### МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ

**Національний технічний університет України “Київський політехнічний інститут”**

СТРУКТУРИ ДАНИХ ТА АЛГОРИТМИ – 1.

ОСНОВИ АЛГОРИТМІЗАЦІЇ

Завдання до виконання лабораторних робіт по кредитному модулю

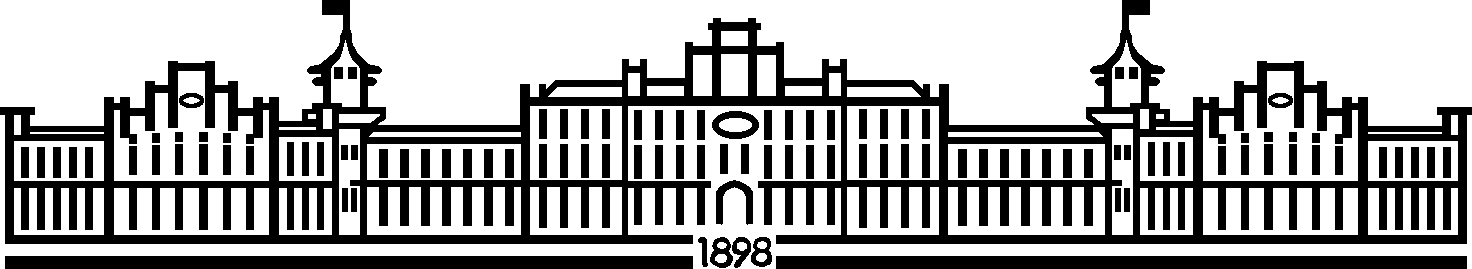
"Структури даних та алгоритми – 1.

Основи алгоритмізації”

для студентів напрямку 6.050102

«Комп’ютерна інженерія»

*Рекомендовано вченою радою факультету прикладної математики НТУУ «КПІ»*



Київ 2013

### МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ

**Національний технічний університет України “Київський політехнічний інститут”**

СТРУКТУРИ ДАНИХ ТА АЛГОРИТМИ – 1.

ОСНОВИ АЛГОРИТМІЗАЦІЇ

Завдання до виконання лабораторних робіт по кредитному модулю

"Структури даних та алгоритми – 1.

Основи алгоритмізації”

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» 2013

УДК 004.42

Структури даних та алгоритми – 1. Основи алгоритмізації: зав- дання 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», 2013. – 57 с.

*Гриф надано вченою радою ФПМ (протокол № 6 від 28.01.2013 р.)*

Навчальне електронне видання містить завдання до виконання лабораторних робіт по кредитному модулю «Структури даних та алгоритми – 1. Основи алгоритмізації».

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

Н а в ч а л ь н е е л е к т р о н н е в и д а н н я

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*

© Марченко Олександр Іванович, 2013

За редакцією автора

## ЗМІСТ

[СТРУКТУРИ ДАНИХ ТА АЛГОРИТМИ – 1. ОСНОВИ](#_bookmark0) [АЛГОРИТМІЗАЦІЇ 1](#_bookmark0)

[ЗМІСТ 3](#_bookmark1)

[ВСТУП 5](#_bookmark2)

1. [ЛАБОРАТОРНА РОБОТА №1.1. РОЗГАЛУЖЕНІ](#_bookmark3) [АЛГОРИТМИ 7](#_bookmark3)
2. [ЛАБОРАТОРНА РОБОТА №1.2. АЛГОРИТМИ З](#_bookmark4) [ВКЛАДЕНИМИ ЦИКЛАМИ ТА МЕТОД ДИНАМІЧНОГО](#_bookmark4) [ПРОГРАМУВАННЯ 14](#_bookmark4)
3. [ЛАБОРАТОРНА РОБОТА №1.3. ВИКОРИСТАННЯ](#_bookmark5) [КЕРУЮЧИХ КОНСТРУКЦІЙ РОЗГАЛУЖЕННЯ ТА ЦИКЛІВ ПРИ РОБОТІ З ОДНОВИМІРНИМИ](#_bookmark5) [МАСИВАМИ 19](#_bookmark5)
4. [ЛАБОРАТОРНА РОБОТА №1.4. АЛГОРИТМИ](#_bookmark6) [ПЕРЕТВОРЕННЯ ОДНОВИМІРНИХ МАСИВІВ](#_bookmark6) [(ВЕКТОРІВ) 24](#_bookmark6)
5. [ЛАБОРАТОРНА РОБОТА №1.5. АЛГОРИТМИ](#_bookmark7) [ЛІНІЙНОГО ПОШУКУ 32](#_bookmark7)
6. [ЛАБОРАТОРНА РОБОТА №1.6. АЛГОРИТМИ](#_bookmark8) [ОБХОДУ ДВОВИМІРНИХ МАСИВІВ (МАТРИЦЬ) 40](#_bookmark8)

## ВСТУП

Дисципліна "Структури даних та алгоритми" є базовою спеціальною нормативною дисципліною підготовки фахівців рівня бакалавр з напрямку 050102 «Комп’ютерна інженерія», а також студентів спеціальностей 8.091.501 "Комп’ютерні системи та мере- жі", 8.091.502 "Системне програмування" та 8.091.503 "Спеціалі- зовані комп’ютерні системи" i викладається у 1-му та 2-му семе- страх.

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

Кредитний модуль «Структури даних та алгоритми – 1. Основи алгоритмізації» призначений для вивчення принципів та концепцій, які покладені в основу розробки обчислювальних і ке- руючих алгоритмів та відповідних їм структур даних, а також структурної та об’єктно-орієнтованої методологій програмування.

Цикл лабораторних робіт складається з шести робіт і при- значений для покриття практичної частини першого кредитного модуля дисципліни «Структури даних та алгоритми».

Роботи дозволяють отримати практичні навички складання розгалужених та циклічних алгоритмів, а також використання базових 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 захи- сту лабораторної роботи;
* варіанти індивідуальних завдань.

