Зміст

ВСТУП. ТРОХИ ПРО КНИГУ ТА МОВУ JAVA
Особливості Java ... 6
Принципи ООП ... 7
Про книгу ... 8
Програмне забезпечення ... 9
Про автора ... 12
Зворотний зв'язок ... 12
Подяки ... 12

РОЗДІЛ 1. ПРОСТІ ПРОГРАМИ
Перша програма ... 13
Ще одна проста програма ... 21
Резюме ... 25

РОЗДІЛ 2. ЗМІННІ ТА ОСНОВНІ ТИПИ ДАНИХ
Змінні базових типів ... 26
Як оголошувати змінні ... 27
Класи-оболонки ... 31
Резюме ... 34

РОЗДІЛ 3. БАЗОВІ ОПЕРАТОРИ ТА ПРИВЕДЕННЯ ТИПІВ
Арифметичні оператори ... 35
Логічні оператори ... 36
Оператори порівняння ... 37
Побітові оператори ... 38
Тернарний оператор ... 40
Оператор присвоєння ... 42
Приведення типів ... 43
Типи літералів ... 44
Комбіновані оператори присвоєння ... 45
Інкремент та декремент ... 46
Пріоритет операцій ... 47
Резюме ... 48

РОЗДІЛ 4. КЕРУЮЧІ ІНСТРУКЦІЇ
Умовний оператор `if` ... 49
Оператор вибору `switch` ... 57
Оператори циклу `while` та `do-while` ... 61
Оператор циклу `for` ... 65
Вибір з результатом ... 70
Резюме ... 71

РОЗДІЛ 5. МАСИВИ
Одновимірні масиви ... 72
Присвоєння масивів ... 77
Двовимірні масиви ... 78

Ініціалізація масиву ... 82
Робота з масивами ... 83
Цикл по колекції ... 88
Резюме ... 89

РОЗДІЛ 6. КЛАСИ ТА ОБ'ЄКТИ

Класи та об'єкти ... 90
Оголошення класу та створення об'єкта ... 91
Методи ... 95
Конструктори ... 100
Перевантаження методів та конструкторів ... 103
Присвоєння об'єктів ... 107
Створення копії об'єкта ... 109
Довільна кількість аргументів ... 113
Резюме ... 115

РОЗДІЛ 7. РОБОТА З ОБ'ЄКТАМИ

Статичні поля та методи ... 116
Об'єкти та методи ... 122
Масиви та об'єкти ... 128
Анонімні об'єкти ... 131
Внутрішні класи ... 134
Аргументи командного рядка ... 137
Резюме ... 141

РОЗДІЛ 8. СПАДКУВАННЯ ТА ІНТЕРФЕЙСИ

Основи спадкування ... 142
Конструктор підкласу ... 145
Перевизначення методів ... 149
Закриті члени класу ... 153
Об'єктні змінні суперкласів ... 156
Абстрактні класи ... 159
Інтерфейси ... 162
Пакети та рівні доступу ... 170
Резюме ... 173

РОЗДІЛ 9. ТЕКСТ ТА ІНШІ УТИЛІТИ

Робота з текстом ... 175
Дата та час ... 185
Математичні утиліти ... 187
Резюме ... 189

РОЗДІЛ 10. ОБРОБКА ВИНЯТКОВИХ СИТУАЦІЙ

Виняткові ситуації та їхні типи ... 190
Обробка винятків ... 192
Створення класів винятків ... 198
Резюме ... 203

