KHATWA TECH · PROGRAMMING NOTES

Java Fundamentals

أساسيات Java من الصفر حتى Encapsulation

مراجعة مركزة تساعدك تفهم الفكرة، تراجع الكود، وتختبر نفسك بأمثلة ونتائج فعلية.

المعيار المستهدف	المستوى	الإصدار
Java 25 LTS	مبتدئ – متوسط	2026

إعداد
محمود منصور
Mahmoud Mansour
23 نوفمبر 2026

JAVA 25 LTS · KHATWA TECH

خريطة المذكرة

الفهرس وطريقة استخدام المراجعة

القسم	الصفحة
Introduction · 01	03
Input & Output · 02	04
Variables & Data Types · 03	06–05
Operators & Conditions · 04	08–07
Loops · 05	10–09
Arrays · 06	13–11
Strings · 07	15–14
Methods · 08	18–16
OOP Fundamentals · 09	21–19
ArrayList · 10	22
Exceptions & Validation · 11	25–23
المشروع · Student Grades Manager	27–26
الأخطاء الشائعة	28
ورقة المراجعة السريعة	29
اختبر نفسك	31–30
الإجابات والحالات الحدّية	34–32
المراجع والخطوة التالية	35

تستخدم المذكرة إزاي؟

1. اقرأ عنوان القسم والفرق الأساسي، وجرّب تفهم كل مثال قبل ما تبص للناتج.
2. شغّل الأكواد بنفسك وغيّر قيمة واحدة كل مرة، وبعدها قارن النتيجة.
3. حلّ أسئلة الصفحتين 30–31 من غير مساعدة، وراجع تفسير الإجابات بعد كده.

طريقة التشغيل احفظ كل مثال كامل في ملف باسم `Main.java` ثم شغّل `javac Main.java` وبعدها `java Main`.

الأمثلة تستخدم الصيغة التقليدية للفئة Main لأنها أوضح للمبتدئين، رغم وجود صيغ أبسط في الإصدارات الحديثة.

JAVA 25 LTS · KHATWA TECH

Introduction · 01

شكل البرنامج وطريقة تشغيل Java

القسم Introduction · 01

Java لغة strongly typed؛ الكود بيتجمع إلى bytecode، وبعدها JVM بتشغّله.

المصطلح	الوظيفة
JDK	أدوات التطوير ومنها `javac` و `java`
JVM	بيشغّل الـbytecode
JRE	بيئة تشغيل؛ وده مفهوم مهم حتى لو تثبيته منفصل مش لازم للمبتدئ

أول برنامج — Java

```java
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello, Java!");
    }
}
```

الناتج (Output)

```
Hello, Java!
```

تقرأ الكود إزاي؟

- `class Main` بتعرّف Class اسمها Main.
- `main` نقطة بدء البرنامج في المثال.
- `System.out.println` بتطبع سطر وبعده newline.
- `//` تعليق سطر، و `/* ... */` تعليق متعدد الأسطر.

انتبه اسم الملف لازم يكون `Main.java` لما يكون عندك `public class Main` .

JAVA 25 LTS · KHATWA TECH

Input & Output · 02

الطباعة وقراءة بيانات المستخدم

القسم 02 · I/O

استخدم println للسطر الكامل، وprint من غير newline، وScanner لقراءة المدخلات.

print وprintln وprintf

Java

```
public class Main {
    public static void main(String[] args) {
        System.out.print("Course: ");
        System.out.println("Java");
        System.out.printf(java.util.Locale.US, "Price: %.2f%n", 149.5);
    }
}
```

الناتج (Output)

```
Course: Java
Price: 149.50
```

قراءة اسم ودرجة

Java

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String name = in.nextLine();
        int grade = in.nextInt();
        System.out.println(name + ": " + grade);
    }
}
```

الإدخال (Input)

```
Mona
90
```

الناتج (Output)

```
Mona: 90
```

الأداة	استخدام سريع
`nextInt()`	عدد صحيح
`nextDouble()`	عدد عشري
`nextLine()`	سطر كامل

خطأ مشهور بعد `nextInt()` أو `nextDouble()` ممكن تفضل newline في الإدخال؛ عالجها قبل `nextLine()` .

JAVA 25 LTS · KHATWA TECH

Variables & Data Types · 03

الأنواع الأساسية وإعلان المتغيرات

القسم 03 · Data Types

اختيار النوع الصح بيحدد القيم اللي تقدر تخزنها والعمليات المتاحة.

النوع	مثال	المعنى
`int`	`int age = 20;`	عدد صحيح
`double`	`double price = 99.5;`	عدد عشري
`char`	`char grade = 'A';`	حرف UTF-16 code unit
`boolean`	`boolean passed = true;`	صح / غلط
`String`	`String name = "Ali";`	نص؛ Reference Type
`final`	`final int MAX = 100;`	متغير ماينفعش تعيد تعيينه

أمثلة على أنواع مختلفة

Java

```java
public class Main {
    public static void main(String[] args) {
        int students = 4;
        double price = 125.5;
        boolean available = true;
        String course = "Java";
        System.out.println(course + ": " + students);
        System.out.println(price);
        System.out.println(available);
    }
}
```

الناتج (Output)