## ЛАБОРАТОРНА РОБОТА №1.1. РОЗГАЛУЖЕНІ АЛГОРИТМИ

**Мета лабораторної роботи**

Метою лабораторної роботи №1.1. «Розгалужені алгоритми» є засвоєння теоретичного матеріалу та набуття практичних навичок використання керуючих конструкцій розгалуження та булевих (ло- гічних) операцій.

## Постановка задачі

Задано дійсне число ***x***. Визначити значення заданої за варі- антом кусочно-безперервної функції ***y***(***x***), якщо воно існує, або вивести на екран повідомлення про неіснування функції для за- даного ***x***.

Розв’язати задачу двома способами (написати дві програми):

1. в програмі дозволяється використовувати тільки одиничні операції порівняння (=, <>, <, <=, >, >=) і **не** дозволяється викори- стовувати булеві (логічні) операції (***not***, ***and***, ***or***, тощо);
2. в програмі необхідно обов’язково використати булеві (логі- чні) операції (***not***, ***and***, ***or***, тощо); використання булевих операцій не повинно бути надлишковим.

## Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варі- анту.
2. Текст обох програм.
3. Тестування обох програм. При тестуванні кожної з програм необхідно підібрати такий набір вхідних значень ***x***, щоб корект- ність всіх гілок алгоритмів була протестована.
4. В якості результату роздрукувати дані тестування для всіх вхідних значень з тестового набору для кожної з програм окремо.

## Контрольні питання

1. Класифікація керуючих конструкцій розгалуження.
2. Запис керуючих конструкцій розгалуження діаграмами дій.
3. Запис керуючих конструкцій розгалуження мовою програму- вання.
4. Таблиця істинності булевої (логічної) операції ***not***.
5. Таблиця істинності булевої (логічної) операції ***and***.
6. Таблиця істинності булевої (логічної) операції ***or***.
7. Запис складних булевих (логічних) виразів.
8. Які існують варіанти використання вкладених умовних кон- струкцій (при наявності альтернативної гілки та при її відсутності) та особливості запису таких вкладених конструкцій мовою про- грамування?

**Варіант 1**

## Варіанти індивідуальних завдань

 10x3 + 7x/5 + 2 , x  [–49,–10) U (0,10]

y = 

 –x + 9 , x  (20, + )

### Варіант 2

 x3 – 5x2 , x  (0,5]

y = 

 x2 – 3 , x  [–32,–20) U (10, + )

### Варіант 3

 –14x – 20 , x  (–21,3] U (12, + )

y = 

 13x2/11 – 6 , x  (– , –41]

### Варіант 4

 4x2 + 2 , x  (–15,3]

y = 

 3x3/4 – 5 , x  (– , –30] U (20, + )

### Варіант 5

 x3 – 6 , x  (–10,–5] U (5,15]

y = 

 2x3 – 3x + 2 , x  [25, + )

### Варіант 6

 x3 + 14 , x  [2,7]

y = 

 –4x3 + 3x – 7 , x  (–13,–3] U (14, + )

### Варіант 7

 –3x2/5 + 9 , x  [0,8] U [16, + )

y = 

 15x – 2 , x  (– , –1)

### Варіант 8

 –6x2 + 8 , x  [0,7)

y = 

 –x3/7 + 10 , x  (– , –10] U [11, + )

### Варіант 9

 –9x3 + 5x2 , x  (2,12] U (22,32)

y = 

 – x2 – 12 , x  (– , 0]

### Варіант 10

 – 5x3 + 10 , x  [8,23)

y = 

 2x3 + 8x2 , x  (– , –19) U (–3,0]

### Варіант 11

 10x3 + 7x/5 + 2 , x  [–49,–10) U (0,10]

y = 

 –x + 9 , x  (20, + )

### Варіант 12

 x3 – 5x2 , x  (0,5] y = 

 x2 – 3 , x  [–32,–20) U (10, + )

|  |  |  |
| --- | --- | --- |
| **Варіант 13** |  | |
|  | –14x – 20 | , x  (–21,3] U (12, + ) |
| y =    | 13x2/11 – 6 | , x  (– , –41] |
| **Варіант 14** |  |  |
|  | 4x2 + 2 | , x  (–15,3] |
| y =    | 3x3/4 – 5 | , x  (– , –30] U (20, + ) |
| **Варіант 15** |  |  |
|  | x3 – 6 | , x  (–10,–5] U (5,15] |
| y =    | 2x3 – 3x + 2 | , x  [25, + ) |
| **Варіант 16** |  |  |
|  | x3 + 14 | , x  [2,7] |
| y =    | –4x3 + 3x – 7 | , x  (–13,–3] U (14, + ) |
| **Варіант 17** |  |  |
|  | –3x2/5 + 9 | , x  [0,8] U [16, + ) |
| y =    | 15x – 2 | , x  (– , –1) |
| **Варіант 18** |  |  |
|  | –6x2 + 8 | , x  [0,7) |
| y =    | –x3/7 + 10 | , x  (– , –10] U [11, + ) |

### Варіант 19

 –9x3 + 5x2 , x  (2,12] U (22,32)

y = 

 – x2 – 12 , x  (– , 0]

### Варіант 20

 – 5x3 + 10 , x  [8,23)

y = 

 2x3 + 8x2 , x  (– , –19) U (–3,0]

### Варіант 21

 10x3 + 7x/5 + 2 , x  [–49,–10) U (0,10]

y = 

 –x + 9 , x  (20, + )

### Варіант 22

 x3 – 5x2 , x  (0,5] y = 

 x2 – 3 , x  [–32,–20) U (10, + )

### Варіант 23

 –14x – 20 , x  (–21,3] U (12, + )

y = 

 13x2/11 – 6 , x  (– , –41]

### Варіант 24

 4x2 + 2 , x  (–15,3]

y = 

 3x3/4 – 5 , x  (– , –30] U (20, + )

### Варіант 25

 x3 – 6 , x  (–10,–5] U (5,15]

y = 

 2x3 – 3x + 2 , x  [25, + )

### Варіант 26

 x3 + 14 , x  [2,7]

y = 

 –4x3 + 3x – 7 , x  (–13,–3] U (14, + )

