

דף עזר ל-Java

לא צריך לזכור את התחביר. צריך לדעת איפה לחפש אותו.

שלד של תוכנית

כל תוכנית מתחילה כך. את הקוד שלכם כותבים בתוך main. שורות import נכתבות בראש הקובץ, מעל השורה public class.

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        // הקוד שלכם כאן
    }
}
```

הערות לא מורצות. הן נועדו לבני אדם. הערה על כמה שורות נפתחת ב-/* ונסגרת ב-*/.

```
// הערה בשורה אחת

/* הערה על
   כמה שורות */
```

הדפסה למסך

```
System.out.println("Hello"); // מדפיס ויורד שורה
System.out.print("Hello");   // מדפיס ונשאר באותה שורה
System.out.println("x = " + x); // טקסט + משתנה
System.out.println();         // שורה ריקה
```

הדפסה מעוצבת: d% מספר שלם, f% עשרוני, 2f% עשרוני עם 2 ספרות, s% טקסט, c% תו, n% ירידת שורה.

```
System.out.printf("%d + %d = %d\n", 2, 3, 5);
System.out.printf("Price: %.2f\n", 9.5); // Price: 9.50
System.out.printf("Name: %s\n", "Dana");
```

תווים מיוחדים בתוך טקסט.

```
System.out.println("Line1\nLine2"); // \n ירידת שורה
System.out.println("A\tB");         // \t טאב
System.out.println("She said \"Hi\""); // \" גרשים בתוך טקסט
```

קלט מהמשתמש

שלב 1: ייבוא ויצירת Scanner

הייבוא נכתב בראש הקובץ. את שורת ה-Scanner כותבים פעם אחת, בתחילת main.

```
import java.util.Scanner; // בראש הקובץ

Scanner sc = new Scanner(System.in); // main בתחילת
```

שלב 2: קליטת מספר

קודם מדפיסים הודעה למשתמש, ואז קולטים.

```
System.out.print("Enter a number: ");
int num = sc.nextInt(); // מספר שלם

double price = sc.nextDouble(); // מספר עשרוני
```

קליטת טקסט ותווים

```
String word = sc.next();           // מילה אחת (עד הרווח הראשון)
String line = sc.nextLine();        // שורה שלמה, כולל רווחים
boolean flag = sc.nextBoolean();    // true / false
char c = sc.next().charAt(0);       // תו בודד
```

זהירות: `nextLine` אחרי `nextInt`

אחרי `nextInt` נשארת ירידת שורה בקלט, ו-`nextLine` הבא יחזיר טקסט ריק. הפתרון: שורת `nextLine` מיותרת באמצע.

```
int age = sc.nextInt();
sc.nextLine();                     // אוכל" את ירידת השורה שנשארה"
String name = sc.nextLine();       // עכשיו זה עובד
```

קליטה חוזרת

```
// קולטים 5 מספרים
for (int i = 0; i < 5; i++) {
    int n = sc.nextInt();
}

// קולטים עד שהשתמש מקליד 0
int x = sc.nextInt();
while (x != 0) {
    x = sc.nextInt();
}

sc.close();                       // בסוף התוכנית
```

משתנים וטיפוסים

מבנה: טיפוס שם = ערך;

```
int age = 16;                      // מספר שלם
double price = 9.99;               // מספר עשרוני
boolean ok = true;                 // true או false
char grade = 'A';                  // תו בודד, בגרש בודד
String name = "Dana";              // טקסט, בגרשיים
final int MAX = 100;               // קבוע: אי אפשר לשנות
```

שינוי ערך של משתנה קיים (בלי לכתוב שוב את הטיפוס).

```
age = 17;
age = age + 1;
age++;                             // age = age + 1
age--;                             // age = age - 1
total += 5;                         // total = total + 5
total -= 2; total *= 3; total /= 2;
```

חישובים ו-Math

חילוק של שני מספרים שלמים נותן מספר שלם. כדי לקבל עשרוני, לפחות אחד מהם צריך להיות עשרוני.