```
Java: 4
125.5
true
```

الفرق اللي يهمك `String` مش primitive؛ هو Class، لكن `int` و `double` و `boolean` أنواع primitive.

JAVA 25 LTS · KHATWA TECH

Casting & Constants · 03

التحويل بين الأنواع وfinal

> التحويل من نوع لنوع ممكن يكون تلقائي (widening) أو يحتاج Cast صريح (narrowing).

Explicit Castو Widening — Java

```java
public class Main {
    public static void main(String[] args) {
        int grade = 87;
        double wide = grade;
        double price = 19.99;
        int whole = (int) price;
        final int passMark = 50;
        System.out.println(wide);
        System.out.println(whole);
        System.out.println(passMark);
    }
}
```

الناتج (Output)

```
87.0
19
50
```

التحويل بين Stringو int — Java

```java
public class Main {
    public static void main(String[] args) {
        String value = "120";
        int amount = Integer.parseInt(value);
        String label = String.valueOf(amount + 30);
        System.out.println(label);
    }
}
```

الناتج (Output)

```
150
```

التحويل	الطريقة
`int → double`	تلقائي عادة
`double → int`	Cast صريح، ويشيل الجزء الكسري
`String → int`	`Integer.parseInt(text)`
`int → String`	`String.valueOf(number)`

انتبه `Integer.parseInt("12x")` بيرمي `NumberFormatException` لأن النص مش عدد صحيح صالح.

JAVA 25 LTS · KHATWA TECH

Operators · 04

حساب ومقارنة ومنطق

القسم Operators · 04

فهم أولوية العمليات والفرق بين قسمة int وdouble مهم عشان الناتج يطلع صح.

Arithmetic Operators — Java

```java
public class Main {
    public static void main(String[] args) {
        int a = 7, b = 2;
        System.out.println(a + b);
        System.out.println(a / b);
        System.out.println(a % b);
        System.out.println((double) a / b);
    }
}
```

الناتج (Output)

```
9
3
1
3.5
```

Comparison & Logical — Java

```java
public class Main {
    public static void main(String[] args) {
        int grade = 65;
        boolean passed = grade >= 50;
        boolean excellent = grade >= 90;
        System.out.println(passed && !excellent);
        System.out.println(grade == 65);
    }
}
```

الناتج (Output)

```
true
true
```

الرموز	الاستخدام
`+ - * / %`	حساب
`== != > < >= <=`	مقارنة
`&& || !`	AND / OR / NOT
`+= -= ++ --`	تعديل قيمة

خطأ مشهور `7 / 2` الناتج `3` لأن الرقمين `int` ؛ استخدم `7.0 / 2` لو عايز `3.5`.

JAVA 25 LTS · KHATWA TECH

Conditions · 04

if / else والشرط المختصر

استخدم if لما القرار بيعتمد على شرط؛ ولو فيه أكتر من احتمال استخدم else if.

تقييم درجة الطالب — Java

```java
public class Main {
    public static void main(String[] args) {
        int grade = 82;
        if (grade >= 90) {
            System.out.println("Excellent");
        } else if (grade >= 50) {
            System.out.println("Passed");
        } else {
            System.out.println("Failed");
        }
    }
}
```

الناتج (Output)

```
Passed
```

Ternary Operator — Java

```java
public class Main {
    public static void main(String[] args) {
        int price = 180;
        String tag = price >= 200 ? "Expensive" : "Affordable";
        System.out.println(tag);
    }
}
```

الناتج (Output)

```
Affordable
```

جرّب بنفسك غيّر `grade` إلى `49` ثم إلى `90` وتوقّع أي فرع هيتنفذ.

الفرق اللي يهمك `if / else` أوضح مع أكتر من خطوة؛ `?:` مناسب لاختيار قيمة واحدة.

JAVA 25 LTS · KHATWA TECH

Switch · 04

اختيار بين حالات معروفة

> switch مناسب لما بتقارن قيمة بحالات ثابتة؛ switch expression بتقلل التكرار في Java الحديثة.

Switch Expression — Java

```java
public class Main {
    public static void main(String[] args) {
        int day = 2;
        String name = switch (day) {
            case 1 -> "Saturday";
            case 2 -> "Sunday";
            case 3 -> "Monday";
            default -> "Unknown";
        };
        System.out.println(name);
    }
}
```

الناتج (Output)

```
Sunday
```

Switch Statement — Java

```java
public class Main {
    public static void main(String[] args) {
        String level = "basic";
        switch (level) {
            case "basic" -> System.out.println("Start here");
            case "advanced" -> System.out.println("Next level");
            default -> System.out.println("Invalid");
        }
    }
}
```

الناتج (Output)

```
Start here
```

الأسلوب	متى أستخدمه؟
`if / else`	Ranges وشروط متنوعة
`switch`	حالات منفصلة لنفس القيمة
`switch expression`	عايز ترجع قيمة من الاختيار

انتبه صيغة `case ... ->` مش محتاجة `break` ، على عكس الصيغة التقليدية اللي ممكن يحصل فيها fall-through.

JAVA 25 LTS · KHATWA TECH

Loops · 05

for وwhile

القسم Loops · 05

Loops بتكرر نفس الأوامر من غير ما تكتبها كل مرة. اختار Loop حسب طريقة تحديد النهاية.

for — عدد مرات معروف

Java

