

Laborator 5 - limbajul Java

Structura programelor Java. Tipuri de date. Instructiuni. Citirea/Scrierea datelor formatare cu ajutorul clasei Scanner

1. Fluxuri de date

Majoritatea aplicatiilor necesita citirea unor informatii care se gasesc pe o sursa externa sau trimiterea unor informatii catre o destinatie externa.

Informatia se poate gasi oriunde:

- intr-un fisier pe disc,
- in retea,
- in memorie
- sau in alt program

si poate fi de orice tip: date primitive, obiecte, imagini, sunete, etc.

Pentru a aduce informatii dintr-un mediu extern, un program Java trebuie sa deschida un canal de comunicatie (flux) de la sursa informatiilor (fisier, memorie, socket, etc) si sa citeasca secvential informatiile respective.

Similar, un program poate trimite informatii catre o destinatie externa deschizand un canal de comunicatie (flux) catre acea destinatie si scriind secvential informatiile respective.

Indiferent de tipul informatiilor, citirea/scrierea de pe/catre un mediu extern respecta urmatorul algoritm:

```
deschide canal comunicatie  
while (mai sunt informatii)  
{  
    citeste/scrie informatie;  
}  
inchide canal comunicatie;
```

Pentru a generaliza, atat sursa externa a unor date cat si destinatia lor sunt vazute ca fiind niste procese care produc, respectiv consuma informatii.

Definitii:

Un **flux** este un canal de comunicatie unidirectional intre doua procese.

Un **proces** care descrie o sursa externa de date se numeste proces producator.

Un proces care descrie o destinatie externa pentru date se numeste **proces consumator**.

Un flux care citeste date se numeste **flux de intrare**.

Laborator - Programare orientată pe obiecte Limbajul Java

Adrian Runceanu

Un flux care scrie date se numeste **flux de iesire**.

Observatii:

Fluxurile sunt canale de comunicatie seriale pe 8 sau 16 biti.

Fluxurile sunt unidirectionale, de la producator la consumator. Fiecare flux are un singur proces producator si un singur proces consumator. Intre doua procese pot exista oricate fluxuri, orice proces putand fi atat producator cat si consumator in acelasi timp, dar pe fluxuri diferite. Consumatorul si producatorul nu comunica direct printr-o interfata de flux ci prin intermediul codului Java de tratare a fluxurilor.

Clasele si intefetele standard pentru lucrul cu fluxuri se gasesc in pachetul **java.io**. Deci, orice program care necesita operatii de intrare sau iesire trebuie sa contina instructiunea de import a pachetului **java.io**:

```
import java.io.*;
```

2. Clasificarea fluxurilor

Exista trei tipuri de clasificare a fluxurilor:

- 1) Dupa directia canalului de comunicatie deschis fluxurile se impart in:
 - fluxuri de intrare (pentru citirea datelor)
 - fluxuri de iesire (pentru scrierea datelor)
- 2) Dupa tipul de date pe care opereaza:
 - fluxuri de octeti (comunicarea seriala se realizeaza pe 8 biti)
 - fluxuri de caractere (comunicarea seriala se realizeaza pe 16 biti)
- 3) Dupa actiunea lor:
 - fluxuri primare de citire/scriere a datelor (se ocupa efectiv cu citirea/scrierea datelor)
 - fluxuri pentru procesarea datelor

3. Intrari si iesiri formate

Incepand cu versiunea 1.5, limbajul Java pune la dispozitie modalitati simplificate pentru afisarea formata a unor informatii, respectiv pentru citirea de date formate de la tastatura.

3.1 Intrari formate

Clasa **java.util.Scanner** ofera o solutie simpla pentru formatarea unor informatii citite de pe un flux de intrare fie pe octeti, fie pe caractere, sau chiar dintr-un obiect de tip File. Pentru a citi de la tastatura vom specifica ca argument al constructorului fluxul **System.in**:

```
Scanner s = Scanner.create(System.in);  
String nume = s.next();  
int varsta = s.nextInt();  
double salariu = s.nextDouble();  
s.close();
```

3.2 Iesiri formate

Laborator - Programare orientată pe obiecte Limbajul Java

Adrian Runceanu

Clasele `PrintStream` și `PrintWriter` pun la dispoziție, pe lângă metodele `print`, `println` care ofereau posibilitatea de a afișa un șir de caractere, și metodele **`format`**, **`printf`** (echivalente) ce permit afișarea formatată a unor variabile.