РОЗДІЛ 11. БАГАТОПОТОКОВЕ ПРОГРАМУВАННЯ
Реалізація потоків у Java ... 204
Головний потік ... 205
Створення дочірнього потоку ... 208
Синхронізація потоків ... 214
Резюме ... 220
РОЗДІЛ 12. СИСТЕМА ВВЕДЕННЯ/ВИВЕДЕННЯ
Потоки даних та консольне введення ... 221
Форматоване виведення ... 225
Робота з файлами ... 230
Резюме ... 240
РОЗДІЛ 13. ЗНАЙОМСТВО З БІБЛІОТЕКОЮ SWING
Графічний інтерфейс ... 241
Створення простого вікна ... 245
Вікно з текстовою міткою ... 246
Вікно з текстом та піктограмою ... 248
Вікно з міткою та кнопкою ... 251
Класи подій ... 255
Резюме ... 259
РОЗДІЛ 14. ПРОГРАМИ З ГРАФІЧНИМ ІНТЕРФЕЙСОМ
Вікно з полем уведення ... 260
Спадкування класів компонентів ... 277
Резюме ... 291
РОЗДІЛ 15. ЛЯМБДА-ВИРАЗИ
Анонімні класи ... 292
Код за замовчуванням ... 300
Функціональні інтерфейси ... 302
Посилання на методи ... 309
Резюме ... 312
РОЗДІЛ 16. ЗНАЙОМСТВО З КОЛЕКЦІЯМИ
Узагальнені класи та методи ... 313
Загальні відомості про колекції ... 317
Робота зі списками ... 319
Робота з множинами ... 321
Робота зі словниками ... 323
Знайомство з ітераторами ... 325
Резюме ... 327

ПІСЛЯМОВА. ПРО ПЛАНИ І ПЕРСПЕКТИВИ ... 329

Вступ

Трохи про книгу та мову Java

Читачу пропонується книга, присвячена мові програмування Java. Головне завдання цієї книги – навчити програмувати у Java і зробити процес навчання максимально простим. Відверто кажучи, це досить складно. Але, як відомо, задач, що не вирішуються, немає. Як би там не було, у книзі зроблено спробу спростити все, наскільки це взагалі можливо, не знижуючи при цьому фактичний рівень подання матеріалу.

Особливості Java

Вище ми згадали мову програмування Java. Насправді Java – це набагато більше, ніж просто мова програмування. Це технологія.

Історія Java бере початок з 1991 року. Саме цього року компанія Sun Microsystems (згодом її поглинула компанія Oracle) розпочала розробку засобів програмної підтримки електронних компонентів побутових приладів. Важливим етапом у цьому проєкті стала розробка спеціальної мови програмування. Вона мала бути простою, зручною та універсальною. У результаті 1995 року з'явилась мова програмування Java (і супутні технології). Ці події вдало співпали з бурхливим розвитком Internet, а мова Java виявилася зручною для використання у всесвітній мережі.

Що означає універсальність мови Java та за рахунок чого вона підтримується? Ключову роль тут відіграє *віртуальна машина*. Саме завдяки використанню віртуальної машини Java-програми є універсальними.

До уваги

Універсальність у цьому випадку означає, що при написанні програми для нас не важливий тип операційної системи та апаратного забезпечення комп'ютера, на якому виконуватиметься програма. Простіше кажучи, ми можемо писати той самий код для будь-якої операційної системи та будь-якого процесора. Це зручно, особливо якщо програма використовується в мережі та заздалегідь не відомо, яка операційна система встановлена на комп'ютері кінцевого користувача.

Для розуміння ситуації необхідно врахувати, що програма, написана мовою програмування (наприклад, Java), зазвичай перед виконанням перетворюється на зрозумілий для комп'ютера *машинний код*. Цю важливу роботу виконують спеціальні програми, які називаються *компіляторами*. Якщо ми використовуємо різні комп'ютери (з різними операційними системами), то код для кожного комп'ютера різний. Але компілятор для Java-програми переводить програмний код не в машинний, а в так званий *проміжний код* або *байт-код*. Він спільний для всіх типів комп'ютерів. Особливості того чи іншого комп'ютера враховує

віртуальна Java-машина. Це програма, яка попередньо встановлюється на комп'ютер, а потім за її допомогою виконується байт-код.

До уваги

Уявіть, що відбувається екскурсія, а туристи – з різних країн і розмовляють різними мовами. Замість того, щоб перекладати на кожну мову, помічник екскурсовода перекладає англійською. Якщо всі туристи знають англійську, то проблему буде вирішено. Роль "учителя" англійської для "туристів" відіграє віртуальна машина. Тобто ми спочатку "навчаємо" комп'ютер англійської мови (встановлюємо на комп'ютер віртуальну машину), а потім уже "спілкуємося" з цим комп'ютером англійською (аналог байт-коду).