```
public class Main {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            System.out.println("Lesson " + i);
        }
    }
}
```

الناتج (Output)

```
Lesson 1
Lesson 2
Lesson 3
```

while — لحد ما الشرط يتغير

Java

```
public class Main {
    public static void main(String[] args) {
        int balance = 3;
        while (balance > 0) {
            System.out.println(balance);
            balance--;
        }
    }
}
```

الناتج (Output)

```
3
2
1
```

خطأ مشهور لو الشرط مش بيتغير جوه `while` ، البرنامج ممكن يدخل في Infinite Loop.

JAVA 25 LTS · KHATWA TECH

Loop Control · 05

break/continue و Nested Loops و do-while

> do-while بتنفّذ الجسم مرة على الأقل، وbreak وcontinue بيغيروا سير التكرار.

Java — **do-while**

```java
public class Main {
    public static void main(String[] args) {
        int tries = 0;
        do {
            tries++;
            System.out.println("Try " + tries);
        } while (tries < 2);
    }
}
```

الناتج (Output)

```
Try 1
Try 2
```

Java — **continue و break**

```java
public class Main {
    public static void main(String[] args) {
        for (int i = 1; i <= 5; i++) {
            if (i == 2) continue;
            if (i == 4) break;
            System.out.println(i);
        }
    }
}
```

الناتج (Output)

```
1
3
```

Java — **Nested Loops**

```java
public class Main {
    public static void main(String[] args) {
        for (int row = 1; row <= 2; row++) {
            for (int col = 1; col <= 3; col++) {
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```

الناتج (Output)

```
***
***
```

JAVA 25 LTS · KHATWA TECH

Arrays · 06

إنشاء Array وقراءة العناصر

القسم Arrays · 06

Array مجموعة قيم من نفس النوع وطولها ثابت؛ أول عنصر رقمه 0.

Index وlength وتعديل عنصر — Java

```java
public class Main {
    public static void main(String[] args) {
        int[] grades = {70, 85, 90};
        System.out.println(grades[0]);
        System.out.println(grades.length);
        grades[1] = 88;
        System.out.println(grades[1]);
    }
}
```

الناتج (Output)

```
70
3
88
```

Enhanced for — Java

```java
public class Main {
    public static void main(String[] args) {
        int[] grades = {10, 20, 30};
        int sum = 0;
        for (int grade : grades) {
            sum += grade;
        }
        System.out.println(sum);
    }
}
```

الناتج (Output)

```
60
```

انتبه آخر Index دايمًا `length - 1` ؛ الوصول إلى `grades[3]` هنا بيرمي `ArrayIndexOutOfBoundsException` .

JAVA 25 LTS · KHATWA TECH

2D Arrays · 06

جداول الصفوف والأعمدة

> 2D Array عبارة عن Array جواها Arrays، وممكن أطوال الصفوف تختلف.

جدول درجات — Java

```java
public class Main {
    public static void main(String[] args) {
        int[][] scores = {{80, 90}, {70, 60}};
        System.out.println(scores[0][1]);
        int total = 0;
        for (int[] row : scores) {
            for (int value : row) total += value;
        }
        System.out.println(total);
    }
}
```

الناتج (Output)

```
90
300
```

أهم الفرق بين الأنواع

الهيكل	الفكرة	الوصول
`int[]`	قائمة واحدة	`a[i]`
`int[][]`	صفوف وأعمدة	`a[r][c]`
`int[][] ragged`	صفوف بأطوال مختلفة	`a[r].length`

> **جرّب بنفسك** غيّر الصف الثاني إلى `{70}` ، وبعدها اطبع طول كل صف باستخدام `scores[0].length` و `scores[1].length` .

JAVA 25 LTS · KHATWA TECH

Arrays Utilities · 06

فرز وبحث وأرقام أعلى/أقل

المكتبة `java.util.Arrays` فيها أدوات جاهزة لفرز Arrays وطباعتها.

Java — **sort وbinarySearch**

```java
import java.util.Arrays;
public class Main {
    public static void main(String[] args) {
        int[] scores = {90, 70, 85};
        Arrays.sort(scores);
        System.out.println(Arrays.toString(scores));
        System.out.println(Arrays.binarySearch(scores, 85));
    }
}
```

الناتج (Output)

```
[70, 85, 90]
1
```

Java — **Min / Max**

```java
public class Main {
    public static void main(String[] args) {
        int[] marks = {60, 95, 40, 80};
        int min = marks[0], max = marks[0];
        for (int mark : marks) {
            if (mark < min) min = mark;
            if (mark > max) max = mark;
        }
        System.out.println("Min=" + min + ", Max=" + max);
    }
}
```

الناتج (Output)

```
Min=40, Max=95
```

انتبه `Arrays.binarySearch` يتطلب Array مرتّبة مسبقًا. وحساب Min/Max باستخدام `arr[0]` يتطلب Array مش فاضية.

