1.3 Розробити блок-схему алгоритму оцінки узгодження емпіричного та теоретичного розподілення за критеріями;

10.1.4 За блок-схемами пп.2.2, 2.3 розробити програми, виконати контрольні розрахунки.

10.2 ГОЛОВНІ ТЕОРЕТИЧНІ ПОЛОЖЕННЯ.

10.2.1 Постановка задачи знаходження параметрів розподілу.

Для оцінки центру групування вибірки випадкових величин використовується:

середнє арифметичне;

медіана;

мода.

Середнє арифметичне $\bar{x}$ для вибірки випадкових величин X1, X2, ..., XN розраховується за формулою:

$$X = \frac{1}{N}\sum_{i=1}^{N} X_i$$

Медіана дорівнює середньому члену вибірки випадкових чисел X1, X2, ..., XN поставлених у зростаючому порядку, при непарном N. Якщо N – парне, медіана дорівнює півсумі двох середніх членів. Модой зветься таке значення випадкової величини, ймовірість отримання якого найбільша.

Розсіювання випадкових величин у виборці характеризується:

емпіричною дисперсією;

розмахом.

Емпірична дисперсія S# для вибірки випадкових величин X1, X2,...

XN розраховується за формулою:

$$S^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(x_i - \overline{x}\right)^2$$

Розмах розраховується як різниця максимального та мінімального члена вибірки.

Розподілення випадкової величини зветься розміщена за зростанням сукупність значеннь випадкових величин із вказівкою ймовірності їх виникання.

Емпіричне розподілення може бути представлено у вигляді гістограми, тобто ступенчатого графіку, ординати якого співпадають з частістю влучання даної випадкової величини у інтервал значеннь, відложений по вісі абсцис.

При построєнні гістограм частість Fi й кількість інтервалів розраховується за формулою:

$$Fi = \frac{Ni}{N}; i = 1,2,...,K; K = 1 + 3.32 * Lg(N)$$

де: Ni – кількість влученнь значення випадкової величини до і-го інтервалу;

N – об’єм вибірки.

Емпіричне розподілення випадкової величини X може бути описано аналітично за допомогою функції плотності ймовірності f(X) теоретичного розподілення.

10.2.2 Опис методіки перевірки гіпотези про закон розподілення.

Нехай надана вибірка незалежних випадкових чисел X1, X2, ..., XN маючих емпіричне розподілення Fn(X). Для перевірки узгодження емпіричного розподілення Fn(X) з теоретичним розподіленням F(X) використовується У – критерій Колмогорова та nw – критерій Мізеса – Смірнова. Попередня оцінка виду розподілення може бути зроблена порівнянням гістограм емпіричного та теоретичного розподіленнь.

Перевірка за допомогою критерія Колмогорова виконується наступним чином. Вибіркова послідовність {Xi} впорядковується у незменшуюцуюся послідовність {Xk}. Розраховується абсолютне значення максимального розходження Dnміж емпіричним та теоретичним розподіленням:

$$Dn = \max \mid Fn(\underset{1 \le K \le N}{Xk}) - F(Xk)$$

З отриманого значення Dn розраховується величина

$$Y = \sqrt{12} * Dn$$

яка порівнюється з критичним значенням.

IV. Опис методіки.

Необхідно розробити алгоритми та програми:

Програма RSP.

Алгоритм програми це організація друку масива з результатами спостереженнь по вказаним видам масива й назві файлу.

Програма PZR.

Алгоритм програми складається із слідуючих дій:

Вводиться код розподілення IFCOD, об’єм вибірки N, кількість інтервалів M, параметри розподілення a,b,c, верхня межа AHI, й нижня межа ALO для побудови гістограми;

Результати спостереженнь A1, i=1,N, вводяться з файлу, де записан масив Ai, i=1,N.

Розраховуються:

суми $$S1 = \sum_{i=1}^{N} A1;\ \ S2 = \sum_{i=1}^{N} A12;\ \ S3 = \sum_{i=1}^{N} A13$$

моменти $$m1 = \frac{S1}{N};\ \ m2 = \frac{S2}{N};\ \ m3 = \frac{S3}{N}$$

центральні моменти

$$M1 = m1;\ \ M2 = m2 - m1^2;\ \ M3 = m3 - 3 * m2 * m1 + 2 * m1^3$$

середне $\overline{X} = M1$

дисперсія $$S^2 = \left(N / (N-1) \right) * M2$$

асиметрія $$A = M3 / S^3$$

Розраховується крок STEP, верхні BHI(K) та нижні BLO(K) границі інтервалов

STEP = (AHI-ALO) / M;
BLO(K) = ALO + STEP * (K-1);
BHI(K) = ALO + STEP * K;
K=1, M;

Визначається кількість та накопичена кількість, процент й накопичений процент спостереженнь, влучивших в інтервал [BLO(K), BHI(K)], при K = 1, M для чого розробити програму FRQ.

Оцінюється узгодження емпіричного та теоретичного розподіленнь за крітеріями Колмогорова та Мізес – Смірнова, для чого розробляється підпрограма (процедура).

Програма TUTEST.

Алгоритм складається із слідуючих дій:

Вводяться результати спостереженнь з двох, в кожном з яких знаходяться двохмірні масиви з елементами

A(I,J), I = 1, NA; B(I,J), I = 1, NB; J = 1, NJ, де NA, NB – об'єми порівнюємих вибірок, NJ – число порівнюємих вибірок.

Розраховуються квантіли розподіленнь Фішера, Ст'юдента, Гауса виходячи з даної довірчої ймовірностіза допомогою п/програм QNTF, QNT.

Розраховуютьсясередні та дисперсії вибірок типа A та B для кожної пари вибірок.

Розраховується T – критерій за п/програмою TTEST та U – критерій за п/програмою UTEST.

Порівнянням розрахованих значенньстатистичних критеріїв з їх табличними значеннями формується заключення о різниці вибірок для кожної пари.

10.3 МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ.

Для виконання п. 2.1., 2.2. необхідно ознайомитися із конспектом лекцій, вивчити методіку побудови густостей розподілення випадкових величин.

Для виконання п. 2.2., 2.3. вивчити методіку розрахунку параметрів розподіленнь.

Програми по п. 2.2., 2.3. розробити з врахуванням методіки вводу вихідних даних лаб.№1.

10.4 ЗМІСТ ЗВІТУ

10.4.1 Мета роботи.

10.4.2 Опис логічної структури та блок – схеми алгоритму: розрахунку параметрів виробничих розподілень, оцінки узгоджень емпіричного та теоретичного розподілень.

10.4.3 Тексти програм за п. 2.4.

10.5 КОНТРОЛЬНІ ПИТАННЯ

10.5.1 Що називається розподіленням випадкової величини?