Ще одна особливість мови Java пов'язана з тим, що ця мова повністю *об'єктно-орієнтована*. Це означає, що при написанні програм нам доведеться використовувати *класи* та *об'єкти*. Тому з класами, об'єктами, інтерфейсами, пакетами, методами, полями, потоками – список досить великий, але без цього не обійтися.

Деталі

Є декілька платформ Java. Нас цікавитиме стандартна платформа Java Standard Edition (скорочено Java SE).

Принципи ООП

Об'єктно-орієнтоване програмування (скорочено ООП) – це така парадигма програмування. Простіше кажучи, це деякі принципи, яким підпорядковується створення програм. ООП є альтернативою парадигмі *структурного програмування*. В останньому випадку програма є послідовністю інструкцій, які виконуються одна за одною і можуть групуватися в підпрограми (функції та процедури). У рамках об'єктно-орієнтованого підходу (який, як зазначалося, використовується у Java) програма реалізується як набір взаємодіючих об'єктів. Об'єкти створюються на основі класів. Клас, своєю чергою, описує певний "тип", що поєднує як дані, так і програмний код для обробки цих даних. Фактично клас є шаблоном, на основі якого створюються об'єкти.

До уваги

Концепція класів та об'єктів є наріжною в ООП. Якщо зараз вона не надто зрозуміла – не страшно. Після розгляду прикладів усе стане на свої місця.

Парадигма ООП виникла як реакція на збільшення обсягів програм. Фактично йдеться про зміну способу структурування програми. В ООП дані та код для їхньої обробки поєднуються на рівні класів та об'єктів, за рахунок чого зменшується ймовірність некоректного використання даних.

Деталі

Парадигма ООП базується на трьох принципах: *інкапсуляції*, *поліморфізмі* та *спадкуванні*.

● Інкапсуляція – механізм об'єднання даних та коду для їхньої обробки в одному об'єкті. Взаємодія користувача з об'єктом визначається методами об'єкта.
● Поліморфізм – механізм підтримки різної реалізації для об'єктів з однаковою специфікацією/інтерфейсом. У Java поліморфізм реалізується через перевантаження та перевизначення методів.
● Спадкування дозволяє створювати нові класи на основі вже існуючих. Новий клас автоматично отримує (успадковує) властивості вихідного (батьківського) класу (але при цьому також має свої власні). У Java батьківський клас називається *суперкласом*, а створений на його основі клас – *підкласом*. Через спадкування формується ієрархія пов'язаних класів. У вершині цієї ієрархії – класи з найуніверсальнішими властивостями, а від них за ланцюжком спадкування розташовуються специфічніші класи.

Про книгу

Як зазначалося вище, Java є повністю об'єктно-орієнтованою мовою. З перших програм нам доведеться описувати класи. Для новачків у програмуванні це може бути складно. Тому на початковому етапі ми не зосереджуватимемося на деталях щодо створення класу. Натомість будемо використовувати певні шаблони, сенс яких стане зрозумілим згодом.

Основна частина книги складається з шістнадцяти розділів. Кожен розділ закінчується коротким *резюме*.

● У *розділі 1* подано декілька простих програм. Там є інструкції щодо використання *середовища розробки*.
● У *розділі 2* розглядаються змінні та базові типи Java.
● У *розділі 3* ми познайомимося з арифметичними, логічними, побітовими операторами та операторами порівняння.
● У *розділі 4* розглядаються умовні оператори та оператори циклу.
● У *розділі 5* ми познайомимося з масивами.
● У *розділі 6* ми розпочнемо знайомство з класами та об'єктами.
● У *розділі 7* ми продовжимо знайомитися з принципами ООП.
● У *розділі 8* обговорюються інтерфейси та спадкування.
● У *розділі 9* описуються методи роботи з текстом.
● У *розділі 10* розглядається обробка виняткових ситуацій.
● У *розділі 11* вивчається багатопотокове програмування.
● У *розділі 12* ми ближче познайомимося з системою введення/виведення.
● У *розділі 13* зібрано базову інформацію про бібліотеку Swing, яка використовується при створенні програм з графічним інтерфейсом.
● У *розділі 14* ми продовжимо знайомство з методами створення програм з графічним та інтерфейсом.
● У *розділі 15* обговорюються лямбда-вирази, функціональні інтерфейси та посилання на методи.
● У *розділі 16* подано коротку інформацію про узагальнені класи, методи та колекції.

