# SwTc - Seminar 3

# Lucrul cu fluxuri de intrare-iesire si socket-uri

## *5.1. Descrierea seminarului*

-

## *5.2. Probleme rezolvate si probleme propuse spre rezolvare*

**Problema 5.1.**

**Sa se creeze** in limbajul Java **o clasa** numita **`Pachet`** care are:

**- doua atribute** (variabile membru) **cu caracter de obiect** (`non-static`), cu **nivel de acces** **`protected`**, de **tip sir de caractere**, numite **`mesaj`** si **`adresaDestinatie`**;

**- trei atribute cu caracter de obiect** (`non-static`), cu **nivel de acces** **`protected`**, de **tip intreg**, numite **`portDestinatie`, `lungimeMesaj`,** si respectiv **`numarOrdine`**;

**- un atribut cu caracter global** (`static`), cu **nivel de acces** **`public`**, de **tip intreg**, numit **`numarPachete`**;

**- un constructor cu nivel de acces** **`public`**, care:

- **primeste un parametru de tip sir de caractere** numit **`adresaDestinatie`**, **un parametru de tip intreg** numit **`portDestinatie`** si **un parametru de tip sir de caractere** numit **`mesaj`**,

- **incrementeaza valoarea atributului** **`numarPachete`,**

- **foloseste valorile primite pentru a initializa atributele cu caracter de obiect** (`non-static`)**, cu exceptia atributului** **`numarOrdine`** care este **initializat cu valoarea actualizata a atributului** **`numarPachete`**;

**- o metoda cu nivel de acces** **`public`** numita **`afiseazaPachet()`** care:

- **nu primeste parametri,** si **nu returneaza valori**,

- **afiseaza la consola valorile celor cinci atribute in urmatorul format:**

```
Pachet de lungime 5 adresat localhost:2000 cu numar de ordine 1
al carui continut este <<hello>>
```

**- trei metode cu nivel de acces** **`public`,** numite **`obtineAdresaDestinatie()`, `obtinePortDestinatie()`** si **`obtineMesaj()`** **care:**

- **nu primesc parametri,**

- **returneaza valorile atributelor** **`adresaDestinatie`, `portDestinatie`** **si** **`mesaj`.**

***Rezolvare:***

**O posibila rezolvare** este urmatoarea:

```
 1   class Pachet {
 2     protected String mesaj;
 3     protected String adresaDestinatie;
 4     protected int portDestinatie;
 5     protected int lungimeMesaj;
 6     protected int numarOrdine;
 7     public static int numarPachete = 0;
 8
 9     public Pachet(String adresaDestinatie, int portDestinatie, String mesaj) {
10       this.adresaDestinatie = adresaDestinatie;
11       this.portDestinatie = portDestinatie;
12       this.mesaj = mesaj;
13       this.lungimeMesaj = mesaj.length();
14       numarPachete++;
15       this.numarOrdine = numarPachete;
16     }
17
18     public void afiseazaPachet() {
19       System.out.println("Pachet de lungime " + lungimeMesaj + " adresat " +
20          adresaDestinatie + ":" + portDestinatie + " cu numar de ordine " +
21          numarOrdine + " al carui continut este <<" + mesaj + ">>");
22     }
23
24     public String obtineAdresaDestinatie() {
25       return adresaDestinatie;
26     }
27
28     public int obtinePortDestinatie() {
29       return portDestinatie;
30     }
31
32     public String obtineMesaj() {
33       return mesaj;
34     }
35
36     public static void main(String[] args) {
37       Pachet p = new Pachet("localhost", 2000, "hello");
38       p.afiseazaPachet();
39       System.out.println("Numarul pachetelor create este " + numarPachete);
40       p = new Pachet("localhost", 3000, "hi");
41       p.afiseazaPachet();
42       System.out.println("Numarul pachetelor create este " + numarPachete);
43     }
44   }
```