10.5.2 Поняття теоретичного та емпіричного розподілень.

10.5.3 Як виконати оцінку узгодження емпіричного розподілення з теоретичним.

10.5.4Критерії узгодження.

10.5.5 Дати визначення диференціальній функції розподілення випадкової величини.

10.5.6 Інтегральна функція розподілення.

10.5.7 Густість ймовірності розподілення випадкової величини.

10.5.8 Закон нормального розподілення та області його використання у математичному маделюванні.

10.5.9 Дати визначення початкового та центрального моментів розподілення.

10.5.10 Поняття математичного очікування та середнього арифметичного значення випадкової величини

10.5.11 Поняття дисперсії та середнього квадратичноговідхилення випадкової величини.

10.5.12 Що представляє собою гістограма та як вона будується.

11 ЛАБОРАТОРНА РОБОТА №10

ПОБУДОВА МАТЕМАТИЧНИХ МОДЕЛЕЙ З ВИКОРИСТАННЯМ КОРЕЛЯЦІЙНОГО АНАЛІЗУ ТА МНОЖИ ННОЙ ЛІНІЙНОЙ РЕГРЕСІЇ

Мета роботи: метою роботи є вивчення методів побудови математичних моделей з використанням регресійного аналізу, складання програм для даних методів.

11.1 ЗАВДАННЯ

11.1.1 Вивчити методи кореляційного та регресивного аналізів, використання їх в побудові математичних моделей.

11.1.2 Розробити блок-схему алгоритму розрахунку коефіцієнтів кореляції між незалежними змінними та вектор взаємних кореляцій незалежних змінних із залежними змінними.

11.1.3 Розробити програми по п.2.2.

11.2 ОСНОВНІ ТЕОРЕТИЧНІ ПОЛОЖЕННЯ.

11.2.1 Постановка задачи побудови математичних моделей з використанням кореляційного та регресивного аналізів.

Між випадковими величинами може існувати стохастичний зв'язок.

При цьому одна з випадкових величин реагує на змінення іншої зміненням своєї функції розподілення.

Кореляційний зв'язок – це окремий випадок стохастичного зв'язку. При цьому змінення однієї випадкової величини змінює математичне очікування іншої. При кореляційному аналізі вирішуються три головних задачи – визначається наявність зв'язку, форма зв'язку та сила зв'язку.

Для визначення наявності лінійного зв'язку між двома випадковими величинами за результатами вибірки розраховується коефіцієнт кореляції r.

Чим ближче r до ±1, тим кореляційний зв'язок сильніший і при r =±1 кореляційний зв'язок переходить у функціональний.

Для оцінки зв'язку необхідно побудувати кореляційне поле

(графічне відображення залежності досліджуємих випадкових величин), групування точок навколо якої-небудь кривої вказує на наявність зв'язку.

Якщо на одну випадкову величину, яка зветься залежною, оказують вплив декілька випадкових величин, які звуться незалежними, то такий зв'язок оцінюється множинним коефіцієнтом кореляції.

Одним із засобів оцінки впливу на випадкову величину декількох незалежних змінних є множинна лінійна регресія.

11.2.2 Опис алгоритму побудови множинного лінійної регресії.

Нехай випадкова величина Y залежить від змінних x1, x2, ...,xk, які приймають в кожному спостереженні визначенні значення.Якщо провести серію з n спостереженнь, то результати спостереженнь можна представитиу слідуючому вигляді:

1-е спостереження	y1	x11,x12,x13,...,x1k
2-е спостереження	y1	x21,x22,x23,...,x2k
3-е спостереження	y1	x31,x32,x33,...,x3k
.....		
N-е спостереження	y1	xN1,xN2,xN3,...,xNk

Залежність випадкової величини y від незалежних змінних x1, x2, ..., xk в-ідображується у рівнянні множ1инної регресії:

$$y=b0+b1*x1+b2*x2+...+bk*xk;$$

Для отримання оцінок B0, B1, B2, ..., Bk коефіцієнтів регресії b0, b1, b2, ..., bk використовується метод найменших квадратів.

Оцінки Bs, s=0, k отримуються з рішення систем рівняннь, утворених приравнюванням до 0 окремих похідних за bs формулою:

$$Q = \sum_{i=1}^{N} \left(y_i - b_0 - \sum_{i=1}^{K} b_s * x_i s \right)$$

Для оцінки параметра B0 отримаємо:

$$B_0 = \frac{1}{N} * \sum_{i=1}^{N} y_i$$

Параметр B0 зветься свободним членом та визначає перетин координати при рівних 0 незалежних змінних. Вираз для оцінок Bs, s=1, k , визначаэться наступним чином. Введемо позначки:

$$Lrs = \frac{1}{N} * \sum_{i=1}^{N} \left(xr_i - \bar{xn} \right) * \left(xs_i - \bar{xs} \right); \qquad 1<<r<<s<<K$$

$$L_0 s = \frac{1}{N} * \sum_{i=1}^{N} \left(y_i - \bar{y} \right) * \left(xs_i - \bar{xs} \right);$$

Елементи Lrs образують матрицю, визначник якої позначимо Z та, якщо позначити Zs визначник матриці, отримуємой заміною S-го стовбця L0s, S=1, k то:

$$Bs = \frac{Zs}{Z}$$

На основі елементів матриці розраховуються парні коефіцієнти кореляції ris між незалежними змінними xs, i,s=1, k та коефіцієнти кореляції ri0 між залежной змінной y та незалежними змінними. Елементи ris складають кореляційну матрицю. Виходячи з парних коефіцієнтів кореляції розраховується коефіцієнт множинной кореляції

$$R = \sqrt{\sum_{i=1}^{K} \frac{1}{r_{ij}} * r_i o}$$

де - $r_i o$ елементи матриці, зворотньої кореляційній матриці.

11.3 ОПИС МЕТОДІКИ ТА АЛГОРИТМУ ПРОГРАММ ПО БУДОВИ РЕГРЕСІЙНИХ МОДЕЛЕЙ.

Методіка реалізується за допомогою програми, якій присвоюється назва RGR.

11.3.1 Вводиться алфавітно-цифрове найменування задачи, номер задачи, число спостережень у стовпці N, число змінних M, число варіантів NS, ознака друку таблиці залишків NRESI номера стовпця незалежної змінної, k - число незалежних змінних ISAVE.

Звернення до підпрограми CORRE, у якій зчитуються результати спостережень Xij, X=1,N, j=1,M; розраховуються середні Xj, середнє – квадратичні Sj, суми взаємних відхиленнь Tj, коефіцієнтів кореляції Rjk.

Організується цикл по числу варіантів розрахунку.