До уваги

Запропонований план дій насправді досить умовний. З деякими утилітами ми познайомимося відразу. Наприклад, використання класів з бібліотеки Swing розпочнеться вже у першому розділі під час створення діалогових вікон.

Програмне забезпечення

Мало вміти написати програму. Важливо знати, що з нею робити. Для цього знадобиться спеціальне програмне забезпечення. Його можна встановити абсолютно безкоштовно. Це тішить.

Що ж нам потрібно? Нам потрібен редактор, у якому набиратимемо програмний код. Також нам знадобиться компілятор та віртуальна машина. На перший погляд список гнітючий, але насправді все досить просто і зводиться до встановлення системи розробки (JDK від Java Development Kit) та інтегрованого середовища розробки (IDE від Integrated Development Environment). Система JDK – це, власне, компілятор та віртуальна машина, а IDE – це редактор і купа різних корисних підпрограм, які роблять процес програмування простим та приємним.

Деталі

Спочатку встановлюється JDK, а потім IDE. Можна і навпаки, але тоді частину налаштувань доведеться виконати вручну.

Систему JDK шукаємо на сайті `www.oracle.com` компанії Oracle (за розробку та підтримку Java "відповідає" саме ця компанія). На рис. В.1 показано сайт компанії.

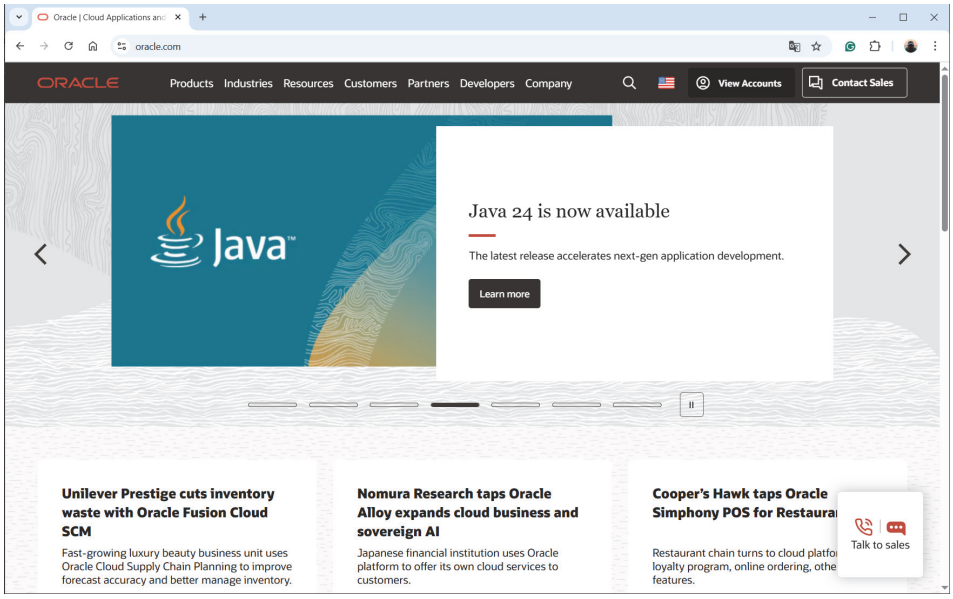

Рис. В.1. Сайт `www.oracle.com` компанії Oracle

Там необхідно знайти розділ завантажень і вибрати потрібний файл для завантаження, як показано на рис. В.2.