## Problema 5.2.

**a) Sa se scrie** in limbajul Java o **clasa `ReceptiePachete` care aiba urmatoarea specificatie** (se vor declara si implementa toate variabilele membru si metodele specificate):

```
 1  class ReceptiePachete {
 2
 3    protected ServerSocket server;
 4
 5    protected Socket socket;
 6
 7    protected DataInputStream inRetea;
 8
 9    protected DataOutputStream outRetea;
10
11    protected BufferedInputStream inBuffered;
12
13    protected BufferedOutputStream outBuffered;
14
15    public ReceptiePachete(int portLocal) {
16
17      // constructorul initializeaza atributul server pasandu-i
18      // constructorului ServerSocket() parametrul portLocal
19      // apoi obtine de la server un obiect cu care initializeaza
20      // atributul socket si in final initializeaza atributele inRetea
21      // si outRetea cu fluxuri atasate la fluxurile obtinute de la
22      // socket, prin intermediul fluxurilor cu stocare temporara
23      // inBuffered si outBuffered
24    }
25
26    public Pachet receptiePachet() {
27
28      // metoda citeste succesiv prin fluxul de intrare (inRetea)
29      // atributele unei variabile locale pachet de tip Pachet care
30      // au fost scrise in fluxul de iesire atasat socket-ului pereche
31      // cu ajutorul urmatoarei secvente de cod:
32      //
33      // outRetea.writeUTF(pachet.mesaj);
34      // outRetea.writeUTF(pachet.adresaDestinatie);
35      // outRetea.writeInt(pachet.portDestinatie);
36      // outRetea.writeInt(pachet.lungimeMesaj);
37      // outRetea.writeInt(pachet.numarOrdine);
38      // outRetea.flush();
39      //
40      // apoi creaza un Pachet cu ajutorul valorilor citite, si apeleaza
41      // metoda care produce afisarea informatiilor pachetului
42    }
43  }
```

**b) Sa se creeze diagrama UML de secventa pentru acest program.**

## Problema 5.3.

**a) Sa se scrie** in limbajul Java o **clasa `TransmitерePachete` care aiba urmatoarea specificatie** (se vor declara si implementa toate variabilele membru si metodele specificate):

```
 1  class TransmiterePachete {
 2
 3    protected Socket socket;
 4
 5    protected DataInputStream inRetea;
 6
 7    protected DataOutputStream outRetea;
 8
 9    protected BufferedInputStream inBuffered;
10
11    protected BufferedOutputStream outBuffered;
12
13    public TransmiterePachete(String adresaServer, int portServer) {
14
15      // constructorul initializeaza atributul socket folosind parametrii
16      // primiti, apoi initializeaza atributele inRetea si outRetea
17      // cu fluxuri atasate la fluxurile obtinute de la socket,
18      // prin intermediul fluxurilor cu stocare temporara
19      // inBuffered si outBuffered
20
21    public void transmiterePachet(Pachet pachet) {
22
23      // metoda trimite succesiv prin fluxul de iesire (outRetea)
24      // atributele parametrului pachet de tip Pachet astfel incat din
25      // fluxul de intrare atasat socket-ului pereche sa se poata citi
26      // informatiile cu ajutorul urmatoarei secvente de cod:
27      //
28      // pachet.lungimeMesaj = inRetea.readInt();
29      // pachet.mesaj = inRetea.readUTF();
30      // pachet.adresaDestinatie = inRetea.readUTF();
31      // pachet.portDestinatie = inRetea.readInt();
32      // pachet.numarOrdine = inRetea.readInt();
33    }
34  }
```

**b) Sa se creeze diagrama UML de secventa pentru acest program.**

## Problema 5.4.

**a) Pornind de la cele doua programe anterior scrise** (Problema 5.2. si Problema 5.3.) **sa se creeze o noua clasa, abstracta, numita `ElementeRetea`, care sa incapsuleze elementele comune celor doua clase.**

**b) Sa se rescrie codurile claselor `TransmiterePachete` si `ReceptiePachete`** pentru cazul in care ele **extind prin mostenire clasa `ElementeRetea`.**

**c) Sa se creeze diagrama UML de clase pentru ansamblul claselor nou create** (`ElementeRetea`, `TransmiterePachete` si `ReceptiePachete`).

## Problema 5.5.

**Se da urmatorul program Java**.

```
 1  import java.io.*;
 2
 3  class Pachet {
 4    public int tip;
 5    public int lungime;
 6    public byte[] date;
 7  }
 8
 9  public class LucruCuPacheteDate1 {
10    static int numarPachete;
11    static Pachet[] tablouPachete = new Pachet[100];
12
13    public static boolean adaugarePachet(int tip, String date) {
14      // creeaza un pachet nou de tipul primit ca parametru
15      // folosind ca date sirul de caractere primit ca parametru
16      // si adauga acel pachet la tabloul de pachete
17      // returneaza true daca operatia s-a efectuat cu succes
18      // si false daca operatia nu a putut fi executata
19    }
20
21    public static void afisarePachete(Pachet[] tp, int numarP) {
22      // afiseaza tipul, lungimea si continutul tuturor pachetelor
23      // din tabloul primit ca parametru
24      }
25    }
26    public static void main (String args[]) throws IOException {
27      BufferedReader inConsola  = new BufferedReader(new
28                                          InputStreamReader(System.in));
29      String sirCitit;
30      int tip;
31
32      for (int no=0; no<5; no++) {
33        System.out.print("Pachet nou, de tipul (intreg): ");
34        sirCitit = inConsola.readLine();
35        tip = Integer.parseInt(sirCitit);
36        System.out.print("Continut: ");
37        sirCitit = inConsola.readLine();
38        if (adaugarePachet(tip, sirCitit))
39                System.out.println("Pachet creat cu succes!\n");
40        else    System.out.println("Pachetul nu a putut fi creat!\n");
41      }
42
43      System.out.println("Afisarea pachetelor: ");
44      afisarePachete(tablouPachete, numarPachete);
45    }
46  }
```