Звернення до підпрограми ORDER, у якій із повної матриці коєфіцієнтів кореляції будується підматриця взаємної кореляції між незалежними змінними і вектор взаємних кореляцій незалежних перемінних із залежними перемінними. Утворюється матриця Rx(K,K) із взаємних коефіцієнтів кореляції між незалежними змінними, а масив Ry(K) із коефіцієнтів кореляції між залежною змінною і незалежними змінними.

Звернення до підпрограми MINV, у якій відбувається звернення матриці Rx(K,K) та розраховується визначник DET.

Звернення до підпрограми ORDER, у якій відбуваються розрахунки множинної лінійної регресії:

розраховуються коефіцієнти регрессії за методом найменших квадратів

$$b_j = p_i * \frac{S_y}{S_j}$$

де Sy – стандартне відхилення залежної змінної; Sj - стандартне відхилення j-ї незалежної змінної, j=1,K

визначається перетин

$$b_0 = \bar{y} * \sum_{j=1}^{k} b_j * \bar{x}_j$$

де $\bar{y}$, $\bar{x}$ – середнє залежної та j-ої незалежної змінних

- розраховуються суми квадратів

$$SSAR = R^2 * D_{yy}$$

$$SSDR = D_{yy} - SSAR$$

розраховується F- критерій

$$F = \frac{SSAR / K}{SSDR / (N - K - 1)}$$

розраховується стандартна похибка оцінки

$$S_{y,1,2,\ldots k} = \sqrt{\frac{SSDR}{(N - K - 1)}}$$

розраховується стандартне відхилення коефіцієнтів регресії

$$S_{bj} = \sqrt{\frac{S_{y,1,2,\ldots k}}{(r_{ij} * D_{jj})}}$$

де Djj – сума квадратів відхилень від середнього для j-ої незалежної змінної

розраховуються t-критерії

$$t_j = \frac{b_j}{S_{bj}}; j = \overline{1,k}$$

Формування та друк вихідної машинограми.

11.4 МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ

Для виконання пп.2.2 необхідно ознайомитися із конспектом лекцій, вивчити методи побудови математичних моделей з використанням регресійного аналізу.

Для виконання пп.2.2 необхідно вивчити методи розрахунку коефіцієнтів кореляції.

Програми на пп. 2.2, 2.3 розробити з урахуванням вводу вихідних даних лаб.№1.

11.5 ЗМІСТ ЗВІТУ.

11.5.1 Мета роботи.

11.5.2 Опис методів кореляційного та регресійного аналізів, побудова математичних моделей.

11.5.3 Тексти програм пп.2.3.

11.6 КОНТРОЛЬНІ ПИТАННЯ

11.6.1 Сформулювати задачу побудови математичної моделі на основі кореляційного та регресійного аналізів.

11.6.2 У чому заключається задача кореляційного аналізу.

11.6.3 У чому заключається задача регресійного аналізу.

11.6.4 Коефіцієнт регресії та його властивості.

11.6.5 Поняття кореляційного відхилення. Його властивості.

11.6.6 Кореляційна таблиця. Методи її побудови.

11.6.7 Як розраховуються коефіцієнти рівняння регресії.

11.6.8 Рівняння регресії першого порядку.

11.6.9 Рівняння регресії другого порядку.

11.6.10 Якими методами можна обчислити коефіцієнти рівняння регресії.

11.6.11 Як визначити адекватність рівняння регресії.

11.6.12 У чому заключається принцип вибіра найкращої математичної моделі.

12 ЛАБОРАТОРНА РОБОТА №11

ДОСЛІДЖЕННЯ МЕТОДІВ ЛІНІЙНОГО ПРОГРАМУВАННЯ

Мета роботи: метою роботи є вивчення методів побудова математичних моделей з використанням лінійного програмування.

12.1 ЗАВДАННЯ

12.1.1 Вивчення методів лінійного програмування.

12.1.2 Розробити алгоритм вирішення транспортної задачі.

12.1.3 Розробити програму по п. 2.2.

12.1.4 Перевірити роботу програми на конкретних даних наданих викладачем.

12.2 ОСНОВНІ ТЕОРЕТИЧНІ ПОЛОЖЕННЯ.

12.2.1 Ставлення задачи лінійного програмування.

Прикладом задачи лінійного програмування може служити транспортна задача. У пунктах A1,…,An знаходяться склади, де храняться товари у кількості X1,…,Xn. У пунктах B1,…,Bm знаходяться споживачі, яким треба поставити ці товари у кількості Y1,…,Ym відповідно. Будемо роздивляти випадок, коли на складі Ai знаходиться товар Xi. Дij - вартість перевезення одиниці вантажа між пунктами Ai та Bj. Дослідим операцію перевезення товару споживачам у кількостях для того, щоб задовольнити потреби споживачів. Позначимо через xij кількість товару, перевозимого з Ai у Bj. Щоб задовольнити необхідні запити треба, щоб xij задовольняло нерівності

$$\sum_i X_{ij} \geq Y_i \qquad (1)$$

Але зі складу номер і ми не можемо вивезти продукта більш, ніж там є. Це означає, що шукані величини повинні задовольняти ще одній системі нерівностей

$$\sum_j X_{ij} \le X_i \qquad (2)$$

Задовольнити умовам (1) та (2) та виконати план перевезеннь, забезпечуючий запити, можна численою кількістю способів. Для того, щоб дослідник операцій зміг вибрати визначене рішення, повинно бути сформульоване деяке правило відбору, визначаєме за допомогою критерія, який відображає наше суб'єктивне представлення цілі. При цьому ми отримаємо одну з можливих оцінок обраного рішення. Проблема критерію вирішується незалежно від досліджуваної операції, тобто критерій повинен бути задан оперуючою стороною. Уданій задачі одним з критеріїв буде вартість перевезень, яка визначається:

$$Y(x) = \sum_i \sum_j d_{ij} * x_{ij} \qquad (3)$$

Задачу о перевезеннях можна слідуючим чином: визначити хij, задовольняюче обмеженням (1) та (2) та забезпечуюче мінімальне значення функції (3). Обмеження (2) – умова балансу або закон зберігання, тобто умова фізичного типу. Умова (1) називається метою операції. Ці дві умови складають модель операції, Реалізація операції буде залежати від критеріїв. Тобто необхідними посиланнями задачи лінійного програмування є система рівняннь, які називаються обмеженнями, та цільова функція, відображаюча крітерій дослідження.

12.2.2 Опис алгоритму вирішення задач лінійного програмування.

Оскільки у задачах лінійного програмування число змінних більш числа рівнянь (n>m), то маємо нескінчену множину рішень. Іншими словами, маємо множину наборів змінних, x1,x2,x3,…, xn, які в принципі задовольняють усім заданим умовам та кожний такий набір можна вважати рішенням. Так як усі хі є які то фізичні

величини, вони не можуть бути <0, отже існують додаткові обмеження: xi>=0…xn>=0.