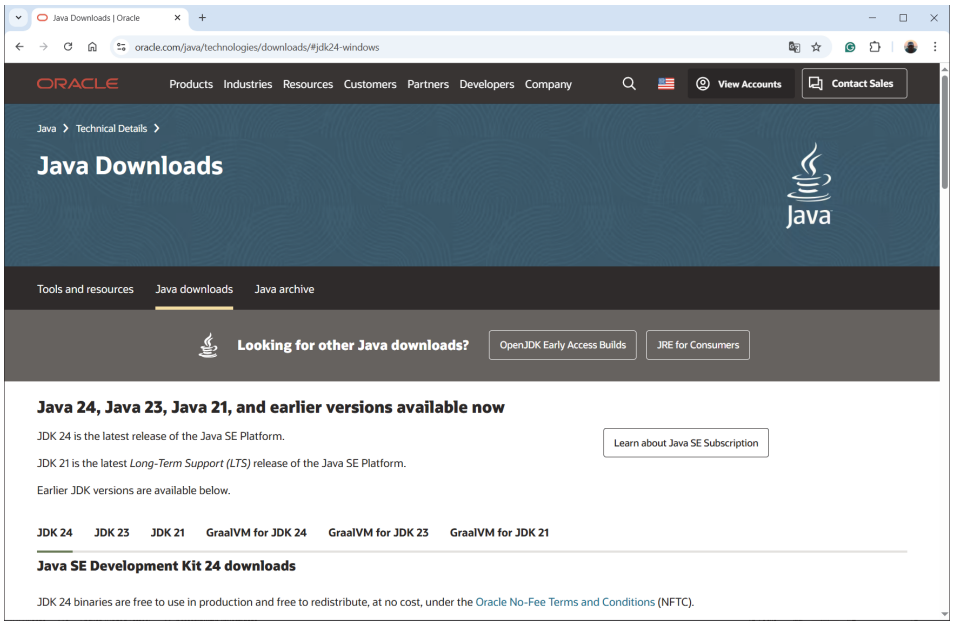

Рис. B.2. Завантаження системи JDK

Програмне забезпечення необхідно завантажити та встановити. У процесі встановлення є сенс погоджуватися з тими варіантами, які пропонуються за замовчуванням.

Після встановлення JDK переходимо до встановлення інтегрованого середовища розробки. Тут є різні варіанти – у тому сенсі, що існує досить багато безкоштовних або комерційних застосунків, які надають широкий спектр можливостей для створення Java-програм. Ми скористаємося середовищем розробки IntelliJ IDEA від компанії JetBrains. Вікно браузера з відкритим сайтом `www.jetbrains.com` показано на рис. B.3.

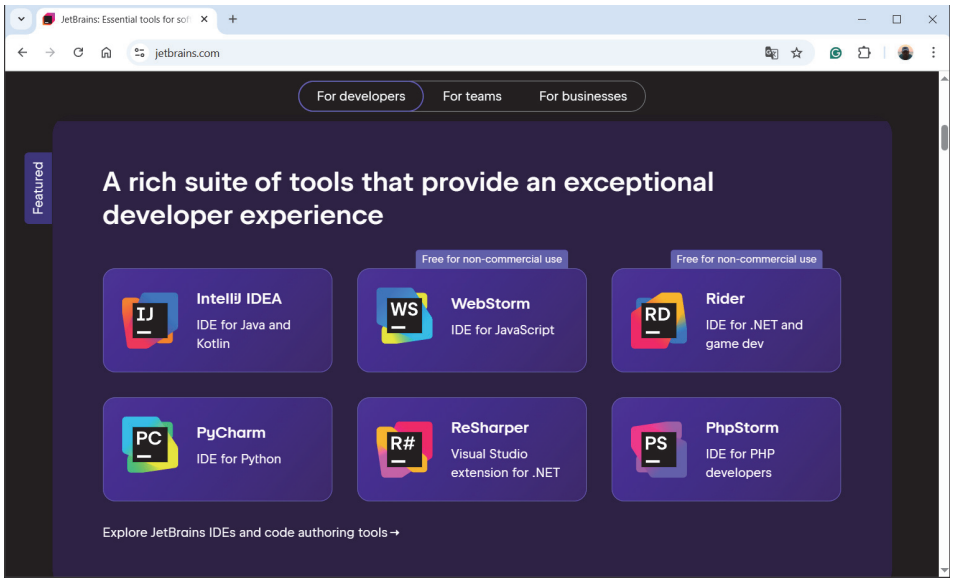

Рис. B.3. Сайт `www.jetbrains.com` компанії JetBrains

На цьому сайті потрібно знайти розділ завантажень, завантажити інсталяційні файли для середовища розробки IntelliJ IDEA і встановити цей застосунок. Якщо заздалегідь встановлено систему JDK, то всі налаштування виконаються автоматично. На рис. В.4 показано початкове вікно середовища розробки IntelliJ IDEA.