```
int sum = 7 + 2;                    // 9      חיבור
int diff = 7 - 2;                   // 5      חיסור
int prod = 7 * 2;                   // 14     כפל
int q = 7 / 2;                      // 3      חילוק שלמים
double d = 7 / 2.0;                 // 3.5    חילוק עשרוני
int r = 7 % 2;                      // 1      שארית
boolean even = n % 2 == 0;          // זוגי n
boolean div5 = n % 5 == 0;          // מתחלק ב-5 n
int lastDigit = n % 10;              // ספרת האחדות
int withoutLast = n / 10;           // בלי ספרת האחדות
```

המרת טיפוסים (casting)

```
int x = (int) 3.9; // 3 (חותך, לא מעגל)  
double avg = (double) sum / count; // ממוצע מדויק
```

פעולות Math

המחלקה Math לא דורשת import. היא זמינה תמיד, כמו String ו-System.

```
int m = Math.abs(-5); // 5 ערך מוחלט  
double p = Math.pow(2, 3); // 8.0 חזקה  
double s = Math.sqrt(16); // 4.0 שורש  
int mx = Math.max(a, b); // הגדול  
int mn = Math.min(a, b); // הקטן  
long rd = Math.round(2.5); // 3 עיגול  
double fl = Math.floor(2.9); // 2.0 עיגול למטה  
double ce = Math.ceil(2.1); // 3.0 עיגול למעלה
```

מספרים אקראיים

nextInt(n) מגריל מ-0 עד n-1. אם צריך טווח אחר, מוסיפים. הייבוא נכתב בראש הקובץ.

```
import java.util.Random; // בראש הקובץ  
  
Random rnd = new Random();  
int n = rnd.nextInt(10); // 0..9  
int dice = rnd.nextInt(6) + 1; // 1..6  
int x = rnd.nextInt(max - min + 1) + min; // min..max
```

דרך נוספת, בלי import.

```
int n = (int) (Math.random() * 10); // 0..9  
int dice = (int) (Math.random() * 6) + 1; // 1..6
```

תנאים

```
if (score >= 90) {  
    System.out.println("Excellent");  
} else if (score >= 60) {  
    System.out.println("Pass");  
} else {  
    System.out.println("Fail");  
}
```

אופרטורים

השוואה: == (שווה), != (שונה), <, >, <=, >= (חיבור תנאים: && (וגם), || (או)), ! (לא).

```
if (age >= 18 && age <= 120) { }  
if (x < 0 || x > 100) { }  
if (!(x == 5)) { } // x != 5  
  
// 0 < x < 10 טווח: לא כותבים  
if (x > 0 && x < 10) { }
```

```
switch (day) {
    case 1:
        System.out.println("Sunday");
        break;
    case 2:
        System.out.println("Monday");
        break;
    default:
        System.out.println("Other");
}
```

השוואת ערכים

מספרים ותווים משווים עם ==. טקסט (String) משווים תמיד עם equals, לא עם ==.

```
if (a == b) { } // מספרים
if (c1 == 'a') { } // תווים

if (s1.equals(s2)) { } // טקסט
if (s1.equalsIgnoreCase(s2)) { } // בלי הבדל בין גדולות לקטנות
if (!s1.equals(s2)) { } // טקסט שונה

if (s1.compareTo(s2) < 0) { } // s1 בסדר אלפביתי s2 לפני
```

לולאות

for: כשיוודעים כמה פעמים

```
for (int i = 0; i < 5; i++) { // i = 0,1,2,3,4
    System.out.println(i);
}

for (int i = 1; i <= 10; i++) { } // 1..10
for (int i = 10; i >= 1; i--) { } // 10..1
for (int i = 0; i < 20; i += 2) { } // קפיצות של 2
```

while: כל עוד התנאי נכון

```
while (n > 1) {
    n = n / 2;
}
```

do-while: לפחות פעם אחת

```
int num;
do {
    System.out.print("Enter 1-10: ");
    num = sc.nextInt();
} while (num < 1 || num > 10);
```

break ו-continue: ולולאה בתוך לולאה

```
for (int i = 0; i < 10; i++) {
    if (i == 3) continue; // מדלג לסיבוב הבא
    if (i == 7) break; // יוצא מהלולאה
}

for (int i = 0; i < 3; i++) { // שורות
    for (int j = 0; j < 4; j++) { // עמודות
        System.out.print("*");
    }
    System.out.println();
}
```

```

int sum = 0; // סכום
for (int i = 1; i <= 10; i++) {
    sum += i;
}

int count = 0; // ספירה
for (int i = 1; i <= 100; i++) {
    if (i % 3 == 0) count++;
}

int max = Integer.MIN_VALUE; // מקסימום
for (int i = 0; i < 5; i++) {
    int n = sc.nextInt();
    if (n > max) max = n;
}

```

טקסט (String)

האינדקס הראשון הוא 0. תו אחרון נמצא באינדקס 1 - length().