Частіше за усе у задачах лінійного програмування усі або декілька обмежень мають вид нерівностей, але такі нерівності, легко перетворити у рівняння, вводячи додаткову змінну: Xn+k. Тобто у загальному вигляді система рівнянь буде мати такий вигляд:

$$a_{i1} * x_1 + a_{i2} * x_2 + \ldots + a_{in} * x_n + X_n + k = b_i$$

$$L = c_{i1} * x_{i1} + c_{i2} * x_{i2} + \ldots + c_{in} * x_{in}$$

Наприклад:

Нехай маємо 4 склади A1,A2,A3,A4.

На цих складах маємо продукцію a1,a2,a3. Маємо 3 споживача B1, B2, B3 яким відповідно треба завезти продукцію у кількості в1, в2, в3.

Якщо на цих складах була б кількість продукції, для задоволення запитів, то виконалася б рівняння:

$$a_1 + a_2 + a_3 + a_4 = b_1 + b_2 + b_3$$

Але частіше всього нерівність

$$a_1 + a_2 + a_3 + a_4 \geq b_1 + b_2 + b_3$$

Позначимо через Cij кошт перевезення одиниці вантажу з одного пункта у інший.

Обмеження та цільова функція виглядають так:

$$x_{11} + x_{21} + x_{31} + x_{41} = b_1$$

$$x_{12} + x_{22} + x_{32} + x_{42} = b_2$$

$$x_{13} + x_{23} + x_{33} + x_{43} = b_3$$

x – кількість вантажу, яке перевозиться з кожного складу, кожному споживачів.

$$x_{11} + x_{12} + x_{13} = a_1$$

$$x_{21} + x_{22} + x_{23} = a_2$$

$$x_{31} + x_{32} + x_{33} = a_3$$

$$x_{41} + x_{42} + x_{43} = a_4$$

Головною ціллю задач лінійного програмування є знаходження оптимальних значень x, себто з множини значень треба вибрати одне, те саме, що обертає у мінімум лінійну форму цільової

функції. Оскільки число змінних більше, ніж число рівнянь, для отримання однозначного рішення необхідно зменшити число змінних.

Для цього уявимо x =0. Залишившуся систему можна вирішити звичайними методами. Набір використаних для рішення m змінних зветься базисом. Інші n-m змінні звуться незалежними. У кожній реальній системі може існувати декілька базисів, кожний зі своїм набором базисних та незалежних змінних. Якщо серед базисних рішень будуть такі які надають від'ємні значення змінним, то вони є неприпустимими і виключаються. Серед припустимих базисних рішень шукаються такі, які мінімізують (максимізують) лінійну цільову функцію. Ці рішення і є оптимальними.

12.3 ОПИС МЕТОДИКИ РІШЕННЯ ТРАНСПОРТНОЇ ЗАДАЧИ

Рішення транспортної задачи розглядається на конкретному прикладі транспортної задачи на залізниці, у якій відшукується оптимальне розподілення товарного порожняка.

Задача включає в себе визначення такого плану перегону порожніх товарних вагонів з пунктів оправлення (з надлишком вагонів) у пункти слідування (з вадою вагонів), щоб сумарні витрати на перегін вагонів були мінімальними при умові, що виконуються усі накладені обмеження.

У таблиці 3.1. та 3.2. наведені умови задачи та питомі (на один товарний вагон) витрати на перегін.

У таблицы 3.1. вказано, що в пунктах S1, S2, S3, маємо залишки порожніх товарних вагонів у кількостях 9, 4 та 8 вагонів, відповідно, у пунктах D1, D2, D3, D4, D5 не вистачає, відповідно, 3, 5, 4, 6 та 3 вагонів. (для спрощення уявимо, що задача сбалансована, тобто сумарна кількість надлишкових вагонів дорівнюється сумарній кількості невистаючих вагонів).

Таблиця 3.1
Умови задачи

Пункти призначення / Пункти відправлення	D1	D2	D3	D4	D5	Над - лишок
S1	X11	X12	X13	X14	X15	9
S2	X21	X22	X23	X24	X25	4
S3	X31	X31	X33	X34	X35	8
Потреба	3	5	4	6	3	21

Таблиця 3.2
Питомі витрати

Пункти призначення / Пункти відправлення	D1	D2	D3	D4	D5
S1	C11 -10	C12 -20	C13 -5	C14 -9	C15 -10
S2	C21 -2	C22 -10	C23 -8	C24 -30	C25 -6
S3	C31 -1	C31 -20	C33 -7	C34 -10	C35 -4

У таблиці 3.2. наведени питомі витрати Cij на перегін одного порожнього вагона з пункту відправлення у пункт призначення J (i-строка, j-стовбець). Таким чином, треба визначити такі значення Xij (i=1, 2, 3; j=1, 2, 3, 4, 5) таблиці 3.1., щоб задовольнити наданим обмеженням та мінімізувати сумарні витрати.

12.4 МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ

Для виконання пп.2.2, 2.3. необхідно ознайомитися із конспектом лекцій, вивчити методи лінійного програмування.

Для виконання пп.2.3., 2.4. необхідно отримати у викладача вихідні дані.

Отримати результати рішення на конкретних даних, отримати роздрук вихідних даних.

12.5 ЗМІСТ ЗВІТУ.

12.5.1 Мета роботи.
12.5.2 Опис метода рішення транспортної задачи.
12.5.3 Схема алгоритму.
12.5.4 Текст програми.

12.6 КОНТРОЛЬНІ ПИТАННЯ.

12.6.1 Сформулювати мету задачи лінійного програмування.
12.6.2 Методи рішення задач лінійного програмування на прикладі транспортної задачі.
12.6.3 Поняття цільової функції.
12.6.4 Принцип визначення обмежень у задачи лінійного програмування.
12.6.5 Поняття базисного рішення.