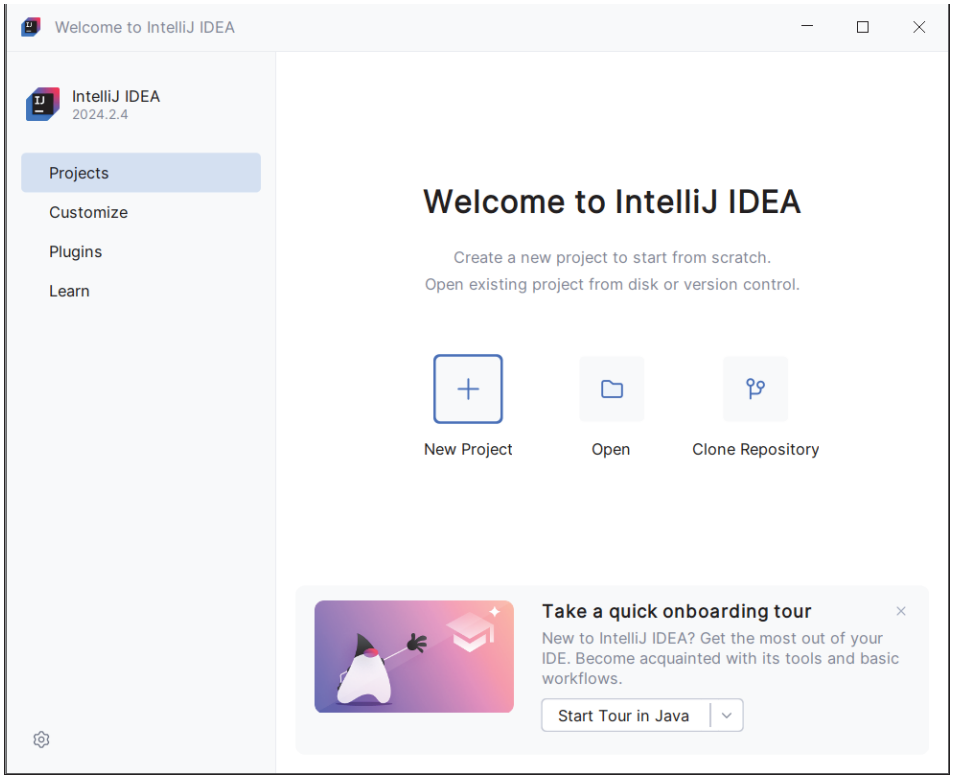

Рис. В.4. Початкове вікно середовища розробки IntelliJ IDEA

Для створення нового проєкту вибирається відповідна команда у вікні.

До уваги

Зовнішній вигляд вікна середовища розробки IntelliJ IDEA може відрізнятись від показаного на рисунках. Ми можемо задати параметри зовнішнього вигляду, для чого використовується спеціальна піктограма в нижній частині початкового вікна. Аналогічна команда доступна через головне меню програми.

Також слід врахувати, що у залежності від налаштувань середовища розробки та його версії при створенні нового проєкту може пропонуватись код за замовчуванням (шаблонний код, який автоматично додається до проєкту). Призначення інструкцій Java та методи роботи з проєктами ми розглянемо в основній частині книги.

Детальніше методи роботи з інтегрованим середовищем розробки розглядаються у *розділі 1* під час обговорення перших програм.

Про автора

Автор книги – *Васильєв Олексій Миколайович*, доктор фізико-математичних наук, професор кафедри програмних систем і технологій факультету інформаційних технологій Київського національного університету імені Тараса Шевченка. Автор книг з програмування та математичного моделювання. Сфера наукових інтересів: квантитативна лінгвістика, дослідження біофізичних систем, математична економіка та моделювання соціально-політичних процесів.

Зворотний зв'язок

Висловити свої зауваження та пропозиції стосовно цієї або інших книг автора можна за електронною поштою `oleksii@vasyliev.kyiv.ua`. Автор заздалегідь вдячний своїм читачам за конструктивну критику. Інформацію про вже видані книги, а також деякі корисні матеріали, що стосуються цих книг (наприклад, програмні коди прикладів), можна знайти на сайті `www.vasyliev.kyiv.ua` та на YouTube-каналі `www.youtube.com/@oleksii.vasyliev`.