JAVA 25 LTS · KHATWA TECH

Strings · 07

الطول والبحث والمقارنة

القسم Strings · 07

String Object مش Array، وله methods جاهزة؛ والمقارنة بالمحتوى عن طريق equals.

أشهر String Methods — Java

```java
public class Main {
    public static void main(String[] args) {
        String course = "Java Basics";
        System.out.println(course.length());
        System.out.println(course.charAt(0));
        System.out.println(course.substring(5));
        System.out.println(course.contains("Basic"));
    }
}
```

الناتج (Output)

```
11
J
Basics
true
```

equals مقابل == — Java

```java
public class Main {
    public static void main(String[] args) {
        String input = new String("Java");
        String course = "Java";
        System.out.println(input.equals(course));
        System.out.println(input == course);
    }
}
```

الناتج (Output)

```
true
false
```

الفرق اللي يهمك `equals()` بيقارن محتوى String؛ `==` بيقارن الـreferences لما تتعامل مع Objects.

JAVA 25 LTS · KHATWA TECH

StringBuilder & Split · 07

بناء النصوص وتقسيمها

> String غير قابل للتعديل (immutable). عند تجميع أجزاء كتير، StringBuilder بيكون أنسب.

StringBuilder — Java

```java
public class Main {
    public static void main(String[] args) {
        StringBuilder report = new StringBuilder("Ali");
        report.append(": ").append(92);
        System.out.println(report);
    }
}
```

الناتج (Output)

```
Ali: 92
```

split وtrim — Java

```java
public class Main {
    public static void main(String[] args) {
        String data = "HTML,CSS,Java";
        String[] courses = data.split(",");
        System.out.println(courses.length);
        System.out.println(courses[2]);
        System.out.println(" Java ".trim());
    }
}
```

الناتج (Output)

```
3
Java
Java
```

Method	المهمة
`length()`	عدد char units في النص
`toUpperCase()`	أحرف كبيرة
`strip()`	إزالة whitespace من الأطراف
`replace(a, b)`	استبدال جزء نصي
`isBlank()`	فحص لو النص كله whitespace

انتبه `split()` بيستخدم Regex؛ لو الفاصل حرف خاص زي النقطة، اكتب `split("\\.")` في كود Java.

JAVA 25 LTS · KHATWA TECH

Methods · 08

تعريف method وقيمة return

القسم 08 · Methods

Method بتجمع خطوات محددة تحت اسم واحد عشان تعيد استخدامها بسهولة.

Method بترجع قيمة — Java

```java
public class Main {
    static int total(int a, int b) {
        return a + b;
    }
    public static void main(String[] args) {
        System.out.println(total(30, 20));
    }
}
```

الناتج (Output)

```
50
```

void Method — Java

```java
public class Main {
    static void greet(String name) {
        System.out.println("Hello, " + name);
    }
    public static void main(String[] args) {
        greet("Mona");
    }
}
```

الناتج (Output)

```
Hello, Mona
```

الجزء	معناه
`static`	تقدر تناديها من main من غير Object
`int`	نوع القيمة الراجعة
`return`	ترجع نتيجة وتنهي method
`void`	مفيش قيمة راجعة

JAVA 25 LTS · KHATWA TECH

Overloading & Scope · 08

أكثر من توقيع وحدود رؤية المتغير

Overloading يعني Methods بنفس الاسم لكن Parameters مختلفة في العدد أو النوع.

Java — **Method Overloading**

```java
public class Main {
    static int discount(int price, int amount) {
        return price - amount;
    }
    static double discount(double price, double rate) {
        return price * (1 - rate);
    }
    public static void main(String[] args) {
        System.out.println(discount(100, 20));
        System.out.println(discount(100.0, 0.25));
    }
}
```

الناتج (Output)

```
80
75.0
```

Java — **Pass by Value و Scope**

```java
public class Main {
    static int doubleIt(int number) {
        number *= 2;
        return number;
    }
    public static void main(String[] args) {
        int n = 5;
        System.out.println(doubleIt(n));
        System.out.println(n);
    }
}
```

الناتج (Output)

```
10
5
```

الفرق اللي يهمك Java بتستخدم Pass by Value دائمًا؛ مع Object القيمة المنسوخة هي الـreference، مش Object نفسه.

JAVA 25 LTS · KHATWA TECH

Recursion · 08

Method تنادي نفسها بشرط توقف

> Recursion مناسب لحل مشكلة بتتصغر كل مرة؛ لازم Base Case تمنع الاستدعاءات اللامتناهية.

حساب Factorial — Java

```java
public class Main {
    static int factorial(int n) {
        if (n <= 1) return 1;
        return n * factorial(n - 1);
    }
    public static void main(String[] args) {
        System.out.println(factorial(4));
    }
}
```

الناتج (Output)

```
24
```

مجموع من 1 إلى n — Java

```java
public class Main {
    static int sumTo(int n) {
        if (n <= 0) return 0;
        return n + sumTo(n - 1);
    }
    public static void main(String[] args) {
        System.out.println(sumTo(5));
    }
}
```

الناتج (Output)

```
15
```

انتبه لو مفيش Base Case أو الحجم مش بيقل، ممكن يحصل `StackOverflowError` .