Додаток А

Експериментальні дані

Таблиця А.1 – Експериментальні дані

№	Y1	X1	Y2	X2	Y3	X3	Y4	X4	Y5	X5
1	24.2	50.3	96.7	121.5	63.1	0.009	0.0	0.0	4.5	2.6
2	22.3	42.7	102.3	130.3	56.7	0.007	0.011	0.012	0.0	0.0
3	25.6	50.7	101.1	129.7	65.1	0.016	0.0	0.005	5.4	4.1
4	24.8	47.9	100.3	127.1	63.9	0.016	0.010	0.0	4.8	3.1
5	22.9	44.3	409.5	409.5	58.7	0.016	0.052	0.007	40.9	24.6
6	22.9	43.1	89.5	108.7	55.5	0.017	0.078	0.007	18.8	7.7
7	22.1	41.1	91.5	121.4	51.1	0.008	0.013	0.013	40.9	5.6
8	22.9	42.7	94.7	233.1	55.1	0.007	0.0	0.0	40.9	6.4
9	24.7	46.3	89.5	122.3	59.1	0.0	0.006	0.013	23.6	5.0
10	23.2	44.3	90.3	110.3	56.7	0.007	0.013	0.010	9.3	5.7
11	25.8	51.1	100.7	149.1	68.3	0.0	0.0	0.010	17.5	6.4
12	26.0	51.1	109.1	150.3	67.5	0.009	0.0	0.0	4.9	3.2
13	24.0	46.5	108.7	144.3	60.7	0.010	0.0	0.010	4.9	3.3
14	23.9	47.9	106.7	147.1	60.7	0.007	0.0	0.011	6.0	5.9
15	23.8	44.7	72.7	53.5	59.7	4.1	0.010	4.1	5.7	4.1
16	21.9	40.7	98.7	125.5	58.3	0.0	0.0	0.012	5.1	3.3
17	21.7	40.7	101.5	130.3	58.1	0.0	0.011	4.1	4.6	3.1
18	22.6	42.7	102.3	134.3	60.7	0.016	0.013	0.0	5.1	3.4
19	23.0	43.9	101.5	131.5	62.3	0.007	0.0	0.007	5.0	3.3
20	22.3	41.5	102.3	133.5	58.1	0.011	0.0	0.012	4.6	2.9
21	18.3	29.5	99.9	129.9	46.1	0.007	0.011	0.0	5.7	4.3
22	19.7	31.5	102.3	135.5	49.1	0.0	0.006	0.007	4.8	3.0
23	19.0	30.5	102.3	142.3	47.1	0.017	0.0	0.0	11.4	5.7
24	17.6	28.3	102.3	147.1	43.3	0.0	0.009	0.007	17.2	6.4
25	18.6	29.5	106.3	167.5	45.9	0.0	0.010	0.013	34.2	5.2
26	24.5	47.7	97.3	122.7	65.7	0.017	0.010	0.0	4.5	2.8
27	25.1	49.5	99.8	126.3	66.3	0.0	0.011	0.0	4.5	2.8
28	24.3	47.5	100.7	127.9	64.7	0.0	0.007	0.013	4.5	2.8
29	24.8	47.9	102.3	131.9	65.7	0.009	0.013	0.0	4.6	3.0
30	26.1	51.1	102.3	133.5	70.3	0.010	0.0	0.013	4.7	3.0
31	23.3	43.9	102.3	133.5	59.1	0.017	0.0	0.011	5.1	3.4
32	24.7	47.1	101.5	132.7	63.1	0.0	0.0	0.012	11.1	7.6
33	25.3	48.7	106.1	144.3	63.9	0.016	0.007	0.012	16.8	6.2
34	22.3	42.1	109.3	143.5	55.9	0.014	0.007	0.012	5.9	4.2
35	22.4	41.9	107.9	140.7	55.9	0.007	0.007	0.0	7.4	5.1

Продовження таблиці А.1

№	Y6	X6	Y7	X7	Y8	X8	Y9	X9	Y10	X10
1	0.201	0.037	0.098	17.5	21.7	0.41	0.736	0.956	6.3	0.296
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.215	0.042	0.123	18.6	22.1	0.484	0.618	0.802	6.3	0.484
4	0.217	0.039	0.107	18.3	21.6	0.486	0.619	0.804	6.2	0.484
5	0.217	0.034	0.089	17.5	20.4	0.488	0.62	0.804	5.8	0.488
6	0.174	0.033	0.093	17.3	20.1	0.123	0.612	0.92	1.6	0.12
7	0.179	0.031	0.097	17.6	20.1	0.488	0.62	0.797	6.1	0.486
8	0.183	0.034	0.111	18.5	21.1	0.484	0.616	0.796	6.3	0.484
9	0.173	0.037	0.093	17.9	21.3	0.485	0.618	0.794	8.0	0.485
10	0.180	0.034	0.088	16.5	20.4	0.488	0.62	0.792	7.4	0.485
11	0.223	0.040	0.115	18.0	21.9	0.495	0.628	0.8	6.2	0.342
12	0.243	0.048	0.409	18.5	22.6	0.494	0.628	0.802	6.2	0.494
13	0.246	0.041	0.131	17.8	21.3	0.493	0.625	0.802	6.4	0.49
14	0.251	0.038	0.111	17.5	20.9	0.488	0.628	0.802	4.6	0.49
15	4.1	0.011	0.013	40.9	40.9	0.432	0.744	1.068	6.3	0.0
16	0.209	0.035	0.090	17.0	19.7	0.288	0.624	0.91	7.0	0.288
17	0.219	0.037	0.1	17.2	19.9	0.489	0.621	0.795	6.6	0.488
18	0.223	0.038	0.111	17.7	20.5	0.484	0.616	0.792	6.9	0.485
19	0.223	0.037	0.095	17.4	20.3	0.484	0.617	0.793	7.8	0.482
20	0.228	0.034	0.087	17.4	20.0	0.488	0.621	0.79	7.1	0.488
21	0.219	0.027	0.058	16.2	17.7	0.444	0.736	1.015	6.1	0.444
22	0.231	0.029	0.063	17.0	18.8	0.488	0.623	0.802	5.6	0.49
23	4.1	0.028	0.06	16.5	18.1	0.488	0.622	0.801	6.0	0.488
24	0.235	0.025	0.058	14.4	17.7	0.488	0.621	0.8	4.3	0.482
25	0.235	0.028	0.059	17.3	18.7	0.49	0.623	0.8	6.1	0.491
26	0.203	0.041	0.112	18.1	21.5	0.362	0.725	0.964	6.4	0.292
27	0.211	0.039	0.105	18.2	21.8	0.488	0.622	0.792	6.1	0.489
28	0.214	0.042	0.409	18.3	40.9	0.484	0.619	0.793	6.5	0.488
29	0.221	0.044	0.134	18.3	21.7	0.487	0.619	0.793	6.6	0.487
30	0.223	0.042	0.117	18.3	22.1	0.49	0.622	0.79	6.4	0.489
31	0.222	0.037	0.101	18.1	21.3	0.439	0.697	1.017	6.0	0.438
32	0.215	0.041	0.409	19.4	23.0	0.488	0.621	0.8	5.8	0.487
33	0.237	0.037	0.097	18.7	22.5	0.486	0.621	0.8	6.1	0.488
34	0.244	0.034	0.089	17.9	20.5	0.488	0.621	0.8	6.1	0.488
35	0.237	0.037	0.104	17.9	20.5	0.49	0.623	0.795	6.2	0.49