```

String s = "Hello";
int len = s.length(); // 5
char c = s.charAt(0); // 'H'
char last = s.charAt(s.length() - 1); // 'o'
String part = s.substring(1, 3); // "el" (מ-1 עד 3, לא כולל)
String rest = s.substring(2); // "llo" (מ-2 עד הסוף)
int pos = s.indexOf("l"); // 2 (אם לא נמצא -1)
boolean has = s.contains("ell"); // true
String up = s.toUpperCase(); // "HELLO"
String low = s.toLowerCase(); // "hello"
String rep = s.replace("l", "L"); // "HeLLo"
String tr = " hi ".trim(); // "hi"
boolean empty = s.isEmpty(); // false
String[] parts = "a,b,c".split(","); // {"a","b","c"}
String joined = s + " " + "World"; // חיבור טקסטים

```

מעבר על כל התווים

```

for (int i = 0; i < s.length(); i++) {
    char c = s.charAt(i);
    System.out.println(c);
}

```

בדיקות והמרות

```

int n = Integer.parseInt("42"); // טקסט למספר שלם
double d = Double.parseDouble("3.5"); // טקסט לעשרוני
String s = String.valueOf(42); // מספר לטקסט

boolean dig = Character.isDigit(c); // ספרה?
boolean let = Character.isLetter(c); // אות?
char up = Character.toUpperCase(c);
int digitValue = '7' - '0'; // 7

```

מערכים

```
int[] arr = new int[5];           // 5 0 תאים, כולם
int[] nums = {4, 8, 15, 16};      // מערך עם ערכים
String[] names = new String[3];   // תאים ריקים (null)

arr[0] = 10;                      // כתיבה (אינדקס מ-0)
int first = arr[0];               // קריאה
int len = arr.length;            // אורך (בלי סוגריים!)
int last = arr[arr.length - 1];   // התא האחרון
```

מעבר על מערך

```
for (int i = 0; i < arr.length; i++) {
    System.out.println(arr[i]);
}

for (int x : arr) {                // בלי אינדקס
    System.out.println(x);
}
```

כלים מוכנים (import java.util.Arrays)

```
System.out.println(Arrays.toString(arr)); // [4, 8, 15]
Arrays.sort(arr);                        // מיון
Arrays.fill(arr, 0);                     // ממלא הכל ב-0
```

מערך דו-ממדי

```
int[][] mat = new int[3][4];           // שורות, 4 עמודות 3
mat[1][2] = 5;                         // שורה 1, עמודה 2
int rows = mat.length;
int cols = mat[0].length;

for (int i = 0; i < mat.length; i++) {
    for (int j = 0; j < mat[i].length; j++) {
        System.out.print(mat[i][j] + " ");
    }
    System.out.println();
}
```

רשימה (ArrayList)

דורש import java.util.ArrayList; בראש הקובץ. בין הסימנים <> כותבים Integer, Double, String ולא int או double.

```
ArrayList<Integer> list = new ArrayList<>();
ArrayList<String> names = new ArrayList<>();

list.add(5);                          // מוסיף בסוף
list.add(0, 7);                        // מוסיף באינדקס 0
int x = list.get(0);                   // קריאה
list.set(1, 9);                        // החלפת ערך באינדקס 1
list.remove(0);                        // מוחק לפי אינדקס
list.remove(Integer.valueOf(5));       // מוחק לפי ערך
int size = list.size();                // length! גודל
boolean has = list.contains(9);
int pos = list.indexOf(9);             // אם לא נמצא -1
boolean empty = list.isEmpty();
list.clear();                          // מרוקן
```

```
for (int i = 0; i < list.size(); i++) {
    System.out.println(list.get(i));
}

for (int v : list) {
    System.out.println(v);
}
```

מחסנית (Stack)

מחסנית עובדת כמו ערימת צלחות: מוסיפים ומוציאים רק מלמעלה. הייבוא נכתב בראש הקובץ.