### Варіант 27

 –3x2/5 + 9 , x  [0,8] U [16, + )

y = 

 15x – 2 , x  (– , –1)

### Варіант 28

 –6x2 + 8 , x  [0,7)

y = 

 –x3/7 + 10 , x  (– , –10] U [11, + )

### Варіант 29

 –9x3 + 5x2 , x  (2,12] U (22,32)

y = 

 – x2 – 12 , x  (– , 0]

### Варіант 30

 – 5x3 + 10 , x  [8,23)

y = 

 2x3 + 8x2 , x  (– , –19) U (–3,0]

## 2. ЛАБОРАТОРНА РОБОТА №1.2. АЛГОРИТМИ З ВКЛАДЕНИМИ ЦИКЛАМИ ТА МЕТОД ДИНАМІЧНОГО ПРОГРАМУВАННЯ

**Мета лабораторної роботи**

Метою лабораторної роботи №1.2. «Алгоритми з вкладеними циклами та метод динамічного програмування» є засвоєння теоретичного матеріалу та набуття практичних навичок використа- ння різних циклічних керуючих конструкцій, вкладених циклів, методу динамічного програмування та обчислення кількості опера- цій алгоритмів.

## Постановка задачі

1. Задане натуральне число ***n***. Вирахувати значення заданої формули за варіантом.
2. Для вирішення задачі написати дві програми:
   1. перша програма повинна використовувати для обчи- слення формули вкладені цикли;
   2. друга програма повинна виконати обчислення форму- ли за допомогою одного циклу з використанням методу ди- намічного програмування.
3. Виконати розрахунок кількості операцій для кожного з алгоритмів за методикою, викладеною на лекції, додавши до неї підрахунок кількості викликів стандартних функцій.
4. Програма має правильно вирішувати поставлену задачу при будь-якому заданому ***n***, для якого результат обчислення може бути коректно представлений типом ***double***.
5. Результуючі дані вивести у форматі з сімома знаками після крапки.

## Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варі- анту.
2. Текст обох програм.
3. Розрахунки кількості операцій для кожного з алгоритмів згідно методики, викладеної на лекціях.
4. Тестування програм. З метою створення еталону для переві- рки коректності програм потрібно розписати у звіті обчислення за- даної формули для ***n***=3 з підстановкою фактичних чисел і виконати обчислення цієї формули на калькуляторі.
5. В якості результату тестування роздрукувати результати ви- конання обох програм для ***n***=1, ***n***=2 та ***n***=3.
6. Результати обчислень обома програмами та за допомогою калькулятора повинні співпадати.
7. Результати розрахунків обома програмами для заданого значення ***n***.

## Контрольні питання

1. Класифікація циклічних керуючих конструкцій алгоритмів та особливості їх використання.
2. Запис циклічних керуючих конструкцій алгоритмів діа- грамами дій.
3. Запис циклічних керуючих конструкцій алгоритмів мовою програмування.
4. Методика спрощеного підрахунку кількості операцій алгори- тмів.
5. Метод динамічного програмування: визначення та значи- мість при створенні алгоритмів.

## Варіанти завдань

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1. | *n* 4*i*  *i*  *S*   *i*  *i*  1   *j*  cos( *j*)  *j*  1 | | 2  . | *n i*2  1  *P*   *i*  *i*  1   *j*  sin( *j*)  *j*  1 | |
| 3. | *n* 3*i*  ln(*i*  1)  *S*   *i*  *i*  1  2 *j*  1  *j*  1 | | 4  . | *n* 2*i*  ln(*i*  3)  *P*   *i*  *i*  1  2 *j*  1  *j*  1 | |
| 5. | *n i*  2*i*  *S*   *i*  *j*  1  *i*  1    *j*  1  10  | | 6  . | *n i*  *i*  *P*   *i*  4   *i*  1  *j*  1  *j*  1  3  | |
| 7. | *n*  *S*    *i*  1 | *i*  (*i*  1) | 8  . | *n*  *P*    *i*  1 | *i*  *i* |
| *i*   ( *j*  1)  sin( *j*  *j*  1 | *i*    *j*  sin( *j*)  *j*  *j*  1 |
| 9. | *n* (2*i*  1)2  *S*   *i*  *i*  1   *j*  1  *j*  1 | | 1  0  . | *n i*  (*i*  1)  *P*   *i*  *i*  1  sin( *j*)  1  *j*  1 | |
| 11. | *n i*  cos(*i*)  *S*   *i*  *i*  1   *j*  sin( *j*)  *j*  1 | | 1  2  . | *n i*  cos(*i*)  *S*   *i*  *i*  1   *j*  sin( *j*)  *j*  1 | |
| 13. | *n* 3  sin2 (*i*)  *S*   *i*  *i*  1  ln( *j*  2)  *j*  1 | | 1  4  . | *n* cos(*i*)  1  *P*   *i*  *i*  1  sin( *j*)  *j*  1 | |
| 15. | *n* sin(*i*)  2  *S*   *i*  *i*  1 *i*   sin( *j*)  *j*  1 | | 1  6  . | *n* 3  cos2 (*i*)  *P*   *i*  *i*  1  ln( *j*  2)  *j*  1 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 17. | *i*    *j*  cos( *j*)  *n*  *S*   *j*  1  *i* 1 4*i*  *i* | 1  8  . | *i*  *n*   *j*  sin( *j*) *P*   *j*  1  *i*2  1  *i*  1 |
| 19. | *i*   2 *j*  1  *n*  *S*   *j*  1  *i* 1 3*i*  ln(*i*  1) | 2  0  . | *i*  *n*  2 *j*  1 *P*   *j*  1 *i* 1 2*i*  ln(*i*  3) |
| 21. | *i*  *j*  2      *n* 10  *S*   *j*  1    *i* 1 *i*  2*i* | 2  2  . | *i*  4 *j*     1  *n* 3  *P*   *j*  1    *i* 1 *i*  *i* |
| 23. | *i*   ( *j*  1)  sin( *j*  *n*  *S*   *j* 1  *i*1 *i*  (*i*  1) | 2  4  . | *i*  *n*   *j*  sin( *j*)  *j* *P*   *j*  1  *i* 1 *i*  *i* |
| 25. | *i*    *j*  1  *n*  *S*   *j* 1  *i* 1 2*i*  12 | 2  6  . | *i*  sin( *j*)  1  *n*   *P*   *j*  1  *i* 1 *i*  (*i*  1) |
| 27. | *i*    *j*  cos( *j*)  *n*  *S*   *j* 1  *i*1 *i*  cos(*i*) | 2  8  . | *i*  *n*   *j*  cos( *j*) *P*   *j*  1  *i* 1 *i*  sin(*i*) |
| 29. | *i*  *n*  ln( *j*  2)  *S*   *j* 1  *i*1 3  sin 2 (*i*) | 3  0  . | *i*  *n*  sin( *j*) *P*   *j*  1 *i* 1 cos(*i*)  1 |