جرّب بنفسك ارسم استدعاءات `factorial(3)` على ورقة، وبعدين راجع الضرب أثناء الرجوع.

JAVA 25 LTS · KHATWA TECH

Classes & Objects · 09

الفئة كقالب والـObject كنسخة

القسم 09 · OOP

Class بتجمع بيانات وسلوك؛ والـObject نسخة ليها قيم فعلية.

Class وObject

Java

```
class Student {
    String name;
    int grade;
    void printReport() {
        System.out.println(name + ": " + grade);
    }
}
public class Main {
    public static void main(String[] args) {
        Student student = new Student();
        student.name = "Ali";
        student.grade = 88;
        student.printReport();
    }
}
```

الناتج (Output)

```
Ali: 88
```

المصطلح	المعنى
`class`	قالب لتعريف النوع
`object`	نسخة من class
`field`	بيانات داخل class
`method`	سلوك جوه class
`new`	إنشاء Object

الفرق اللي يهمك Class تعريف؛ Object بيانات فعلية موجودة أثناء التشغيل. مش كل Class لازم تكون `public` .

JAVA 25 LTS · KHATWA TECH

Constructors & this · 09

تهيئة الـObject من أول لحظة

Constructor بيتنادى وقت `new` وبيستقبل بيانات إنشاء الـObject.

Constructor وthis — Java

```
class Course {
    String name;
    int price;
    Course(String name, int price) {
        this.name = name;
        this.price = price;
    }
    void show() {
        System.out.println(name + " costs " + price);
    }
}
public class Main {
    public static void main(String[] args) {
        Course c = new Course("Java", 250);
        c.show();
    }
}
```

الناتج (Output)

```
Java costs 250
```

انتبه Constructor ملوش نوع إرجاع، واسمه مطابق لاسم الـClass. `this.name` يشير للـfield وليس parameter.

JAVA 25 LTS · KHATWA TECH

Encapsulation · 09

حماية الحالة والتحكم في التعديل

خلّي الحقول `private` ، واعمل Methods واضحة تتأكد من القيم قبل تعديلها.

Java — **Getter/Setter وPrivate Fields**

```java
class Student {
    private int grade;
    Student(int grade) { setGrade(grade); }
    public void setGrade(int grade) {
        if (grade < 0 || grade > 100)
            throw new IllegalArgumentException("Bad grade");
        this.grade = grade;
    }
    public int getGrade() { return grade; }
}
public class Main {
    public static void main(String[] args) {
        Student s = new Student(90);
        System.out.println(s.getGrade());
    }
}
```

الناتج (Output)

```
90
```

الفرق اللي يهمك Encapsulation مش مجرد Getter وSetter لكل field؛ الأهم إن الـClass تمنع الحالة غير الصحيحة.

جرّب بنفسك بدّل `new Student(90)` إلى `new Student(120)` وشوف ليه التحقق مهم.

JAVA 25 LTS · KHATWA TECH

ArrayList · 10

قائمة قابلة لزيادة وتقليل العناصر

القسم 10 · ArrayList

بعكس Array، ArrayList بتزيد أو تقل وقت التشغيل، وبتستخدم Objects زي Integer.

عمليات ArrayList — **Java**

```java
import java.util.ArrayList;
public class Main {
    public static void main(String[] args) {
        ArrayList<String> courses = new ArrayList<>();
        courses.add("Java");
        courses.add("C++");
        courses.set(1, "Python");
        System.out.println(courses.get(0));
        System.out.println(courses.size());
        courses.remove("Python");
        System.out.println(courses);
    }
}
```

الناتج (Output)

```
Java
2
[Java]
```

Array	ArrayList
field `length`	method `size()`
حجم ثابت	حجم متغير
ينفع primitives	تخزن references / wrapper types
`a[i]`	`list.get(i)`

انتبه `ArrayList<Integer>` مش `ArrayList<int>`؛ Generics تحتاج reference types.

JAVA 25 LTS · KHATWA TECH

Exceptions · 11

التعامل مع الأخطاء أثناء التشغيل

القسم 11 · Exceptions

Exception تقدر تتعامل معها عند نقطة مناسبة بدل توقف البرنامج بالكامل.

try / catch / finally — **Java**

```java
public class Main {
    public static void main(String[] args) {
        try {
            int grade = Integer.parseInt("abc");
            System.out.println(grade);
        } catch (NumberFormatException ex) {
            System.out.println("Invalid number");
        } finally {
            System.out.println("Finished");
        }
    }
}
```

الناتج (Output)

```
Invalid number
Finished
```

النوع	السبب المشهور
`NumberFormatException`	نص مش رقم
`ArrayIndexOutOfBoundsException`	Index برّه Array
`NullPointerException`	استخدام reference قيمته null
`IllegalArgumentException`	قيمة argument غير مقبولة

انتبه متستخدمش `catch (Exception e)` لإخفاء كل الأخطاء من غير سبب؛ عالج الاستثناء المتوقع تحديدًا.

JAVA 25 LTS · KHATWA TECH

Input Validation · 11

حماية البرنامج من إدخال غير صالح

اقرأ قيمة مدخلة كنص، وجرّب تحويلها، وبعد كده افحص حدودها؛ ده بيسهّل التعامل مع الأخطاء.