```
import java.util.Stack; // בראש הקובץ

Stack<Integer> st = new Stack<>();
Stack<String> names = new Stack<>();

st.push(5); // מוסיף לראש המחסנית
st.push(8);
int top = st.peek(); // מסתכל על הראש (8), לא מוציא
int x = st.pop(); // מוציא את הראש ומחזיר אותו (8)
boolean empty = st.isEmpty();
int size = st.size();
```

זהירות: מחסנית ריקה

pop או peek על מחסנית ריקה גורמים לשגיאה (EmptyStackException). בודקים isEmpty לפני.

```
if (!st.isEmpty()) {
    int x = st.pop();
}
```

מעבר על מחסנית עד שהיא מתרוקנת

```
while (!st.isEmpty()) {
    int x = st.pop(); // יוצאים בסדר הפוך מסדר הכניסה
    System.out.println(x);
}
```

תור (Queue)

תור עובד כמו תור בחנות: נכנסים מהסוף ויוצאים מההתחלה. את התור יוצרים עם LinkedList. הייבוא נכתב בראש הקובץ.

```
import java.util.LinkedList; // בראש הקובץ
import java.util.Queue;

Queue<Integer> q = new LinkedList<>();
Queue<String> names = new LinkedList<>();

q.add(5); // מוסיף לסוף התור
q.add(8);
int first = q.peek(); // מסתכל על הראשון (5), לא מוציא
int x = q.remove(); // מוציא את הראשון ומחזיר אותו (5)
boolean empty = q.isEmpty();
int size = q.size();
```

זהירות: תור ריק

remove על תור ריק גורם לשגיאה. אפשר לבדוק isEmpty לפני, או להשתמש ב-poll שמחזיר null כשהתור ריק (וגם peek).

```
if (!q.isEmpty()) {
    int x = q.remove();
}

Integer y = q.poll();          // אם התור ריק null
```

מעבר על תור עד שהוא מתרוקן

```
while (!q.isEmpty()) {
    int x = q.remove();          // יוצאים בסדר הכניסה
    System.out.println(x);
}
```

פעולות (methods)

הפעולות נכתבות בתוך המחלקה, מחוץ ל-main.

```
// לא מחזירה ערך
public static void greet(String name) {
    System.out.println("Hello " + name);
}

// void מחזירה ערך: כותבים את הטיפוס במקום
public static int add(int a, int b) {
    return a + b;
}

public static boolean isEven(int n) {
    return n % 2 == 0;
}

// מקבלת מערך
public static int sum(int[] arr) {
    int total = 0;
    for (int x : arr) {
        total += x;
    }
    return total;
}

public static void main(String[] args) {
    greet("Dana");                // קריאה לפעולה
    int s = add(2, 3);            // שומרים את מה שחזר
    if (isEven(s)) { }
    int t = sum(new int[]{1, 2, 3});
}
```

מחלקות ואובייקטים

מחלקה נכתבת בקובץ משלה, עם אותו שם כמו המחלקה (Student.java).

```
public class Student {
    private String name;           // תכונות
    private int grade;

    public Student(String name, int grade) {    // בנאי
        this.name = name;
        this.grade = grade;
    }

    public String getName() {        // get
        return name;
    }

    public void setGrade(int grade) {    // set
        this.grade = grade;
    }

    public String toString() {        // איך מדפיסים אובייקט
        return name + ": " + grade;
    }
}
```

שימוש במחלקה (בתוך main)

```
Student s = new Student("Dana", 90);    // יצירת אובייקט
s.setGrade(95);                          // קריאה לפעולה של האובייקט
String n = s.getName();
System.out.println(s);                   // toString-משתמש ב
```

static ובדיקת null

משתנה או פעולה static שייכים למחלקה ולא לאובייקט, ולכן קוראים להם דרך שם המחלקה.

```
static int count = 0;                    // משותף לכל האובייקטים

static int twice(int x) { return x * 2; }

public static void main(String[] args) {
    int y = Main.twice(4);                // twice(4) בתוך אותה מחלקה אפשר גם
    String name = null;
    if (name != null) { }                 // בודקים לפני שמשתמשים
}
```