# 3. Масиви та ініціалізація класів

## Масиви

Ввести $n$ чисел з консолі.

1.1. Створіть масив об'єктів String та ініціалізуйте його. Роздрукуйте масив за допомогою циклу for () в один рядок через кому та роздрукуйте в кожний рядок в новому рядку з вказанням індексу масиву.

1.2. Напишить програму MaxMin.java що дозволяє користувачу ввести скільки завгодно цілих чисел в новому рядку доки той не введе число 0 та виводить найбільше та найменше значення серед цих чисел.

1.3. Напишить програму, що приймає з консолі число $n$, читає $n$ дійсних чисел з консолі, та знаходить середнє з цих чисел(mean) та відхилення (sample standard deviation).

1.4. Напишить програму, що вводить цілі числа та виводить число яке зустрічається підряд найбільшу кількість разів та скільки саме разів (якщо таких чисел декілька - то виводиться найбільше з таких чисел) в заданому нижче форматі.

Приклад: 1 2 2 1 5 1 1 7 7 7 7 1 1

Результат: 4 підряд чисел 7.

1.5. Напишить програму Closest.java що вводить з командного рядку числа $x, y, z$ (координату точки A) та читає з консолі масив чисел $(x_i, y_i, z_i)$, і виводить ту з нових точок, що найближча до $(x, y, z)$.

1.6. Знайти серед масиву чисел таке число, в якому кількість цифр мінімальна. Якщо їх декілька, виведіть всі з них.

1.7. Знайдіть в масиві кількість чисел, що містять лише парні цифри, та кількість чисел з рівною кількістю парних та непарних цифр.

1.8. Знайти в масиві число, цифри в якому йдуть по зростанню. Якщо їх декілька, виведіть всі.

1.9. Знайти в масиві число, яке містить лише різні цифри. Якщо їх декілька, знайдіть найбільше з них.

1.10. Серед чисел знайти число-паліндром. Якщо таких чисел більше за одне, то виведіть друге з них.

1.11. Вводяться двовимірні координати та маси послідовності об'єктів (кожен об'єкт в окремому рядку, дані - через кому). Напишіть програму

для обчислення їх центру мас чи центроїда. Центроїд - це середнє положення $n$ об'єктів, зважене за масою. Програма зчитує послідовності позицій і мас $(x_i, y_i, m_i)$ з консолі і роздруковує їх центр мас $(x, y, m)$.

1.12. Кількість слів і рядків. Напишить програму, щоб вона читала текст з консолі та друкувала кількість символів, слів та рядків у тексті.

1.13. Проблема автостопщика (Проблема вимогливої нареченої). Ви опитуєте N кандидатів на єдину посаду Керівника. Щохвилини ви бачите нового кандидата, і у вас є одна хвилина, щоб вирішити, оголосити цю особу Керівником чи ні. Після закінчення співбесіди з кандидатом ви не можете змінити свою думку. Припустимо, що ви можете відразу оцінити кожного кандидата одним дійсним числом від 0 до 1, але, звичайно, ви не знаєте рейтингу кандидатів, яких ще не бачили. Розробити стратегію та написати програму Menager, яка має принаймні 25% шансів вибрати найкращого кандидата (за умови, що кандидати прибудуть у довільному порядку), зчитавши 500 значень даних зі стандартного введення. Рішення: взяти інтерв'ю протягом $N/2$ хвилин і записати рейтинг найкращого кандидата, побаченого на даний момент. Протягом наступних 2 хвилин виберіть першого кандидата, який має вищий рейтинг, ніж записаний. Це дає принаймні 25% шансів, оскільки ви отримаєте найкращого кандидата, якщо другий найкращий кандидат прибуде протягом перших N/2 хвилин, а найкращий кандидат прибуде в останні $N/2$ хвилини. Це можна трохи покращити до $1/e = 0.36788$, використовуючи по суті таку саму стратегію, але перемикаючи її в момент $N/e$.

1.14. Двадцять питань. Напишіть програму QuestionsTwenty.java, яка виводить 20 запитань: користувач задумує число від 1 до мільйона, а комп'ютер здогадується, яке число запитуючи користувач задачючи питання: Більше чи менше чи дорівнює воно якомусь числу. Використовуйте двійковий пошук, щоб переконатися, що комп'ютеру потрібно не більше 20 відгадок.