**3. ЛАБОРАТОРНА РОБОТА №1.3. ВИКОРИСТАННЯ КЕРУЮЧИХ КОНСТРУКЦІЙ**

## РОЗГАЛУЖЕННЯ ТА ЦИКЛІВ ПРИ РОБОТІ З ОДНОВИМІРНИМИ МАСИВАМИ

**Мета лабораторної роботи**

Метою лабораторної роботи №1.3. «Використання керуючих конструкцій розгалуження та циклів при роботі з одновимірними масивами» є засвоєння теоретичного матеріалу та набуття практи- чних навичок створення складних комбінованих алгоритмів з ви- користанням конструкцій розгалуження, циклічних керуючих кон- струкцій та одновимірних масивів.

## Постановка задачі

1. Задано дійсні числа ***a***, ***b***, ціле число ***n*** та одновимірний масив (вектор) дійсних випадкових чисел ***Y***[***n***].
2. Отримати одновимірний масив (вектор) дійсних чисел ***Z***[***n***], елементи якого zi = ***f*** (y1, y2, … ,yn), де ***f*** – задана за варіантом фу- нкція, а yi – елементи вектора ***Y***[***n***].
3. Отримати результуюче значення ***R*** = ***g*** (z1, z2, … ,zn), де ***g*** – задана за варіантом функція, а zi – елементи вектора ***Z***[***n***].
4. Значення змінних ***a***, ***b*** та ***n*** є початковими даними, які виби- раються самостійно так, щоб функція ***f*** існувала при даних значе- ннях цих змінних.
5. Номери функцій ***f*** та ***g*** визначити за варіантом завдання (див. таблицю 1).
6. Програма має правильно вирішувати поставлену задачу при вхідних даних ***a***, ***b***, ***n***.
7. Значення початкового масиву ***Y***[***n***], проміжного масиву ***Z***[***n***] та результуюче значення ***R*** вивести у форматі з трьома знаками після крапки.

## Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варі- анту.
2. Текст програми, вхідні дані.
3. Тестові початкові дані для налагодження програми і ре- зультати виконання програми для цих тестових даних.
4. Діапазон генерування випадкових чисел для формування початкового масиву ***Y***[***n***] повинен бути встановлений таким чином, щоб згенеровані елементи масиву належали різним діапазонам існування заданих функцій.
5. Результати розрахунків для довільних значень ***a***, ***b*** та ***n***.

## Контрольні питання

1. Класифікація керуючих конструкцій та особливості їх ви- користання.
2. Класифікація циклічних керуючих конструкцій алгоритмів та особливості їх використання.
3. Які існують варіанти використання вкладених умовних кон- струкцій (при наявності альтернативної гілки та при її відсутності) та особливості запису таких вкладених конструкцій мовою про- грамування?
4. Яким умовам повинен задовольняти лічильник (параметр) конструкції циклу з лічильником та початкове і кінцеве значення лічильника?
5. Чи відповідає структурному стилю програмування викори- стання значення лічильника (параметра) конструкції циклу з лічи- льником після завершення циклу та після переривання роботи ци- клу конструкцією виходу з циклу?
6. Які подібності та відмінності мають циклічні конструкції з передумовою та післяумовою?

## Варіанти завдань

*Таблиця 1*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ варі- анту** | **Функ- ція**  **f** | **Функ-**  **ція g** | **№**  **варі- анту** | **Функ- ція**  **f** | **Функ-**  **ція g** |
| **1** | 1 | 1 | **16** | 6 | 9 |
| **2** | 2 | 2 | **17** | 7 | 10 |
| **3** | 3 | 3 | **18** | 8 | 1 |
| **4** | 4 | 4 | **19** | 9 | 2 |
| **5** | 5 | 5 | **20** | 10 | 3 |
| **6** | 6 | 6 | **21** | 1 | 7 |
| **7** | 7 | 7 | **22** | 2 | 8 |
| **8** | 8 | 8 | **23** | 3 | 9 |
| **9** | 9 | 9 | **24** | 4 | 10 |
| **10** | 10 | 10 | **25** | 5 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **11** | 1 | 4 | **26** | 6 | 2 |
| **12** | 2 | 5 | **27** | 7 | 3 |
| **13** | 3 | 6 | **28** | 8 | 4 |
| **14** | 4 | 7 | **29** | 9 | 5 |
| **15** | 5 | 8 | **30** | 10 | 6 |

### Функція f(y1, … , yn) для отримання ряду значень zi

1. *zi* = *f* ( *y*1 *,* . .. *, yn* )={

2. *zi* = *f* ( *y*1 *,* . .. *, yn* )={

*yi* +5b *при yi* ∉[11 *,* 33] *,*

*yi* −10 *a при yi* ∈[11 *,* 33].

*ay2 при*∣*y* ∣≤2,

*i*

*i*

3. *zi* = *f* ( *y*1 *,* . .. *, yn* )={

0,5*by i при*∣*y i*∣>2.

2a + *y*2 *при y* ∈(0, 15) *,*

*i*

*i*

4. *zi* = *f* ( *y*1 *,* . .. *, yn* )=

2b *при yi* ∉( 0, 15).