تحويل آمن + فحص range — Java

```java
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String text = in.nextLine();
        try {
            int grade = Integer.parseInt(text.trim());
            if (grade < 0 || grade > 100) {
                System.out.println("Out of range");
            } else {
                System.out.println("Grade: " + grade);
            }
        } catch (NumberFormatException ex) {
            System.out.println("Invalid input");
        }
    }
}
```

الإدخال (Input)

```
85
```

الناتج (Output)

```
Grade: 85
```

Input	Output
`85`	`Grade: 85`
`200`	`Out of range`
`hello`	`Invalid input`
`-1`	`Out of range`

جرّب بنفسك اختبر القيم `0` و `100` و `-1` ، وتأكد من نتيجة كل حالة.

JAVA 25 LTS · KHATWA TECH

المشروع · Student Grades Manager

البرنامج الكامل — Java Console

الهدف: إدخال بيانات الطلاب والتحقق منها، ثم استخراج الحالة والمتوسط وأقل وأعلى درجة.

خطوات التشغيل احفظ الكود كما هو في `Main.java` ؛ شغّل البرنامج وقدّم عدد الطلاب ثم اسم ودرجة لكل طالب.

الكود الكامل للمشروع — Java

```java
import java.util.ArrayList;
import java.util.Scanner;

class Student {
    private final String name;
    private final int grade;
    Student(String name, int grade) {
        if (name.isBlank() || grade < 0 || grade > 100)
            throw new IllegalArgumentException("Invalid student");
        this.name = name;
        this.grade = grade;
    }
    String getName() { return name; }
    int getGrade() { return grade; }
    String getStatus() { return grade >= 50 ? "Passed" : "Failed"; }
}

public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        ArrayList<Student> students = new ArrayList<>();
        int count;
        try {
            count = Integer.parseInt(in.nextLine().trim());
            if (count < 1 || count > 20) throw new IllegalArgumentException();
            for (int i = 0; i < count; i++) {
                String name = in.nextLine().trim();
                int grade = Integer.parseInt(in.nextLine().trim());
                students.add(new Student(name, grade));
            }
        } catch (IllegalArgumentException | java.util.NoSuchElementException ex) {
            System.out.println("Invalid input");
            return;
        }
        int total = 0, min = 101, max = -1;
        for (Student s : students) {
            int grade = s.getGrade();
            total += grade;
            min = Math.min(min, grade);
            max = Math.max(max, grade);
            System.out.println(s.getName() + ": " + s.getStatus());
        }
        System.out.printf(java.util.Locale.US,
            "Average: %.2f%nMin: %d%nMax: %d%n",
            (double) total / students.size(), min, max);
    }
}
```

JAVA 25 LTS · KHATWA TECH

المشروع · النتائج والاختبارات

قراءة الناتج وتطوير الكود

عند إدخال البيانات التالية، البرنامج يطبع حالة كل طالب ثم المتوسط وأقل وأعلى درجة.

الإدخال (Input)

```
3
Ali
90
Mona
45
Omar
75
```

الناتج الحقيقي (Output)

```
Ali: Passed
Mona: Failed
Omar: Passed
Average: 70.00
Min: 45
Max: 90
```

السطر	التفسير
`Ali: Passed`	درجة Ali تساوي 90، أكبر أو تساوي 50
`Mona: Failed`	درجة Mona تساوي 45، أقل من 50
`Omar: Passed`	درجة Omar تساوي 75
`Average: 70.00`	المجموع 210 على 3 طلاب
`Min: 45`	أقل درجة
`Max: 90`	أعلى درجة

3 أفكار تطور بيها المشروع

1. اعمل عرض لكل الطلاب مرتّب من أعلى درجة لأقل درجة.
2. أضف البحث عن طالب بالاسم، وإظهار نتيجته لو موجود.
3. احسب عدد الناجحين والراسبين وأظهر نسبتهم.

اختبار مهم لو العدد `0` أو `21` أو الدرجة برّه `0..100`، الناتج يكون `Invalid input`.

JAVA 25 LTS · KHATWA TECH

الأخطاء الشائعة

رسالة الخطأ — السبب — تعمل إيه

> رسائل Java غالبًا بتحدد نوع المشكلة. اعرف هل دي Compile-time ولا Runtime ولا Logic.

رسالة الخطأ	السبب المحتمل	الحل
`cannot find symbol`	اسم variable أو Method غير معروف	راجع الاسم وScope وimport
`';' expected`	ناسي ;	راجع نهاية الجملة
`class Main is public, should be declared...`	اسم الملف مش مطابق	سمّي الملف Main.java
`incompatible types`	نوع غير متوافق	راجع النوع أو Cast المناسب
`NumberFormatException`	نص مش رقم صالح	استخدم try/catch وفحص Input
`ArrayIndexOutOfBoundsException`	Index خارج الحدود	راجع index مقابل length
`NullPointerException`	Object reference قيمته null	تأكد من الإنشاء أو تحقق من null
`ArithmeticException: / by zero`	قسمة int على 0	افحص المقسوم عليه قبل القسمة
`InputMismatchException`	Scanner استقبل نوع مختلف	افحص hasNext أو اقرأ نص وحوّله
`StackOverflowError`	Recursion من غير توقف صحيح	أضف Base Case
ناتج غير متوقع مع `==`	مقارنة Object references	استعمل equals للمحتوى
متوسط غير دقيق	قسمة int صحيحة	اعمل Cast لـdouble قبل القسمة

انتبه Runtime Error مش هو نفسه Compiler Error. Logic Error ممكن البرنامج يشتغل عادي لكن يطلع نتيجة غلط.