1.15. Напишіть програму Pnorm.java, яка приймає аргумент командного рядка $p$, зчитує дійсні числа зі стандартного вводу та роздруковує їх $p$-норму, де $p$-норма вектора $(x_1, ..., x_N)$ $p$ з $(|x_1|^p + |x_2|^p + \ldots + |x_N|^p)$.

1.16. Декілька однакових чисел в масиві, що йдуть одне за одним згортаються в одне число, яке є на одиницю більше за початкове.

Приклад: 1 1 1 4 перетвориться в 2 4, так як 1 1 1 згорнеться в 2.

Масив піддається згортці допоки в ньому знаходяться повторювані поспіль числа.

Згортка проводиться завжди з лівого краю і як тільки найлівіша послідовність до кінця згорнута, можна приступати до наступних чисел. Наприклад: 1 1 2 2 3 -> 2 2 2 3 -> 3 3 -> 4

На вхід програма отримує: в першому рядку число N - довжина масиву

В наступному рядку N цілих чисел - елементи масиву.

Необхідно вивести згорнутий масив.

1.17. У масивах монотонний відрізок - це кілька поспіль елементів які зростають, або спадають. Наприклад: масив -1 3 2 5 6 7 3 2 1 1 1 має монотонні відрізки: -1 3, 2 5 6 7, 7 3 2 1, 1, 1.

Завдання полягає у виведенні всіх монотонних відрізків максимальної довжини. Для прикладу вище: такий відрізок всього один довжиною 4: 7 3 2 1, а в прикладі 1 2 1 2 1 таких відрізків 4: 1 2, 2 1, 1 2, 2 1

Необхідно враховувати обидва порядки!

На вхід програма отримує в першому рядку довжину масиву, далі йдуть елементи масиву.

Виведіть всі відрізки, кожен з нового рядка.

1.18. Програма отримує на вхід довжину ялинки.

Програма повинна надрукувати ялинку. Sample Input 1:

3

Sample Output 1:

```
..*..
.***.
*****
```

Sample Input 2:

4

Sample Output 2:

```
...*...
..***..
.*****.
```

На вхід подається довжина масиву та масив з 0 та 1.

Приклад:

4

1 1 0 0

1 - підйом(/), 0 - спуск (\)

В даному прикладі треба вивести горний масив:



Гарантується, що послідовність 0 та 1 утворює коректний горний масив.

1.19. Спіраль на кожному повороті збільшує довжину прямого шляху на 1, початкова довжина - 2, початковий напрямок - вгору, повороти лише направо.

YYYYY
YXXXX
YXYYY
YXYXY
YXXXY
YYYYY
Дано число: кількість Y спіралі, які необхідно вивести.

Виведіть спіраль, заповнивши порожні місця точками.

1.20. Контрольні суми. Міжнародний стандартний номер книги (ISBN) - це 10 -значний код, який однозначно визначає книгу. Крайня права цифра - це цифра контрольної суми, яку можна однозначно визначити з інших 9 цифр за умови, що $d_1 + 2 * d_2 + 3 * d_3 + \ldots + 10 * d_{10}$ має бути кратним 11 (тут $d_i$ позначає $i$-ту цифру справа). Цифра контрольної суми $d_1$ може бути будь -яким значенням від 0 до 10: за умовою ISBN використовується значення X для позначення 10. Приклад: цифра контрольної суми, що відповідає 020131452, дорівнює 5, оскільки це єдине значення $d_1$ від 0 до 10, для якого $d_1 + 2 * 2 + 3 * 5 + 4 * 4 + 5 * 1 + 6 * 3 + 7 * 1 + 8 * 0 + 9 * 2 + 10 * 0$ кратно 11. Напишіть програму ISBN.java, що приймає 9-значне ціле число як аргумент командного рядка, обчислює контрольну суму та друкує 10-значний номер ISBN. Це нормально, якщо ви не надрукуєте жодних нулів.

1.21. Форматований номер ISBN. Напишіть програму ISBN2.java, яка зчитує 9-значне ціле число з аргументу командного рядка, обчислює контрольну цифру та друкує повністю відформатований номер ISBN, наприклад, 0-201-31452-5.