5− *yi при*∣*yi*∣>*a ,*

{

3*byi при*∣*y i*∣≤*a* .

2y3 *при*∣ *y* ∣≤4b *,*

{

*i i*

5. *zi* = *f* ( *y*1 *,* . .. *, yn* )=

6. *zi* = *f* ( *y*1 *,* . .. *, yn* )={

*ayi* −2, *при*∣*yi*∣>4b .

7−*byi при yi*∉(−20 *,* 10) *,*

1−*ayi при yi* ∈(−20 *,* 10) .

*yi* + *ab при yi* ∈[−15 *,* 25] *,*

7. *zi* = *f* ( *y*1 *,* . .. *, yn* )={

2a+3b *при yi* ∉[−15 *,* 25 ].

*a при*∣ *y* ∣>7a *,*

8. {∣ *yi*∣+3 *i*

*zi* = *f* ( *y*1 *,* . .. *, yn* )=

6*by при*∣*y* ∣≤7a .

*i i*

9. *zi* = *f* ( *y*1 *,* . .. *, yn* )=

10 *a*− *yi при yi* ∈(−50 ,0) *,*

1+ *y* 2 *при y* ∉(−50 ,0) .

{

*i i*

*ay3 при*∣*y* ∣≤10 *,*

{

*i i*

10. *zi* = *f* ( *y*1 *,* . .. *, yn* )=

1+ *y i при*∣*yi*∣>10 .

### Функція g(z1, … , zn) для отримання результуючого значення R

1. *R*= *g* ( *z*1

2.

3.

1. *R*= *g* ( *z*1
2. *R*= *g* ( *z*1
3. *R*= *g* ( *z*1

*,*. . . *, zn*

*,*. . . *, zn*

*,*. . . *, zn*

*,*. . . *, zn*

)= min ((−1)*i az* )

1≤*i* ≤*n*

*R*= *g* ( *z ,*. . . *, z* )=max ((−1)*i b*+ *z* )

1

*n*

1≤*i* ≤*n n*

*i*

*R*= *g* ( *z ,*. . . *, z* )= (−

1

*n*

∏

1 )

*i*

*i* =1

*z i* +*a*

*ib*

*i*

)=max∣*z* −(−1)*i ab*∣

*i*

1≤*i* ≤*n*

)= min ∣(−1)*i bz* −4 *ia*∣

*i*

1≤*i* ≤*n n*

)=∑(−1)*i*+1( *az* −3b)

*i*

*i*=1

*n*  2

1. *R*= *g* ( *z*

*,*. . . *, z* )=∑(−1)*i* (√ *z*2+ *a*2−*z* )

1 *n*

*i*=1

*n*

*i i*

*a* + *z* 2

1. *R*= *g* ( *z*

*,*. . . *, z* )=∏(−1 )*i*+1 *i*

1 *n*

*i* =1

2+ *b*2

1. *R*= *g* ( *z*1
2. *R*= *g* ( *z*1

*,*. . . *, zn*

*,*. . . *, zn*

*n*

)=∑(−1)*i* √10 *b*+ *az2*

*i*

*i*=1

*n*

)=∏(−1 )*i*+1 ( 6 *bz* −*i*2 *a* )

*i*

*i* =1

## 4. ЛАБОРАТОРНА РОБОТА №1.4. АЛГОРИТМИ ПЕРЕТВОРЕННЯ

**ОДНОВИМІРНИХ МАСИВІВ (ВЕКТОРІВ)**

## Мета лабораторної роботи

Метою лабораторної роботи №1.4. «Алгоритми перетворення одновимірних масивів (векторів)» є засвоєння теоретичного матері- алу та набуття практичних навичок створення складних алгоритмів перетворення даних у одновимірних масивах.

## Постановка задачі

Задано натуральне число ***n*** і одновимірний масив цілих чисел

***A***[***4n***], який складається з чотирьох частин по ***n*** елементів кожна.

1 n n+1 2n 2n+1 3n 3n+1 4n

**1-ша частина**

**2-га частина**

**3-тя частина**

**4-та частина**

Перетворити масив «на тому ж місці» (тобто результат пере- творення повинен бути у тому ж масиві ***A***), виконавши пере- становки елементів масиву ***A*** у порядку, заданому за варіантом.

Розв’язати задачу двома способами (написати дві програми):

* 1. для виконання перетворення масиву ***A*** в програмі дозволяє- ться використовувати один додатковий проміжний масив ***В*** розмі- ром не більше ***4n*** елементів. Кращим варіантом рішення буде ви- користання проміжного масиву ***В*** розміром у ***n*** елементів;
  2. для виконання перетворення масиву ***A*** в програмі дозволяє- ться використати тільки одну просту додаткову проміжну змінну цілого типу.

## Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варі- анту.
2. Текст обох програм.
3. Тестування обох програм. При тестуванні кожної з програм треба підібрати такий набір вхідних значень масиву ***A***, щоб можна було легко відстежити коректність перетворення масиву. Необхідно протестувати мінімум три випадки: 1) ***n*** = 1; 2) ***n*** – парне; 3) ***n*** – непарне.
4. В якості результату роздрукувати дані тестування для всіх вхідних значень масиву ***А*** з тестового набору для кожної з програм окремо.

## Контрольні питання

1. Класифікація масивів.
2. Запис оголошення та використання одновимірних масивів (векторів) діаграмами дій.
3. Запис оголошення та використання одновимірних масивів (векторів) мовою програмування.
4. Що означає термін «перетворення масиву»?
5. Що є необхідним для виконання перетворення масиву?

## Варіанти індивідуальних завдань

### Умовні позначення.

1. Цифри 1, 2, 3, 4 позначають номер частини початкового масиву ***А***. У варіантах завдання ці цифри стоять на тих місцях, на які повинна бути переставлена позначена вказаним номером части- на масиву ***А***.
2. Стрілка «вправо» () позначає, що порядок взаємного розташування елементів відповідної частини масиву ***А*** після пере- становки повинен бути збережений таким самим, як у початковому масиві.
3. Стрілка «вліво» () позначає, що порядок взаємного розташування елементів відповідної частини масиву ***А*** після пере- становки повинен бути оберненим до того розташування, яке було у початковому масиві, тобто під час перестановки цієї частини масиву виконується «дзеркальне відображення» її елементів.