Продовження таблиці А.1

№	Y11	X11	Y12	X12	Y13	X13	Y14	X14	Y15	X15
1	0.628	0.788	0.549	0.674	0.753	8.6	0.772	0.004	0.0	2.4
2	0.0	0.0	0.0	0.0	0.0	0.0	2.4	0.034	6.2	19.4
3	0.616	0.749	0.517	0.661	0.747	8.5	1.0	0.056	7.7	40.9
4	0.616	0.75	0.548	0.674	0.755	8.6	0.976	0.059	7.5	40.9
5	0.616	0.749	0.549	0.674	0.755	8.6	0.951	0.054	11.5	40.9
6	0.57	0.778	0.543	0.669	0.747	9.0	0.87	0.004	0.0	2.2
7	0.616	0.745	0.543	0.669	0.751	8.9	1.2	0.058	8.8	26.6
8	0.616	0.747	0.495	0.642	0.735	8.7	1.3	0.055	5.2	26.2
9	0.613	0.744	0.491	0.642	0.735	8.7	1.2	0.057	9.5	21.7
10	0.616	0.744	0.491	0.641	0.735	5.7	1.1	0.056	2.9	23.1
11	0.623	0.75	0.549	0.676	0.759	8.6	1.2	0.057	8.7	11.6
12	0.622	0.752	0.507	0.657	0.747	8.4	1.2	0.061	7.5	28.4
13	0.62	0.752	0.503	0.656	0.747	8.5	1.2	0.061	20.5	27.5
14	0.622	0.751	0.503	0.651	0.743	8.5	1.3	0.056	11.7	28.1
15	0.184	0.848	0.547	0.675	0.755	8.6	0.915	0.004	0.0	2.3
16	0.569	0.776	0.543	0.669	0.75	9.0	0.853	0.004	0.0	2.8
17	0.617	0.746	0.493	0.645	0.735	8.7	1.1	0.061	8.0	40.9
18	0.161	0.744	0.541	0.667	0.749	9.0	1.2	0.06	0.95	40.9
19	0.612	0.745	0.487	0.643	0.735	8.8	1.1	0.063	6.4	40.9
20	0.616	0.744	0.525	0.661	0.743	9.0	1.2	0.061	20.6	40.9
21	0.642	0.808	0.511	0.653	0.735	8.2	0.758	0.003	0.0	2.2
22	0.618	0.749	0.551	0.677	0.759	8.5	0.927	0.056	5.6	29.7
23	0.617	0.749	0.527	0.667	0.754	8.3	0.967	0.059	5.9	30.1
24	0.617	0.749	0.51	0.658	0.747	8.2	0.975	0.065	8.1	28.6
25	0.62	0.749	0.511	0.661	0.751	8.2	0.923	0.056	7.3	28.9
26	0.617	0.789	0.546	0.671	0.751	8.8	0.679	0.004	0.0	2.5
27	0.618	0.745	0.487	0.646	0.741	8.6	0.389	0.052	10.6	40.9
28	0.616	0.744	0.481	0.638	0.734	8.5	0.973	0.059	6.2	40.9
29	0.616	0.746	0.503	0.654	0.743	8.6	1.0	0.061	7.3	40.9
30	0.618	0.745	0.491	0.646	0.741	8.6	1.0	0.051	13.3	40.9
31	0.628	0.797	0.529	0.668	0.751	8.3	0.647	0.004	0.0	2.5
32	0.616	0.748	0.551	0.677	0.759	8.4	0.821	0.058	9.0	29.5
33	0.616	0.748	0.497	0.653	0.745	8.0	0.807	0.063	5.0	29.6
34	0.618	0.747	0.503	0.652	0.743	8.2	0.815	0.067	4.2	29.7
35	0.618	0.746	0.503	0.654	0.746	8.1	0.823	0.057	5.1	29.3

Продовження таблиці А.1

№	Y16	X16	Y17	X17	Y18	X18	Y19	X19	Y20	X20
1	19.7	81.9	110.5	1.8	4.7	4.0	6.4	27.0	81.9	113.4
2	0.0	71.5	105.7	2.0	0.0	0.0	0.0	0.0	81.9	135,4
3	17.3	81.9	107.3	2.1	4.8	3.9	6.4	26.3	81.9	108.3
4	17.1	81.9	97.1	2.2	5.2	4.0	6.4	25.9	81.9	98.0
5	16.5	81.9	123.7	2.3	4.3	4.0	6.4	26.6	81.9	125.9
6	18.0	81.9	409.5	4.1	0.0	0.0	7.3	7.3	81.9	409.5
7	16.9	81.9	409.5	4.1	0.8	0.8	6.7	8.1	81.9	345.1
8	16.8	81.9	409.5	4.1	0.4	0.3	6.8	7.3	81.9	409.5
9	15.9	81.9	409.5	4.1	0.0	0.0	8.2	8.2	81.9	409.5
10	15.9	81.9	409.5	4.1	0.0	0.0	4.3	4.3	81.9	409.5
11	15.4	81.9	102.3	2.1	5.1	4.0	6.4	26.8	81.9	107.0
12	16.2	81.9	108.7	2.3	4.9	4.0	6.4	26.6	81.9	111.7
13	16.3	81.9	195.3	3.3	2.5	3.8	6.5	17.9	81.9	176.7
14	14.9	81.9	145.7	2.3	3.6	4.2	6.4	28.9	81.9	137.3
15	16.8	81.9	123.9	1.9	4.2	4.0	6.4	27.4	81.9	125.8
16	17.2	81.9	409.5	4.1	0.0	0.0	7.2	7.2	81.9	409.5
17	15.5	81.9	336.3	4.1	1.2	1.5	6.6	9.5	81.9	265.5
18	15.8	81.9	409.5	4.1	0.0	0.0	7.1	7.1	81.9	409.5
19	15.8	81.9	409.5	4.1	0.0	0.0	8.2	8.2	81.9	409.5
20	15.7	81.9	409.5	4.1	0.0	0.0	7.2	7.2	81.9	409.5
21	18.4	81.9	83.7	1.3	6.0	4.2	6.3	24.5	81.9	88.3
22	17.2	79.0	78.5	1.5	6.3	4.2	6.2	25.3	71.0	77.3
23	17.0	78.7	78.6	1.6	6.2	4.2	6.3	24.2	81.9	80.7
24	17.8	81.9	107.9	1.9	4.7	4.1	6.3	24.6	81.9	113.1
25	15.5	81.9	104.6	1.0	1.0	2.8	6.3	16.2	81.9	110.5
26	15.4	81.9	182.3	2.4	2.8	4.0	6.6	19.0	81.9	185.6
27	15.8	81.9	134.3	2.5	3.9	4.0	6.5	27.7	81.9	135.1
28	13.2	81.9	409.5	4.1	0.5	0.4	6.7	7.3	81.9	329.8
29	15.9	81.9	409.5	4.1	0.04	0.01	6.8	6.8	81.9	409.5
30	15.1	81.9	132.5	2.7	5.6	3.7	6.5	27.5	76.7	79.1
31	18.4	61.4	68.2	1.3	7.2	4.0	6.2	23.7	71.4	71.0
32	17.3	80.6	85.9	1.8	6.1	4.0	6.2	21.5	81.9	86.3
33	16.9	59.6	65.6	1.7	7.2	4.0	6.2	23.1	63.4	66.9
34	16.5	54.1	60.7	1.7	7.1	4.0	6.2	22.1	57.6	60.9
35	15.0	68.5	70.3	1.7	7.2	4.0	6.2	25.3	72.7	73.5