1.22. Коди UPC. Універсальний код продукту (UPC) - це 12 -значний код, який однозначно визначає продукт. Найменш значуща цифра d1 (крайня права) - це контрольна цифра, яка визначається однозначно, зробивши такий вираз кратним 10: $(d_1 + d_3 + d_5 + d_7 + d_9 + d_{11}) + 3(d_2 + d_4 + d_6 + d_8 + d_{10} + d_{12})$ Наприклад, контрольна цифра, що відповідає 0-48500-00102 (Tropicana Pure Premium Orange Juice), дорівнює 8,

оскільки (8 + 0 + 0 + 0 + 5 + 4) + 3 (2 + 1 + 0 + 0 + 8 + 0) = 50 і 50 кратно 10. Напишіть програму, яка зчитує 11 -значне ціле число з параметра командного рядка, обчислює контрольну цифру та друкує повний UPC. Підказка: використовуйте змінну типу long для збереження 11 -значного числа.

1.23. Внесення змін. Напишіть програму, яка зчитує в командному рядку ціле число N (кількість пенні) і друкує найкращий спосіб (найменша кількість монет) для нарахування даної суми за допомогою монет (лише 1,25, 10 та 50 копійок). Наприклад, якщо N = 73, надрукуйте

- 50 коп
- 10 коп * 2
- 1 коп * 3

Підказка: використовуйте жадібний алгоритм.

1.24. Рюкзак. Дано: кількість предметів, ваги предметів та вагу, яка може витримати рюкзак.

Необхідно вибрати кілька предметів і покласти їх в рюкзак так, щоб не перевищити його максимальну вагу. Це класична NP повне завдання. Поліноміального (оптимального) рішення немає. Завдання вирішується рекурсивно повним перебором. Використовуйте деревоподібну рекурсію, де в одному випадку ми беремо предмет і йдемо до наступного, а в іншому не беремо і йдемо до наступного.

Необхідно вивести: максимальну сумарну вагу предметів, які можна покласти у рюкзак.

Вводяться два цілих числа N - кількість предметів W - максимальна вага рюкзака.

Далі N цілих чисел - ваги предметів.

Потрібно підрухувати та вивести ціле число – максимальну сумарна вага всіх предметів, які можна покласти у рюкзак.

Приклад 1:

5 12 1 2 3 4 5

Реузльтат 1:

12

Приклад 2:

5 15 2 2 3 3 10

Реузльтат 2:

15

Sample Input 3:

5 17 1 3 18 3 15

Sample Output 3:

16

## Батовимірні масиви

Ввести з консолі матрицю a[n][m] для натуральних $n, m$. Написати методи задання значення елементів матриці в інтервалі значень від -n до n за допомогою датчика випадкових чисел та за допомогою консолі та виводу матриці.

2.1. Впорядкувати рядки (стовпці) матриці в порядку зростання значень елементів рядка(стовпчика).

2.2. Виконати циклічний зсув заданої матриці на k позицій вправо (вліво, вгору, вниз).

2.3. Знайти і вивести найбільше число зростаючих елементів матриці, що йдуть підряд.

2.4. Знайти суму елементів матриці, розташованих між першим і другим додатними елементами кожного рядка.

2.5. Транспонувати квадратну матрицю.

2.6. Обчислити норму матриці.

2.7. Повернути матрицю на 90 (180, 270) градусів проти годинникової стрілки.

2.8. Обчислити визначник матриці.

2.9. Побудувати матрицю, віднімаючи з елементів кожного рядка матриці її середнє арифметичне.

2.10. Знайти максимальний елемент(ти) в матриці і видалити з матриці всі рядки і стовпці, що містять його.

2.11. Ущільнити матрицю, видаляючи з неї рядки і стовпці, заповнені нулями.

2.12. У матриці знайти мінімальний елемент і перемістити його на місце заданного елемента шляхом перестановки рядків і стовпців.

2.13. Перетворити рядки матриці таким чином, щоб елементи, рівні нулю, розташовувалися після всіх інших

2.14. Перевірка парності. Булева матриця має властивість парності, коли кожен рядок і кожен стовпець мають парну суму. Це простий тип коду для виправлення помилок, тому що якщо один біт пошкоджено під

час передачі (біт перевертається від 0 до 1 або від 1 до 0), його можна виявити та виправити. Ось вхідний файл 4 x 4, який має властивість парності:
1 0 1 0
0 0 0 0
1 1 1 1
0 1 0 1
Напишіть програму ParityCheck.java, яка приймає ціле число N як вхідний сигнал командного рядка і зчитує в булевій матриці N-по-N зі стандартного входу та виводить, що:

- матриця має властивість парності, або
- вказує, який єдиний пошкоджений біт (i, j) можна перевернути,щоб відновити властивість парності, або
- вказує, що матриця була пошкоджена (для відновлення властивості парності потрібно змінити більше двох бітів).