### Варіант 1

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **1**   | **4**   | **2**   | **3**   |

### Варіант 2

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **4**   | **1**   | **2**   | **3**   |

### Варіант 3

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **4**   | **1**   | **2**   |

### Варіант 4

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **1**   | **4**   | **3**   | **2**   |

### Варіант 5

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **4**   | **1**   | **2**   |

### Варіант 6

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **4**   | **2**   | **3**   | **1**   |

### Варіант 7

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **4**   | **2**   | **1**   | **3**   |

### Варіант 8

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **4**   | **1**   | **2**   | **3**   |

### Варіант 9

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **4**   | **3**   | **2**   | **1**   |

### Варіант 10

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **4**   | **1**   | **2**   |

### Варіант 11

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **1**   | **4**   | **2**   |

### Варіант 12

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **4**   | **1**   | **2**   |

### Варіант 13

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **2**   | **4**   | **1**   |

### Варіант 14

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **2**   | **1**   | **4**   | **3**   |

### Варіант 15

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **2**   | **3**   | **4**   | **1**   |

### Варіант 16

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **2**   | **1**   | **4**   |

### Варіант 17

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **2**   | **4**   | **3**   | **1**   |

### Варіант 18

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **2**   | **4**   | **3**   | **1**   |

### Варіант 19

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **4**   | **3**   | **2**   | **1**   |

### Варіант 20

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **2**   | **4**   | **1**   | **3**   |

### Варіант 21

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **2**   | **3**   | **4**   | **1**   |

### Варіант 22

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **4**   | **3**   | **2**   | **1**   |

### Варіант 23

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **1**   | **2**   | **4**   |

### Варіант 24

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **4**   | **3**   | **2**   | **1**   |

### Варіант 25

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **2**   | **4**   | **1**   | **3**   |

### Варіант 26

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **1**   | **4**   | **2**   |

### Варіант 27

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **4**   | **2**   | **1**   |

### Варіант 28

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **4**   | **3**   | **1**   | **2**   |

### Варіант 29

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **3**   | **1**   | **2**   | **3**   |

### Варіант 30

1 n n+1 2n 2n+1 3n 3n+1 4n

|  |  |  |  |
| --- | --- | --- | --- |
| **2**   | **1**   | **4**   | **3**   |

## 5. ЛАБОРАТОРНА РОБОТА №1.5. АЛГОРИТМИ ЛІНІЙНОГО ПОШУКУ

**Мета лабораторної роботи**

Метою лабораторної роботи №1.5. «Алгоритми лінійного пошуку» є засвоєння теоретичного матеріалу та набуття практи- чних навичок рішення задачі пошуку заданої категорії елементів за допомогою різних алгоритмів методу лінійного пошуку у двовимі- рних масивах.

## Постановка задачі

1. Написати програму розв’язання задачі пошуку (за варіантом) у двовимірному масиві (матриці) одним з алгоритмів методу лі- нійного пошуку.
2. Розміри матриці ***m*** та ***n*** взяти самостійно у межах від 7 до 10.
3. Виконати тестування та налагодження програми на комп’ютері. При тестуванні програми необхідно підбирати такі вхі- дні 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 до- зволяється використовувати матриці меншого розміру.

## Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варі- анту.
2. Текст програми, вхідні дані.
3. Тести для налагодження програми і результати, отримані для них на комп’ютері.

## Контрольні питання

1. Які умові закінчення лінійного пошуку?
2. Написати алгоритм лінійного пошуку за допомогою циклу з лічильником та конструкцією виходу з циклу.
3. Написати алгоритм лінійного пошуку за допомогою циклу з передумовою і без використання «бар’єру».
4. Написати алгоритм лінійного пошуку за допомогою циклу з передумовою і з використанням «бар’єру».
5. Чим відрізняється загальна поведінка алгоритмів лінійного пошуку з використанням «бар’єру» і без його використання?

## Варіанти індивідуальних завдань

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

### Варіант 1

Задано матрицю дійсних чисел A[m,n]. У кожному рядку матриці визначити присутність заданого дійсного числа X і його місцезнаходження (координати).

### Варіант 2

Задано матрицю дійсних чисел A[n,n]. У головній діагоналі матриці знайти перший додатний і останній від’ємний елементи, а також поміняти їх місцями.

### Варіант 3

Задано матрицю дійсних чисел A[m,n]. При обході матриці по рядках знайти в ній перший додатний елемент і його місце- знаходження (координати).

### Варіант 4

Задано матрицю дійсних чисел A[m,n]. У кожному стовпчику матриці визначити присутність заданого дійсного числа X і його місцезнаходження (координати).

### Варіант 5

Задано матрицю дійсних чисел A[n,n]. У побочній діагоналі матриці знайти перший мінімальний і останній максимальний елементи, а також поміняти їх місцями.

### Варіант 6

Задано матрицю дійсних чисел A[m,n]. При обході матриці по стовпчиках знайти в ній перший мінімальний елемент і його місцезнаходження (координати).

### Варіант 7

Задано матрицю дійсних чисел A[m,n]. У кожному рядку матриці знайти останній від’ємний елемент і його місцезнаходжен- ня (координати).

### Варіант 8

Задано матрицю дійсних чисел A[n,n]. У головній діагоналі матриці знайти перший від’ємний і останній додатний елементи, а також поміняти їх місцями.

### Варіант 9

Задано матрицю дійсних чисел A[m,n]. При обході матриці по рядках знайти в ній останній нульовий елемент і його місце- знаходження (координати).

### Варіант 10

Задано матрицю дійсних чисел A[m,n]. У кожному стовпчику матриці знайти останній максимальний елемент і його місце- знаходження (координати).