Продовження таблиці А.1

№	Y21	X21	Y22	X22	Y23	X23	Y24	X24	Y25	X25
1	1.2	0.663	0.725	0.942	0.658	0.722	0.985	27.3	6.5	3.5
2	1.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	1.3	0.666	0.73	0.938	0.66	0.735	1.3	27.1	6.5	3.6
4	1.6	0.664	0.73	0.934	0.66	0.724	1.0	26.2	6.5	3.5
5	1.6	0.665	0.728	0.929	0.66	0.728	1.1	28.9	6.5	3.6
6	4.1	0.645	0.71	0.904	0.648	0.714	1.1	7.5	7.5	0.0
7	3.3	0.649	0.717	0.921	0.649	0.722	1.2	8.3	7.0	0.74
8	3.9	0.648	0.715	0.922	0.648	0.72	1.2	7.1	7.0	0.0
9	4.1	0.653	0.718	0.909	0.65	0.724	1.1	8.4	8.4	0.0
10	4.1	0.653	0.718	0.912	0.65	0.72	1.0	8.0	8.0	0.0
11	1.1	0.668	0.736	0.952	0.661	0.74	1.3	27.4	6.4	3.6
12	1.1	0.672	0.738	0.953	0.668	0.748	1.4	26.7	6.4	3.6
13	1.7	0.664	0.731	0.941	0.662	0.736	1.3	17.6	6.6	3.4
14	1.2	0.672	0.737	0.96	0.668	0.748	1.4	22.4	6.5	3.4
15	1.0	0.668	0.736	0.944	0.666	0.742	1.4	23.7	6.5	3.5
16	4.1	0.654	0.716	0.9	0.649	0.715	0.934	7.3	7.3	0.0
17	2.4	0.652	0.72	0.936	0.652	0.726	1.2	9.1	6.7	1.3
18	4.1	0.65	0.717	0.922	0.651	0.72	1.1	7.3	7.2	0.0
19	4.1	0.634	0.705	0.912	0.632	0.705	0.978	8.2	8.2	0.0
20	4.1	0.646	0.712	0.917	0.644	0.712	0.993	7.3	7.3	0.0
21	0.983	0.672	0.736	0.932	0.666	0.732	1.1	25.1	6.4	3.6
22	1.1	0.672	0.736	0.94	0.666	0.736	1.3	25.2	6.4	3.7
23	1.1	0.672	0.736	0.946	0.666	0.738	1.3	24.2	6.4	3.7
24	1.4	0.669	0.734	0.943	0.666	0.736	1.2	25.2	6.4	3.7
25	1.3	0.668	0.732	0.936	0.666	0.733	1.2	28.2	6.4	3.7
26	1.7	0.656	0.721	0.912	0.656	0.656	0.943	19.6	6.8	3.7
27	1.7	0.656	0.721	0.919	0.65	0.65	0.974	28.7	6.7	3.6
28	4.1	0.65	0.718	0.92	0.65	0.65	0.982	7.4	6.9	0.29
29	4.1	0.656	0.72	0.922	0.656	0.656	0.98	6.9	6.9	0.0
30	0.323	0.626	0.704	0.96	0.594	0.594	0.966	29.1	6.5	4.3
31	0.915	0.668	0.73	0.928	0.66	0.724	0.949	24.1	6.3	3.7
32	1.2	0.666	0.73	0.936	0.66	0.724	1.0	25.4	6.4	3.7
33	1.1	0.669	0.732	0.94	0.66	0.726	1.0	23.7	6.3	3.7
34	1.1	0.67	0.734	0.941	0.66	0.728	1.0	22.5	6.3	3.7
35	1.1	0.667	0.73	0.933	0.658	0.724	0.982	25.8	6.4	3.7

Продовження таблиці А.1

№	Y26	X26	Y27	X27	Y28	X28	Y29	X29	Y30	X30
1	6.5	82.8	94.6	1.2	0.648	0.709	0.882	0.64	0.703	0.955
2	0.0	81.9	123.9	1.6	0.0	0.0	0.0	0.0	0.0	0.0
3	6.7	66.6	90.1	1.4	0.0	0.0	0.0	0.0	0.0	0.0
4	6.8	71.6	84.1	1.5	0.648	0.712	0.893	0.637	0.704	0.968
5	6.5	68.8	81.6	1.3	0.644	0.708	0.891	0.637	0.706	1.1
6	0.0	81.9	409.5	4.1	0.176	0.384	0.869	0.174	0.389	1.1
7	1.0	81.9	377.1	3.7	0.628	0.694	0.906	0.626	0.698	1.2
8	0.0	81.9	409.5	4.1	0.626	0.693	0.877	0.624	0.697	1.1
9	0.0	81.9	409.5	4.1	0.624	0.69	0.866	0.624	0.696	1.1
10	0.0	81.9	409.5	4.1	0.626	0.692	0.869	0.624	0.693	0.957
11	6.8	81.9	91.7	1.3	0.653	0.716	0.904	0.644	0.72	1.3
12	6.6	81.9	98.3	1.5	0.652	0.717	0.904	0.648	0.726	1.3
13	4.9	81.9	175.7	2.2	0.648	0.712	0.896	0.642	0.716	1.2
14	5.8	81.9	124.3	1.6	0.648	0.714	0.9	0.644	0.724	1.3
15	5.9	81.9	113.8	1.2	0.652	0.714	0.895	0.647	0.72	1.3
16	0.0	81.9	409.5	4.1	0.628	0.693	0.869	0.623	0.693	0.901
17	1.8	81.9	297.1	2.9	0.634	0.7	0.888	0.632	0.705	1.2
18	0.0	81.9	409.5	4.1	0.629	0.694	0.88	0.629	0.697	1.0
19	0.0	81.9	409.5	4.1	0.626	0.69	0.871	0.622	0.692	0.937
20	0.0	81.9	409.5	4.1	0.622	0.689	0.874	0.62	0.688	0.954
21	7.2	69.1	73.5	0.957	0.656	0.72	0.896	0.648	0.713	1.1
22	7.6	64.7	68.2	1.2	0.656	0.72	0.908	0.648	0.716	1.2
23	7.2	31.4	36.7	0.711	0.66	0.724	0.912	0.648	0.717	1.2
24	6.2	81.9	87.5	0.879	0.632	0.701	0.912	0.61	0.693	1.2
25	6.9	81.9	94.7	1.4	0.653	0.717	0.902	0.645	0.716	1.2
26	4.9	81.9	166.2	1.6	0.64	0.704	0.879	0.632	0.698	0.912
27	6.3	81.9	117.9	1.6	0.64	0.706	0.888	0.632	0.699	0.943
28	0.44	81.9	409.5	4.1	0.637	0.701	0.885	0.636	0.7	0.946
29	0.0	81.9	409.5	4.1	0.634	0.698	0.878	0.631	0.696	0.936
30	8.4	81.9	128.5	1.7	0.635	0.701	0.884	0.628	0.694	0.926
31	7.8	53.8	64.0	0.871	0.656	0.718	0.896	0.645	0.708	0.922
32	7.6	69.1	69.8	1.1	0.65	0.716	0.92	0.642	0.708	0.98
33	7.8	57.4	63.5	1.1	0.656	0.719	0.909	0.648	0.712	0.991
34	7.5	54.7	57.8	1.1	0.656	0.72	0.904	0.648	0.709	0.98
35	8.4	81.9	12.7	0.101	0.575	0.683	0.972	0.553	0.63	0.93