Використовуйте якомога менше внутрішньої пам'яті. Підказка: вам навіть не потрібно зберігати матрицю!

## Ініціалізація

3.1. Створіть клас, що містить неініціалізоване посилання на рядок. Продемонструйте, що Java посилання ініціалізує як null.

3.2. Створіть клас із полем String, ініціалізованим у точці визначення, та іншим полем, ініціалізованим конструктором. Чим відрізняються два підходи?

3.3. Створіть клас з конструктором за замовчуванням (без аргументів), що друкує якесь повідомлення. Створіть об'єкт цього класу.

3.4. Додайте перевантажений конструктор до попередньої вправи, яка приймає аргумент String і друкує його разом з вашим повідомленням.

3.5. Створіть клас під назвою Dog, що містить два члени-рядки: name і say. У main () створіть два собачі об'єкти з іменами "spot" (каже: "Ruff!") Та "scruffy" (каже: "Wurf!"). Потім надрукуйте їхні імена та те, що вони говорять. Створіть нове посилання на собаку та призначте її об'єкту "spot". Перевірте порівняння за допомогою == та equals() для всіх посилань.

3.6. Створіть клас під назвою Dog із перевантаженим методом bark(). Цей метод слід перевантажити на основі різних примітивних типів даних і надрукувати різні типи гавкання, виття тощо, залежно від того, яка версія з перевантаженням викликається. Напишіть main(), який викликає всі різні версії.

3.7. Змініть попередню вправу так, щоб два із перевантажених методів мали два аргументи (двох різних типів), але в зворотному порядку один щодо одного. Переконайтеся, що це працює.

3.8. Створіть клас без конструктора, а потім створіть об'єкт цього класу в main (), щоб переконатися, що конструктор за замовчуванням автоматично синтезований.

3.9. Створіть клас двома методами. У межах першого методу двічі викликайте другий метод: перший раз без використання this, а другий раз із використанням this - просто щоб побачити, як він працює; Ви не повинні використовувати цю форму на практиці.

3.10. Створіть клас з двома (перевантаженими) конструкторами. Використовуючи this, викличте другий конструктор всередині першого.

3.11. Створіть клас з методом finalize(), який друкує повідомлення. У main() створіть об'єкт свого класу. Поясніть поведінку вашої програми.

3.12. Змініть попередню вправу так, що ваш finalize() завжди буде викликатися.

3.13. Створіть клас під назвою Tank, який можна заповнити та спорожнити, і має умову завершення, що він повинен бути порожнім під час очищення об'єкта. Напишіть finalize (), який перевіряє цю умову завершення. У main () перевірте можливі сценарії, які можуть статися під час використання вашого Tank.

3.14. Створіть клас із статичним полем String, ініціалізованим у точці визначення, та іншим, ініціалізованим статичним блоком. Додайте статичний метод, який друкує обидва поля і демонструє, що обидва вони ініціалізовані перед їх використанням.

3.15. Створіть клас із рядком, який ініціалізується за допомогою ініціалізації екземпляра.

3.16. Створіть перелік з шести видів валюти. Вкиличить методи value() і для друку кожного значення та його порядкового номеру. Напишіть оператор switch для переліку у попередньому прикладі. Для кожного випадку виведіть опис цієї валюти.

3.17. Створіть клас з конструктором, який приймає аргумент String. Під час побудови надрукуйте аргумент. Створіть масив посилань на об'єкти для цього класу, але насправді не створюйте об'єкти для призначення до масиву. Під час запуску програми зверніть увагу, чи друкуються повідомлення про ініціалізацію з викликів конструктора.

3.18. Виконайте попередню вправу, створивши об'єкти для приєднання до масиву посилань.

3.19. Напишіть метод, який приймає масив рядків довільної кількісті. Переконайтеся, що в цей метод можна передати або розділений комами

список рядків або рядок [].

3.20. Створіть main(), який використовує змінну кількість аргументів замість звичайного синтаксису main (). Роздрукуйте всі елементи в отриманому масиві args. Перевірте це за допомогою різної кількості аргументів командного рядка.