JAVA 25 LTS · KHATWA TECH

ورقة مراجعة سريعة

عايز أعمل كذا ← الصيغة المختصرة

عايز أعمل كذا	الصيغة المختصرة
طباعة سطر	`System.out.println(value);`
طباعة بدون سطر	`System.out.print(value);`
طباعة برقمين عشريين	`System.out.printf("%.2f%n", value);`
قراءة سطر	`String x = in.nextLine();`
قراءة int من نص	`int x = Integer.parseInt(text);`
تعريف رقم ثابت	`final int LIMIT = 100;`
شرط	`if (grade >= 50) { ... }`
اختيار مختصر	`String s = ok ? "Yes" : "No";`
for loop	`for (int i = 0; i < n; i++) { ... }`
for-each	`for (int x : values) { ... }`
طول Array	`arr.length`
طول String	`text.length()`
مقارنة String	`a.equals(b)`
بحث داخل String	`text.contains("Java")`
تقسيم String	`text.split(",")`
تعريف Method	`static int add(int a, int b) { return a+b; }`
إنشاء Object	`Student s = new Student();`
إنشاء ArrayList	`ArrayList<String> list = new ArrayList<>();`
إضافة إلى List	`list.add("Java");`
قراءة عنصر List	`list.get(0)`
حماية Field	`private int grade;`
التعامل مع Exception	`try { ... } catch (NumberFormatException e) { ... }`

ملاحظة الصيغ اللي فيها `...` مجرد قوالب سريعة، مش برامج كاملة للتشغيل.

JAVA 25 LTS · KHATWA TECH

اختبر نفسك · توقع الناتج

6 أسئلة من غير ما تشغّل الكود أولًا

كل كود قصير هنا بيتنفّذ داخل `main()`. اكتب الناتج بالترتيب قبل ما تفتح الحلول في صفحة 32.

س1 · Integer Division

```
int a = 9, b = 2;
System.out.println(a / b);
System.out.println(a % b);
```

الناتج: .....

س2 · Conditions

```
int grade = 49;
System.out.println(grade >= 50 ? "Passed" : "Failed");
```

الناتج: .....

س3 · Loops

```
int sum = 0;
for (int i = 1; i <= 3; i++) sum += i;
System.out.println(sum);
```

الناتج: .....

س4 · Arrays

```
int[] a = {4, 7, 9};
System.out.println(a[1] + a.length);
```

الناتج: .....

س5 · Strings

```
String s = "Java";
System.out.println(s.substring(1, 3));
```

الناتج: .....

س6 · References

```
int[] a = {3};
int[] b = a;
b[0] = 8;
System.out.println(a[0]);
```

الناتج: .....

JAVA 25 LTS · KHATWA TECH

اختبر نفسك · اكتب الحل

6 تمارين مع Input → Output

اكتب كل حل كاملًا في ملف `Main.java` ، واقرأ Input من الـConsole. الحلول الكاملة في الصفحات 33–34.

ت1 · ناجح أم راسب؟

اقرأ درجة واحدة؛ اطبع Passed لو 50 أو أكتر، وإلا Failed.

Input
```
85
```

Output
```
Passed
```

ت2 · مجموع الأرقام

اقرأ عددًا n؛ اطبع مجموع الأعداد من 1 إلى n (لـ n ≥ 0).

Input
```
5
```

Output
```
15
```

ت3 · أكبر قيمة

اقرأ n ثم n درجات؛ اطبع أكبر درجة؛ لو n = 0 اطبع Empty.

Input
```
4
60 95 70 88
```

Output
```
95
```

ت4 · عدد الكلمات

اقرأ سطرًا؛ اطبع عدد الكلمات بعد trim، ولو فاضي اطبع 0.

Input
```
Java is easy
```

Output
```
3
```

ت5 · فلترة الدرجات

اقرأ 4 درجات؛ اطبع عدد الدرجات اللي >= 50.

Input
```
30 60 50 99
```

Output
```
3
```

ت6 · Class + Validation

اعمل Student بـ private grade وتحقق من 0..100. اطبع Passed لو grade >=50؛ لو خارج الحدود اطبع Invalid.

Input
```
120
```

Output
```
Invalid
```

JAVA 25 LTS · KHATWA TECH

إجابات توقع الناتج

تفسير الأسئلة الستة

النواتج دي اتراجعت بالتشغيل؛ راجع سبب كل ناتج مش مجرد الرقم.

س1 · Integer Division

الناتج

```
4
1
```

القسمة الصحيحة بتشيل الجزء الكسري، و% بيرجع الباقي.

س2 · Conditions

الناتج

```
Failed
```

49 أصغر من 50، فالشرط false.

