



Laborator 5 – 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**.

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 interfețele standard pentru lucrul cu fluxuri se gasesc în pachetul **java.io**. Deci, orice program care necesita operații de intrare sau ieșire trebuie să contină instrucțiunea de import a pachetului **java.io**:

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

2. Clasificarea fluxurilor

Există trei tipuri de clasificare a fluxurilor:

- 1) După direcția canalului de comunicare deschis fluxurile se împart în:
 - fluxuri de intrare (pentru citirea datelor)
 - fluxuri de ieșire (pentru scrierea datelor)
- 2) După tipul de date pe care operează:
 - fluxuri de octeti (comunicarea serială se realizează pe 8 biți)
 - fluxuri de caracter (comunicarea serială se realizează pe 16 biți)
- 3) După acțiunea lor:
 - fluxuri primare de citire/scriere a datelor (se ocupă efectiv cu citirea/scrierea datelor)
 - fluxuri pentru procesarea datelor

3. Intrari și ieșiri formate

Incepând cu versiunea 1.5, limbajul Java pune la dispozitie modalitati simplificate pentru afisarea formatata a unor informatii, respectiv pentru citirea de date formatate 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 Ieșiri formate

Clasele PrintStream și PrintWriter pun la dispozitie, pe langa metodele print, println care ofera posibilitatea de a afisa un sir de caractere, si metodele **format**, **printf** (echivalente) ce permit afisarea formatata a unor variabile.

System.out.printf("%s %8.2f %2d %n", nume, salariu, varsta);
Formatarea sirurilor de caractere se bazeaza pe clasa **java.util.Formatter**.

Probleme rezolvate:

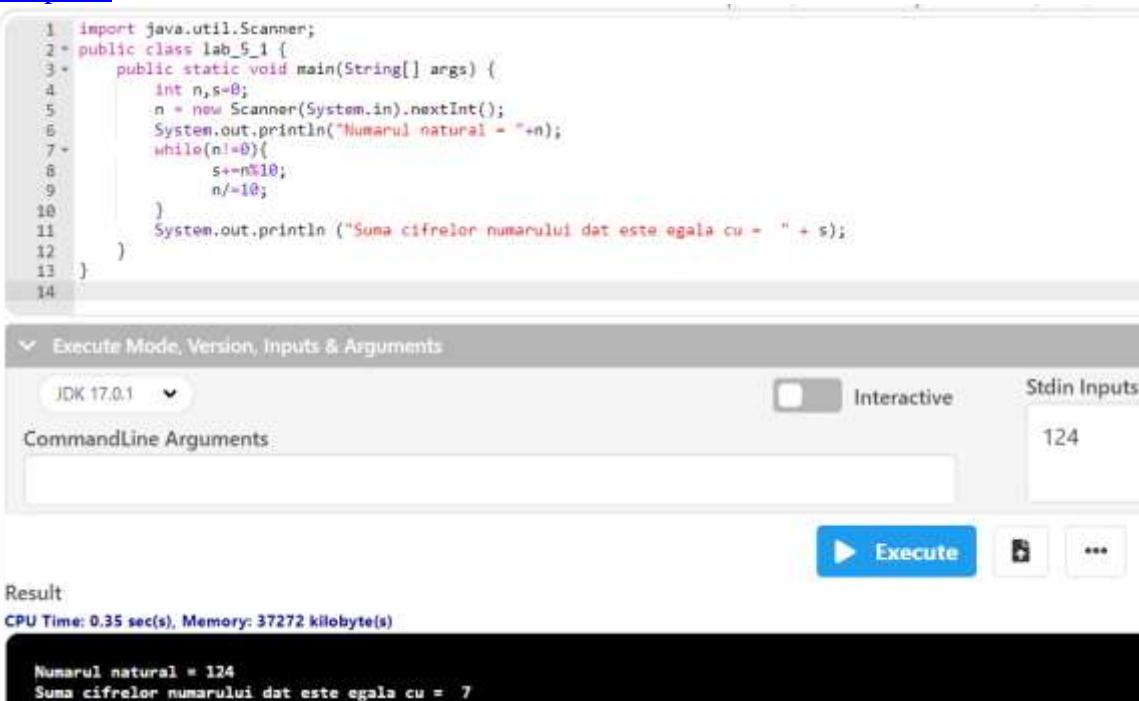
Scripti, compilati si rulati toate exemplele din acest laborator:

L5_1) Se citeste un numar n natural. Sa se calculeze suma cifrelor lui.

Exemplu: Pentru n=124 obtinem 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);
    }
}
```

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



The screenshot shows the jdoodle online Java compiler interface. The code area contains the provided Java code for calculating the sum of digits of a number. The execution mode is set to 'Interactive' with 'Stdin Inputs' containing the value '124'. The result section shows the output: 'Numarul natural = 124' and 'Suma cifrelor numarului dat este egala cu = 7'. The CPU time and memory usage are also displayed.