Продовження таблиці А.1

№	Y31	X31	Y32	X32	Y33	X33	Y34	X34	Y35	X35
1	26.7	6.4	4.2	7.2	81.9	118.9	14.8	2.8	5.8	24.6
2	0.0	0.0	0.0	0.0	81.9	160.7	13.3	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	81.9	119.0	12.0	0.0	0.0	0.0
4	25.3	6.4	4.1	7.1	81.9	106.0	11.9	2.7	5.5	23.0
5	26.8	6.4	4.2	8.0	81.9	128.4	12.9	2.8	6.2	25.9
6	0.58	0.58	0.0	0.0	81.9	409.5	16.9	0.0	0.0	7.3
7	8.1	6.8	0.76	1.1	81.9	409.5	16.9	0.08	0.08	6.9
8	7.2	6.9	0.2	0.32	81.9	409.5	16.9	0.04	0.08	3.2
9	8.2	8.2	0.0	0.0	81.9	409.5	16.9	0.0	0.0	8.0
10	7.9	7.9	0.0	0.0	81.9	409.5	16.9	0.0	0.0	7.8
11	27.5	6.3	4.2	7.5	81.9	118.1	12.6	2.7	6.0	25.1
12	27.0	6.3	4.1	7.1	81.9	409.5	13.2	0.0	0.0	6.3
13	17.6	6.5	4.0	5.6	81.9	315.2	16.7	2.7	4.3	12.2
14	28.8	6.4	4.1	7.2	81.9	175.5	14.7	2.7	6.1	18.5
15	28.1	6.4	4.1	7.0	81.9	152.7	16.6	2.8	5.7	25.9
16	7.2	7.2	0.0	0.0	81.9	409.5	16.8	0.0	0.0	7.2
17	9.1	6.7	1.4	1.9	81.9	409.5	16.7	0.12	0.12	6.2
18	7.2	7.2	0.0	0.0	81.9	409.5	16.8	0.0	0.0	7.2
19	8.2	8.3	0.0	0.0	81.9	409.5	16.8	0.0	0.0	8.2
20	7.3	7.3	0.0	0.0	81.9	409.5	16.8	0.0	0.0	7.2
21	24.8	6.2	4.2	7.6	81.6	88.0	13.5	2.8	6.0	22.3
22	18.6	6.2	4.2	6.2	72.6	77.9	10.5	2.7	6.3	22.2
23	23.4	6.1	4.4	8.7	74.2	82.3	10.7	2.7	5.8	21.3
24	17.6	6.2	4.5	3.8	81.9	194.3	12.2	0.0	0.0	6.4
25	26.5	6.3	4.2	7.6	81.9	113.5	12.0	2.8	6.2	24.8
26	19.5	6.6	4.2	5.5	81.9	255.9	16.7	2.9	4.8	17.9
27	26.2	6.4	4.3	7.0	81.9	409.5	15.9	0.0	0.0	5.9
28	7.9	6.7	0.29	0.42	81.9	409.5	16.8	0.08	0.04	6.8
29	6.8	6.8	0.0	0.0	81.9	409.5	16.8	0.0	0.0	6.7
30	18.4	6.5	2.0	2.2	81.9	176.3	14.5	2.8	2.1	13.1
31	23.7	6.2	4.3	7.9	81.9	96.4	12.5	2.8	6.0	22.0
32	24.6	6.2	4.2	4.2	81.9	409.5	11.5	0.0	0.0	6.9
33	23.1	6.2	4.2	7.7	81.9	99.0	10.3	2.7	5.7	21.5
34	22.2	6.2	4.2	7.4	74.2	82.3	9.7	2.8	5.5	20.1
35	14.7	5.7	0.96	0.76	81.9	409.5	9.7	0.0	0.0	7.2

13 ЛІТЕРАТУРА

Основна:

9.1 Костин А.Е., Шаньгин В.Ф. «Организация и обработка структур даних в вычислительных системах».-М.:Высшая школа,1987,-242 с.
9.2 Вирт Н. «Алгоритмы+структуры данных=программы». -М. Мир, 1985.
9.3 Бертисс А.Т. «Структуры данных»./Пер. с англ. - М.: Статистика, 1974.
9.4 Трамбле Ж., Соренсон П. «Введение в структуры данных».-М.: Машиностроение, 1982.

Додаткова:

9.5 Дрибас В.П. «Реляционные модели баз данных».-Минск: БГУ, 1982.
9.6 Зайцев В.Ф. «Кодирование информации в ЕС ЭВМ». -М.: Радио и связь, 1986.
9.7 Макаровский Б.Н. «Информационные системы и структуры данных».-М.:Статистика, 1980.
9.8 «Данные в языках программирования: Абстракции и топология»./Пер. с англ.-М.:Мир,1982.
9.9 Вегнер П. «Программирование на языке АДА». /Пер. с англ.-М.,Мир,1983.
9.10 Ленгсам И.О. Генстайм М.О. «Структуры данных для персональных ЭВМ».,1992.
9.11 Кнут Д. «Искусство программирования для ЭВМ».-М.:Мир, 1976-1978.