### Варіант 11

Задано матрицю дійсних чисел A[n,n]. У побочній діагоналі матриці знайти перший максимальний і останній мінімальний елементи, а також поміняти їх місцями.

### Варіант 12

Задано матрицю дійсних чисел A[m,n]. При обході матриці по стовпчиках знайти в ній останній мінімальний елемент і його місцезнаходження (координати).

### Варіант 13

Задано матрицю дійсних чисел A[m,n]. У кожному рядку матриці знайти перший максимальний елемент і його місце- знаходження (координати).

### Варіант 14

Задано матрицю дійсних чисел A[n,n]. У головній діагоналі матриці знайти перший мінімальний і останній максимальний елементи, а також поміняти їх місцями.

### Варіант 15

Задано матрицю дійсних чисел A[m,n]. При обході матриці по рядках визначити в ній присутність заданого дійсного числа X і його місцезнаходження (координати).

### Варіант 16

Задано матрицю дійсних чисел A[m,n]. У кожному стовпчику матриці знайти перший від’ємний елемент і його місцезнаходження (координати).

### Варіант 17

Задано матрицю дійсних чисел A[n,n]. У побочній діагоналі матриці знайти перший додатний і останній від’ємний елементи, а також поміняти їх місцями.

### Варіант 18

Задано матрицю дійсних чисел A[m,n]. При обході матриці по стовпчиках визначити в ній присутність заданого дійсного числа X і його місцезнаходження (координати).

### Варіант 19

Задано матрицю дійсних чисел A[m,n]. У кожному рядку матриці знайти перший нульовий елемент і його місцезнаходження (координати).

### Варіант 20

Задано матрицю дійсних чисел A[n,n]. У головній діагоналі матриці знайти перший максимальний і останній мінімальний елементи, а також поміняти їх місцями.

### Варіант 21

Задано матрицю дійсних чисел A[m,n]. При обході матриці по рядках знайти в ній останній максимальний елемент і його місцезнаходження (координати).

### Варіант 22

Задано матрицю дійсних чисел A[m,n]. У кожному стовпчику матриці знайти останній додатний елемент і його місцезнаходжен- ня (координати).

### Варіант 23

Задано матрицю дійсних чисел A[n,n]. У побочній діагоналі матриці знайти перший від’ємний і останній додатний елементи, а також поміняти їх місцями.

### Варіант 24

Задано матрицю дійсних чисел A[m,n]. При обході матриці по стовпчиках знайти в ній перший від’ємний елемент і його місце - знаходження (координати).

### Варіант 25

Задано матрицю дійсних чисел A[m,n]. У кожному рядку матриці знайти останній мінімальний елемент і його місце- знаходження (координати).

### Варіант 26

Задано матрицю дійсних чисел A[m,n]. При обході матриці по стовпчиках знайти в ній останній додатний елемент і його місцезнаходження (координати).

### Варіант 27

Задано матрицю дійсних чисел A[m,n]. При обході матриці по рядках знайти в ній перший максимальний елемент і його місце- знаходження (координати).

### Варіант 28

Задано матрицю дійсних чисел A[m,n]. У кожному стовпчику матриці знайти перший мінімальний елемент і його місцезнаходже- ння (координати).

### Варіант 29

Задано матрицю дійсних чисел A[n,n]. У побочній діагоналі матриці визначити присутність заданого дійсного числа X і його місцезнаходження (координати).

### Варіант 30

Задано матрицю дійсних чисел A[m,n]. При обході матриці по рядках знайти в ній останній мінімальний елемент і його місце- знаходження (координати).

## ЛАБОРАТОРНА РОБОТА №1.6. АЛГОРИТМИ ОБХОДУ ДВОВИМІРНИХ МАСИВІВ

**(МАТРИЦЬ)**

## Мета лабораторної роботи

Метою лабораторної роботи №1.6. «Алгоритми обходу двовимірних масивів (матриць)» є засвоєння теоретичного матері- алу та набуття практичних навичок гнучкої роботи при реалізації алгоритмів з використанням двовимірних масивів (матриць).

## Постановка задачі

1. Оскільки екран монітора має площинні координати так само, як і двовимірний масив (матриця), але, на відміну від останнього, надає можливість візуально спостерігати виконання способу об- ходу, дана лабораторна робота виконується в координатах екрану монітора (дивись методичні вказівки до виконання даної ла- бораторної роботи).
2. Завданням даної лабораторної роботи є виконання заданого за варіантом способу обходу на екрані монітору в текстовому ре- жимі, проставляючи довільний символ клавіатури (наприклад ‘\*’) у порядку заданого способу обходу.
3. Оскільки при виводі символу у правий ніжній кут екрану від- бувається зсув зображення на один рядок вгору (якщо тільки не ви- користовується прямий доступ до відеопам’яті), *останній рядок екрану монітора при виконанні завдання заповнювати не треба*.

## Зміст звіту

1. Загальна постановка задачі та завдання для конкретного варі- анту з рисунком заданого обходу двовимірного масиву (матриці).
2. Текст програми.
3. Тестування програми. При тестуванні необхідно уважно від- стежити правильність заповнення екрану монітора символами (ви- конання обходу матриці), при повільному виводі символів (викори- стовуючи функції затримки виконання програми).
4. Окрім текстового звіту, в якості результату, продемонструвати викладачу правильність роботи програми при повільному виводі символів.

## Контрольні питання

1. Класифікація масивів.
2. Запис оголошення та використання двовимірних масивів діа- грамами дій.
3. Запис оголошення та використання двовимірних масивів мовою програмування.
4. Що означає термін «обхід двовимірного масиву (матриці)»?
5. Скільки циклів та індексних змінних потрібно для виконання обходу двовимірного масиву (матриці)?
6. Як визначити спосіб простого обходу (по рядках чи по стовп- чиках), використаного у програмі, за виглядом заголовків циклів та використання індексних змінних при звертанні до елементів маси- ву?