Подяки

Автор висловлює щиру подяку своїм читачам, студентам та слухачам курсів. Найкращим стимулом у роботі є усвідомлення того, що ця робота комусь потрібна.

Розділ 1

Прості програми

Знайомство з мовою програмування Java почнемо з простих прикладів. Найпростіша програма – та, що виводить повідомлення. Куди виводити повідомлення – питання окреме. Можна відображати повідомлення у консолі, а можна відображати повідомлення в окремому діалоговому вікні. Ми розглянемо обидва випадки. Почнемо з діалогового вікна.

Перша програма

Одночасно з розглядом коду програми, простежимо весь процес створення програми в середовищі IntelliJ IDEA. Наші дії такі.

- Запускаємо середовище розробки IntelliJ IDEA.
- У початковому вікні середовища вибираємо піктограму **New Project**. Якщо у середовищі вже відкрито проєкт, новий можна створити, вибравши команду **New** в меню **File**. У результаті відкриється вікно **New Project**, показане на рис. 1.1.

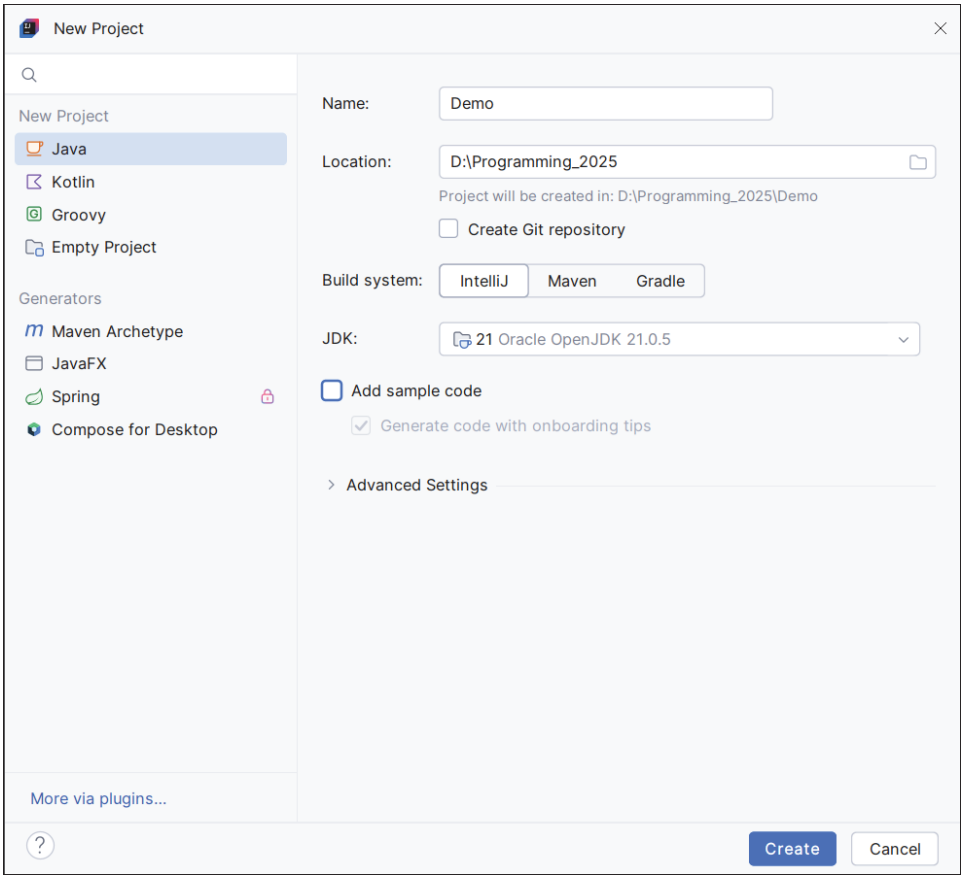

Рис. 1.1. Створення нового проєкту в IntelliJ IDEA

У цьому вікні у лівій частині вибираємо позицію **Java**. При цьому у динамічному списку **JDK** вибираємо ту платформу Java, на основі якої створюється програма користувача.