`System.out.printf("%s %8.2f %2d %n", nume, salariu, varsta);`

Formatarea șirurilor de caractere se bazează pe clasa **`java.util.Formatter`**.

Probleme rezolvate:

Scrieti, compilați și rulați toate exemplele din acest laborator:

L5_1) Se citește un număr n natural. Să se calculeze suma cifrelor lui.

Exemplu: Pentru $n=124$ obținem $s=7$.

```
import java.util.Scanner;
public class lab_5_1 {
    public static void main(String[] args) {
        int n,s=0;
        n = new Scanner(System.in).nextInt();
        System.out.println("Numarul natural = "+n);
        while(n!=0){
            s+=n%10;
            n/=10;
        }
        System.out.println ("Suma cifrelor numarului dat este egala cu = " + s);
    }
}
```

Soluție implementată în compilatorul online <https://www.jdoodle.com/online-java-compiler/>:

Laborator - Programare orientată pe obiecte Limbajul Java

Adrian Runceanu

```

1 import java.util.Scanner;
2 public class lab_5_1 {
3     public static void main(String[] args) {
4         int n,s=0;
5         n = new Scanner(System.in).nextInt();
6         System.out.println("Numarul natural = "+n);
7         while(n!=0){
8             s+=n%10;
9             n/=10;
10        }
11        System.out.println ("Suma cifrelor numarului dat este egala cu = " + s);
12    }
13 }
14

```

Execute Mode, Version, Inputs & Arguments

JDK 17.0.1 Interactive Stdin Inputs

CommandLine Arguments: 124

Execute

Result

CPU Time: 0.35 sec(s), Memory: 37272 kilobyte(s)

```

Numarul natural = 124
Suma cifrelor numarului dat este egala cu = 7

```

L5_2) Se citeste un numar n natural. Sa se afiseze numarul obtinut prin inversarea cifrelor lui.

Exemplu: Pentru n=841 obtinem m=148

```

import java.util.Scanner;
public class lab5_2 {
    public static void main(String[] args) {
        int n;
        n = new Scanner(System.in).nextInt();
        System.out.println("Numarul natural = "+n);
        int m=0;
        while(n!=0){
            m = m*10+n%10;
            n/=10;
        }
        System.out.println ("Numarul obtinut prin inversarea cifrelor este egal cu = " + m);
    }
}

```

Solutie implementata in compilatorul online <https://www.jdoodle.com/online-java-compiler/>:

Laborator - Programare orientată pe obiecte Limbajul Java

Adrian Runceanu

```

1 import java.util.Scanner;
2 public class lab5_2 {
3     public static void main(String[] args) {
4         int n;
5         n = new Scanner(System.in).nextInt();
6         System.out.println("Numarul natural = "+n);
7         int m=0;
8         while(n!=0){
9             m = m*10+n%10;
10            n/=10;
11        }
12        System.out.println ("Numarul obtinut prin inversarea cifrelor este egal cu = " + m);
13    }
14 }
15
16

```

Execute Mode, Version, Inputs & Arguments

JDK 17.0.1 Interactive Stdin Inputs

CommandLine Arguments

841

Execute

Result

CPU Time: 0.24 sec(s), Memory: 37492 kilobyte(s)

```

Numarul natural = 841
Numarul obtinut prin inversarea cifrelor este egal cu = 148

```

L5_3) Se citeste un numar n natural. Sa se afiseze descompunerea sa in factori primi.
Exemplu: Pentru n=12 obtinem $2^2 \cdot 3$.

```

import java.util.Scanner;
public class lab5_3 {
public static void main(String[] args) {
    int n;
    n = new Scanner(System.in).nextInt();
    System.out.println("Numarul natural = "+n);
    int i=2, fm;
    do{
        fm=0;
        while(n%i==0){
            fm++;
            n=n/i;
        }
        if(fm!=0) System.out.println (i+" la puterea "+fm);
        i++;
    }while(n>=1);
}
}