## Методичні вказівки до виконання завдання

* 1. Екран монітора комп’ютера у стандартному текстовому ре- жимі має 25 рядків і 80 стовпчиків. Координати кутових та центральних (якщо ширину екрану вважати рівною 24 рядкам) позицій екрану монітора комп’ютера показані на наступному рису- нку,

**(1,1)**

**(1,25)**

**(80,25)**

Останній рядок екрану заповнювати НЕ треба



|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| **(40,12)** | | **(41,12)** | |
| **(40,13)** | | **(41,13)** | |
|  |  | |  |

**(80,25)**

При виконанні завдання даної лабораторної роботи, при правильно складених алгоритмах, зображення на екрані монітора буде автоматично «підніматись» на один рядок вгору в момент виводу символа у правий ніжній кут екрану з координатами (80,25). *Тому, при складанні алгоритму останній рядок екрану монітора при виконанні завдання заповнювати не треба, тобто*

*останнім рядком вважати 24-й рядок.*

Потрібно зауважити, що можна написати правильну про- граму, яка буде коректно заповнювати і 25-й рядок, але для цього треба використовувати прямий доступ до відеопам’яті комп’ютера).

* 1. Треба звернути увагу на протилежний спосіб нумерації (відкладення) координат у двовимірному масиві (матриці) та на екрані монітора.

При звертанні до елемента двовимірного масиву ***A***[ ***i*** , ***j*** ], перший індекс i є номером рядка і відкладається по вертикалі (вниз), а другий індекс j є номером стовпчика і відкладається по горизонталі (вправо).

При звертанні до позиції екрану монітора (найчастіше за допомогою процедури/функції типу ***gotoxy(X,Y)***, **навпаки**, перший індекс ***X*** є номером стовпчика і відкладається по горизонталі (вправо), а другий індекс ***Y*** є номером рядка і відкладається по вертикалі (вниз).

З врахуванням цієї відмінності, в усіх інших аспектах алгори- тми реалізації обходу двовимірного масиву (матриці) та обходу позицій екрану монітора будуть однаковими.

* 1. Враховуючи високу швидкодію сучасних комп’ютерів екран монітора буде заповнюватись миттєво, в результаті чого від- стежити коректність виконання обходу неможливо без використан- ня в програмі процедур або функцій затримки виконання програми на певний проміжок часу після виводу кожного наступного сим- вола.

У мовах програмування процедура/функція/метод затримки, як правило, має один параметр, який визначає тривалість затримки виконання програми у мілісекундах.

У мовах Turbo Pascal та Free Pascal така процедура має ідентифікатор ***Delay*** і знаходиться у модулі ***Crt***.

У мові С для затримки використовується функція ***usleep()***, яка вимагає підключення заготовочного файлу <***unistd.h***>.

Для виконання затримки у мові C++ можуть викори- стовуватись функції/методу ***Sleep(), sleep()*** або ***delay()*** (Borland C++ Builder 3.1), підключення яких є різним, в залежності від компіля- тора мови С++ та операційної системи (Windows, Linux, MSDOS).

Затримка у мові С# виконується за допомогою методу ***Thread.Sleep()***, для використання якого попередньо треба підклю- чити модуль ***System.Threading***.

У мові Java затримка виконується так само, як у мові C#, з ті- єю відмінністю, що для її використання додаткові оператори під- ключення не потрібні.

## Умовні позначення на схемах варіантів завдань

1. Якщо на схемі завдання зображена тільки одна спірале- подібна чи зигзагоподібна стрілка, то символи на екран повинні бу- ти виведені строго згідно зміні напрямків лінії стрілки та її переги- нів.
2. Якщо на схемі завдання зображені дві зигзагоподібні стрілки, то символи на екран повинні бути виведені строго згідно

зміні напрямків лінії стрілки та її перегинів *почергово*, тобто за принципом «перший символ згідно першої стрілки, перший символ згідно другої стрілки, другий символ згідно першої стрілки, другий символ згідно другої стрілки, і т.д.».

**Варіанти індивідуальних завдань**

# 1.



**2.**



# 3.



**4.**

**5.**

**6.**



**7.**

**8.**

**9.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |



**10.**

**11.**

**12.**



**13.**

**14.**

**15.**



**16.**

**17.**

**18.**



**19.**

**20.**

**21.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |



**22.**

**23.**

**24.**



**25.**

**26.**

**27.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |



**28.**

**29.**

**30.**

## 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. Получи- сленные алгоритмы., М.: Мир, 1977.
6. Кнут Д. Искусство программирования для ЭВМ. т.3. Сорти- ровка и поиск., М.: Мир, 1978.

## Додаткова література

1. Вирт Н. Алгоритмы + структуры данных = программы. М.:Мир, 1985.
2. Г.Буч. Объектно-ориентированное проектирование с приме- рами применения: Пер. с англ. – М.: Конкорд, 1992. – 519 с., ил.
3. Методичні вказівки для виконання лабораторних робіт з дисципліни „Програмування” для студентів спеціальностей

„Комп’ютерні системи та мережі”, „Спеціалізовані комп’ютерні си- стеми”, „Системне програмування” , ч. 1,2 (Укл. Р.Ф. Колінько, О.І. Марченко). Київ НТУУ „КПІ” – 2004, видавництво „Ювета”.

1. Абрамов С.А., Гнездилова Г.Г., Капустина Е.Н., Селюн, М.И. Задачи по программированию., М.: наука, 1988.
2. Martin J., McClure C. Structured techniques : The basis for CASE., 1988.
3. Ахо А., Xопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов., М.: Мир, 1979.
4. Гудман С., Xидетниеми С. Введение в разработку и анализ алгоритмов., М.: Мир, 1981.
5. Дал У., Дейкстра., Xоор К. Структурное программирование., М.: Мир, 1975.
6. Дейкстра Э. Дисциплина программирования. М.: Мир, 1978.
7. Вирт Н. Систематическое программирование. Введение., М.: Мир,1977.
8. Лингер Р., Миллс X., Уитт Б. Теория и практика структурно- го программирования., М.: Мир, 1982.
9. Брукс Ф.П. Как проектируются и создаются программные комплексы., М.: Наука, 1979.