```
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
124

CommandLine Arguments

Result

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

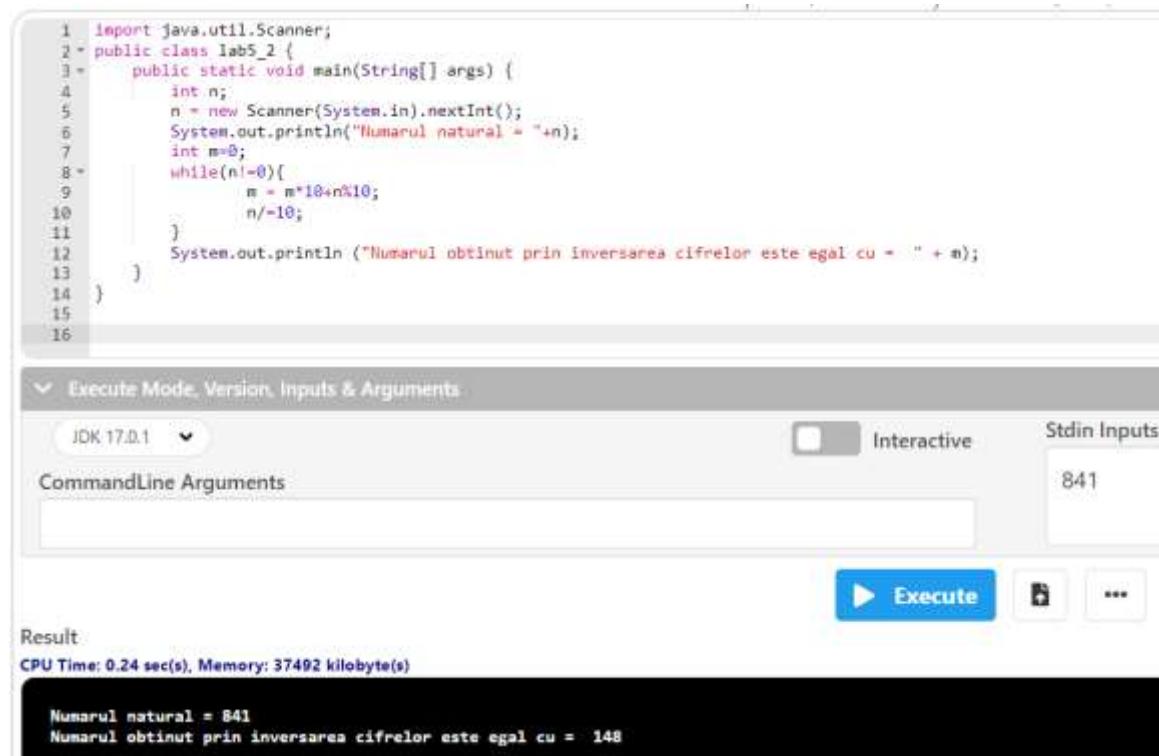
```
Numarul natural = 124
Suma cifrelor numarului dat este egala cu = 7
```

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/>:



The screenshot shows the jdoodle Java online compiler interface. The code area contains the provided Java code. The execution mode is set to "Interactive". The "Stdin Inputs" field contains the value "841". The "Execute" button is visible at the bottom right. The "Result" section displays the output: "Numarul natural = 841" and "Numarul obtinut prin inversarea cifrelor este egal cu = 148".

```
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 }
```

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);
    }
}
```

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

The screenshot shows the jdoodle Java online compiler interface. The code area contains the provided Java code for prime factorization. The execution settings are set to "Interactive" mode with JDK 17.0.1, and the standard input field contains the value "12". The result section displays the output of the program, which is "Numarul natural = 12", followed by "2 la puterea 2" and "3 la puterea 1".

```
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 }

```

Execute Mode, Version, Inputs & Arguments

JDK 17.0.1 Interactive Stdin Inputs
12

Result

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

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

L5_4) Se citeste un numar n natural. Sa se afiseze toti divizorii numarului dat.
Exemplu: Pentru n=12 obtinem {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++;
        }
    }
}
```

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

The screenshot shows the jdoodle Java online compiler interface. The code area contains the provided Java code. Below it, the 'Execute Mode, Version, Inputs & Arguments' section is visible, showing 'JDK 17.0.1' selected, 'Interactive' mode, and 'Stdin Inputs' set to '12'. Under 'CommandLine Arguments', there is an empty input field. At the bottom, there are 'Execute', 'Run', and '...' buttons. The 'Result' section displays the output of the program: 'Numarul natural = 12', 'Divizorii numarului sunt:', and '1, 2, 3, 4, 6, 12,'.

L5_5) Se citeste un numar n natural. Sa se verifice daca este numar prim sau nu.

Exemplu: Pentru n=12 obtinem NU ESTE numar PRIM, iar pentru n=11 obtinem ESTE numar 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++;
        }
        if(prim==true) System.out.println(n+" ESTE numar PRIM! ");
        else System.out.println(n+" NU ESTE numar PRIM! ");
    }
}
```

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

The screenshot shows the jdoodle Java online compiler interface. The code area contains the provided Java code for checking if a number is prime. Below it, the 'Execute Mode, Version, Inputs & Arguments' section is visible, showing 'JDK 17.0.1' selected, 'Interactive' mode off, and 'Stdin Inputs' set to '12'. The 'CommandLine Arguments' field is empty. At the bottom, there are 'Execute', 'Run', and '...' buttons. The 'Result' section shows the output: 'Numarul natural = 12' and '12 NU ESTE numar PRIM!'.

```
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
12

CommandLine Arguments

Execute Run ...

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 si 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 si 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 (pusе 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 coodonatele 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:

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

Lab5_7: Sa se determine cel mai mare divizor comun (*c.m.m.d.c.*) si 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:

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

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).

Lab5_9: Se citesc n numere intregi. 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] Adrian Runcceanu „*Programarea și utilizarea calculatoarelor*”, Editura Academica Brâncuși din Târgu-Jiu, 2003, ISBN 973-8436-44-3
- [4] Adrian Runcceanu, Mihaela Runcceanu, „*Noțiuni de programare – limbajul C++*”, Editura Academica Brâncuși din Târgu-Jiu, 2012, ISBN 978-973-144-550-2
- [5] Adrian Runcceanu, Mihaela Runcceanu, „*Algoritmi implementati in limbajul C++.* Volumul I – *Algoritmi elementari*”, Editura Academica Brâncuși din Târgu Jiu, 2021, ISBN 978-606-9614-06-8