```

Laborator - Programare orientată pe obiecte Limbajul Java

Adrian Runceanu

Soluție implementată în compilatorul online <https://www.jdoodle.com/online-java-compiler/>:

```

1 import java.util.Scanner;
2 public class lab5_3 {
3     public static void main(String[] args) {
4         int n;
5         n = new Scanner(System.in).nextInt();
6         System.out.println("Numarul natural = "+n);
7         int i=2, fm;
8         do{
9             fm=0;
10            while(n%i==0){
11                fm++;
12                n=n/i;
13            }
14            if(fm!=0) System.out.println (i+" la puterea "+fm);
15            i++;
16        }while(n>=1);
17    }
18 }
19

```

Execute Mode, Version, Inputs & Arguments

JDK 17.0.1 Interactive Stdin Inputs

CommandLine Arguments

12

Execute

Result

CPU Time: sec(s), Memory: kilobyte(s)

```

Numarul natural = 12
2 la puterea 2
3 la puterea 1

```

L5_4) Se citește un număr n natural. Să se afișeze toți divizorii numărului dat.
Exemplu: Pentru $n=12$ obținem $\{1,2,3,4,6,12\}$.

```

import java.util.Scanner;
public class lab5_4 {
    public static void main(String[] args) {
        int n;
        n = new Scanner(System.in).nextInt();
        System.out.println("Numarul natural = "+n);
        System.out.println("Divizorii numarului sunt: ");
        int i=1;
        while(i<=n){
            if(n%i==0) System.out.print(i+", ");
            i++;
        }
    }
}

```

Laborator - Programare orientată pe obiecte Limbajul Java

Adrian Runceanu

Soluție implementată în compilatorul online <https://www.jdoodle.com/online-java-compiler/>:

```

1 import java.util.Scanner;
2 public class lab5_4 {
3     public static void main(String[] args) {
4         int n;
5         n = new Scanner(System.in).nextInt();
6         System.out.println("Numarul natural = "+n);
7         System.out.println("Divizorii numarului sunt: ");
8         int i=1;
9         while(i<=n){
10            if(n%i==0) System.out.print(i+", ");
11            i++;
12        }
13    }
14 }
15
16

```

Execute Mode, Version, Inputs & Arguments

JDK 17.0.1 Interactive Stdin Inputs

CommandLine Arguments

12

Execute

Result

CPU Time: 0.23 sec(s), Memory: 37472 kilobyte(s)

```

Numarul natural = 12
Divizorii numarului sunt:
1, 2, 3, 4, 6, 12,

```

L5_5) Se citește un număr n natural. Să se verifice dacă este număr prim sau nu.
Exemplu: Pentru $n=12$ obținem NU ESTE număr PRIM, iar pentru $n=11$ obținem ESTE număr PRIM.

```

import java.util.Scanner;
public class lab2_5 {
    public static void main(String[] args) {
        int n;
        n = new Scanner(System.in).nextInt();
        System.out.println("Numarul natural = "+n);
        int i=2;
        boolean prim=true;
        while(i<=n/2)
        {
            if(n%i==0) prim=false;
            i++;
        }
    }
}

```

Laborator - Programare orientată pe obiecte Limbajul Java

Adrian Runceanu

```
}  
if(prim==true) System.out.println(n+" ESTE numar PRIM! ");  
else System.out.println(n+" NU ESTE numar PRIM! ");  
}  
}
```

Soluție implementată în compilatorul online <https://www.jdoodle.com/online-java-compiler/>:

```
1 import java.util.Scanner;  
2 public class lab2_5 {  
3     public static void main(String[] args) {  
4         int n;  
5         n = new Scanner(System.in).nextInt();  
6         System.out.println("Numarul natural = "+n);  
7         int i=2;  
8         boolean prim=true;  
9         while(i<=n/2)  
10        {  
11            if(n%i==0) prim=false;  
12            i++;  
13        }  
14        if(prim==true) System.out.println(n+" ESTE numar PRIM! ");  
15        else System.out.println(n+" NU ESTE numar PRIM! ");  
16    }  
17 }  
18  
19
```

Execute Mode, Version, Inputs & Arguments

JDK 17.0.1 Interactive Stdin Inputs

CommandLine Arguments

12

Execute

Result

CPU Time: 0.27 sec(s), Memory: 37440 kilobyte(s)

```
Numarul natural = 12  
12 NU ESTE numar PRIM!
```

Probleme propuse spre rezolvare

Lab5_1: Se dau trei numere nenule a, b și k . Sa se verifice daca fractia a/b se simplifica prin k . In caz afirmativ se va afisa si fractia simplificata.