س3 · Loops

الناتج

```
6
```

1 + 2 + 3 = 6.

س4 · Arrays

الناتج

```
10
```

العنصر عند index 1 هو 7، والطول 3.

س5 · Strings

الناتج

```
av
```

substring تبدأ من 1 وتتوقف قبل 3.

س6 · References

الناتج

```
8
```

المتغيرين بيشيروا لنفس الـArray.

طريقة حل الأسئلة ابدأ بنوع المتغير، ثم ترتيب التنفيذ، ثم حالة الذاكرة أو شروط المقارنة.

حالات حدّية · الجزء الأول

الاختبار	Input	Output
ت3: قائمة فاضية	`0`	`Empty`
ت4: سطر فاضي		`0`
ت6: أقل درجة	`0`	`Failed`

JAVA 25 LTS · KHATWA TECH

حلول التمارين · 1 إلى 3

أكواد كاملة والاختبارات الحدّية

ت1 · Passed / Failed — Java

```java
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int grade = in.nextInt();
        System.out.println(grade >= 50 ? "Passed" : "Failed");
    }
}
```

الإدخال (Input)

```
85
```

الناتج (Output)

```
Passed
```

ت2 · Sum from 1 to n — Java

```java
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt(), sum = 0;
        for (int i = 1; i <= n; i++) sum += i;
        System.out.println(sum);
    }
}
```

الإدخال (Input)

```
5
```

الناتج (Output)

```
15
```

ت3 · Maximum with empty case — Java

```java
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        if (n == 0) { System.out.println("Empty"); return; }
        int max = Integer.MIN_VALUE;
        for (int i = 0; i < n; i++)
            max = Math.max(max, in.nextInt());
        System.out.println(max);
    }
}
```

الإدخال (Input)

```
4
60 95 70 88
```

الناتج (Output)

```
95
```

حالات حدّية · الجزء الثاني

الاختبار	Input	Output
ت6: الحد الفاصل	`50`	`Passed`
ت6: أعلى درجة	`100`	`Passed`
ت6: فوق الحد	`101`	`Invalid`

JAVA 25 LTS · KHATWA TECH

حلول التمارين · 4 إلى 6

حلول كاملة للمسائل 4–6

ت4 · Word Count — Java

```java
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String text = in.nextLine().trim();
        System.out.println(text.isEmpty() ? 0 : text.split("\\s+").length);
    }
}
```

الإدخال (Input)

```
Java is easy
```

الناتج (Output)

```
3
```

ت5 · Count Passed — Java

```java
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int passed = 0;
        for (int i = 0; i < 4; i++)
            if (in.nextInt() >= 50) passed++;
        System.out.println(passed);
    }
}
```

الإدخال (Input)

```
30 60 50 99
```

الناتج (Output)

```
3
```

ت6 · Encapsulated Student — Java

```java
import java.util.Scanner;
class Student {
    private final int grade;
    Student(int grade) {
        if (grade < 0 || grade > 100)
            throw new IllegalArgumentException();
        this.grade = grade;
    }
    String status() { return grade >= 50 ? "Passed" : "Failed"; }
}
public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int grade = in.nextInt();
        try { System.out.println(new Student(grade).status()); }
        catch (IllegalArgumentException e) { System.out.println("Invalid"); }
    }
}
```

الإدخال (Input)

```
120
```

الناتج (Output)

```
Invalid
```

JAVA 25 LTS · KHATWA TECH

المراجع والخطوة التالية

توثيق رسمي ومهمة تدريبية بعد المراجعة

> كل الروابط التالية من توثيق Oracle الرسمي للنسخة المستهدفة Java 25.

JDK 25 Documentation	https://docs.oracle.com/en/java/javase/25/
Java Language Specification 25	https://docs.oracle.com/javase/specs/jls/se25/html/
Java 25 API Docs	https://docs.oracle.com/en/java/javase/25/docs/api/index.html
System / PrintStream	https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/io/PrintStream.html
Scanner	https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/util/Scanner.html
String / StringBuilder	https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/lang/String.html
Arrays	https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/util/Arrays.html
ArrayList	https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/util/ArrayList.html
Exceptions / Throwable	https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/lang/Throwable.html
Java Language Updates	https://docs.oracle.com/en/java/javase/25/language/index.html

بعد ما تخلص المذكرة

نفّذ مشروع `Course Registration Manager` : خزّن أسماء الكورسات والأسعار في `ArrayList` ، واعمل `Course class` بـ `private fields` و `constructor` و `validation` ، ثم اعرض إجمالي السعر.

قائمة تحقق قبل ما تقول إنك خلصت

- بتفهم الفرق بين `Array` و `ArrayList` .
- بتكتب Methods وتفهم `return` و `scope` .
- بتعرف تعمل Class وConstructor وتحمي البيانات باستخدام Encapsulation.
- بتفحص Input وتتعامل مع Exceptions المتوقعة.

> **ملاحظة حول التوافق** المحتوى مستهدف Java 25 LTS، والأمثلة تستخدم ميزات مستقرة متوافقة مع Java 21 وما بعدها؛ لم نستخدم Preview Features.