Lab5_2: Sa se verifice daca trei numere naturale a, b și c sunt pitagorice sau nu. Numim numere pitagorice, trei numere care indeplinesc una din conditiile $a^2 = b^2 + c^2$, $b^2 = a^2 + c^2$, $c^2 = a^2 + b^2$.

Lab5_3: Se citesc trei numere a, b, c . Sa se verifice daca aceste numere (puse in orice ordine) sunt in progresie aritmetica si sa se afiseze ratia progresiei in caz afirmativ.

Lab5_4: Se dau trei numere a, b, c . Sa se verifice daca pot reprezenta laturile unui triunghi. In caz afirmativ sa se precizeze ce tip de triunghi este: echilateral, isoscel, dreptunghic sau oarecare.

Lab5_5: Un punct in plan este dat prin coordonatele sale (x, y) . Sa se scrie un program care determina daca punctul este in origine, intr-un cadran (1,2,3 sau 4), sau pe una din semiaxe (Ox , Ox' , Oy , Oy').

Exemplu:

(1,1) – cadranul 1

(0,3) – axa Oy

(-2,4) – cadranul 2

Lab5_6: Sa se calculeze valoarea functiei matematice $f(x)$, pentru o valoare a lui x introdusa de la tastatura:

$$f: \mathbb{R} \rightarrow \mathbb{R}, f(x) = \begin{cases} x^2 + 1, & \text{pentru } x \leq -3 \\ x - 2, & \text{pentru } -3 < x < 3 \\ x^2 - 4x + 5, & \text{pentru } x \geq 3 \end{cases}$$

Lab5_7: Sa se determine cel mai mare divizor comun (*c.m.m.d.c.*) și cel mai mic multiplu comun (*c.m.m.m.c.*) a doua numere intregi citite de tastatura. Cmmdc se va calcula folosind cele doua variante:

➤ algoritmul lui *Euclid*

➤ folosind relatia de mai jos:

$$\text{cmmdc}(x, y) = \begin{cases} \text{cmmdc}(a-b, b), & \text{daca } a > b \\ \text{cmmdc}(a, b-a), & \text{daca } a < b \\ a, & \text{daca } a = b \end{cases}$$

Lab5_8: Sa se verifice daca un numar este numar *perfect* sau nu. Spunem ca un numar este numar perfect daca este egal cu suma divizorilor lui, mai putin el insusi. (Exemplu: numarul 6 este perfect, deoarece este egal cu suma divizorilor sai 1,2,3).

Laborator - Programare orientată pe obiecte Limbajul Java

Adrian Runceanu

Lab5_9: Se citesc n numere întregi. Sa se determine minimul si maximul lor.

Lab5_10: Sa se verifice daca un numar este *palindrom* sau nu. Spunem ca un numar este palindrom daca este egal cu rasturnatul sau (adica numarul format din cifrele de la dreapta la stanga ale numarului initial – exemplu : $n = 25652$).

Bibliografie:

[1] <http://www.pbinfo.ro> *Descrierea site-ului: "www.pbinfo.ro îți propune să rezolvi probleme de informatică, cu evaluator automat. Știi pe loc dacă soluția ta este corectă sau dacă trebuie să mai lucrezi la ea. Problemele sunt grupate după programa de informatică pentru liceu. Dar nu trebuie să fii la liceu ca să rezolvi aceste probleme. Poți fi elev de gimnaziu, student, profesor sau pur și simplu pasionat de informatică. De fapt, trebuie doar să vrei!!"*

[2] <https://www.runceanu.ro/adrian>

[3] Tudor Sorin, Vlad Hutănu, **Bazele programării în Java**, Editura L&S Info-Mat, București, 2005.

[4] Doina Logofatu, **Algoritmi fundamentali în Java. Aplicații** – Editura Polirom, Iași, 2007.

[5] Horia Georgescu, **Introducere în universul Java**, Editura Tehnica, București, 2002.

[6] Adrian Runceanu, Gheorghe Gîlcă, Alina Cotoi, **Aplicații JAVA - introducere teoretică și practică**, Editura Academica Brâncuși din Târgu-Jiu, 2017.