**a) Sa se completeze codul metodelor `adaugarePachet()` si `afisarePachete()`.**

***Rezolvare:***

**O posibila rezolvare** este urmatoarea:

```
 1  import java.io.*;
 2
 3  class Pachet {
 4    public int tip;
 5    public int lungime;
 6    public byte[] date;
 7  }
 8
 9  public class LucruCuPacheteDate1 {
10    static int numarPachete;
11    static Pachet[] tablouPachete = new Pachet[100];
12
13    public static boolean adaugarePachet(int tip, String date) {
14      if (numarPachete < 100) {
15        tablouPachete[numarPachete] = new Pachet();
16        tablouPachete[numarPachete].tip = tip;
17        tablouPachete[numarPachete].lungime = date.length();
18        tablouPachete[numarPachete].date = date.getBytes();
19        numarPachete++;
20        return(true);
21      }
22      else {
23        return (false);
24      }
25    }
26
27    public static void afisarePachete(Pachet[] tp, int numarP) {
28      for (int i=0; i<numarP; i++) {
29        System.out.print("Pachetul tip: " + tp[i].tip);
30        System.out.print("  de lungime " + tp[i].lungime);
31        System.out.println("  contine: " + (new String(tp[i].date)));
32      }
33    }
34    public static void main (String args[]) throws IOException {
35      BufferedReader inConsola  = new BufferedReader(new
36                                              InputStreamReader(System.in));
37      String sirCitit;
38      int tip;
39
40      for (int no=0; no<5; no++) {
41        System.out.print("Pachet nou, de tipul (intreg): ");
42        sirCitit = inConsola.readLine();
43        tip = Integer.parseInt(sirCitit);
44        System.out.print("Continut: ");
45        sirCitit = inConsola.readLine();
46        if (adaugarePachet(tip, sirCitit))
47                System.out.println("Pachet creat cu succes!\n");
48        else    System.out.println("Pachetul nu a putut fi creat!\n");
49      }
50
51      System.out.println("Afisarea pachetelor: ");
52      afisarePachete(tablouPachete, numarPachete);
53    }
54  }
```

**b) Sa se creeze diagrama de secventa a mesajelor (MSC) pentru codul metodei `main()`.**

## Problema 5.6.

**a) Sa se adauge clasei anterioare o metoda care permite selectia pachetelor ce au lungime maxima specificata, metoda primind ca parametru aceasta lungime si returnand un tablou de pachete core contine pachetele selectate.**

***Rezolvare:*** O posibila rezolvare este urmatoarea:

```
 1    public static Pachet[] selectiePachete(int lungimeMaxima) {
 2      int k = 0;
 3
 4      for (int i=0; i<numarPachete; i++) {
 5        if (tablouPachete[i].lungime <= lungimeMaxima)  k++;
 6      }
 7
 8      Pachet[] tp = new Pachet[k];
 9
10      k = 0;
11
12      for (int i=0; i<numarPachete; i++) {
13        if (tablouPachete[i].lungime <= lungimeMaxima) {
14          tp[k] = new Pachet();
15          tp[k].tip = tablouPachete[i].tip;
16          tp[k].lungime = tablouPachete[i].lungime;
17          tp[k].date = (new String(tablouPachete[i].date)).getBytes();
18          k++;
19        }
20      }
21      return (tp);
22    }
```

**b) Sa se adauge modifice metoda main() care permite selectia pachetelor ce au lungime maxima specificata, metoda primind ca parametru aceasta lungime si returnand un tablou de pachete core contine pachetele selectate.**

***Rezolvare:*** O posibila rezolvare este urmatoarea:

```
 1    public static void main (String args[]) throws IOException {
 2
 3      BufferedReader inConsola  = new BufferedReader(new
 4                                       InputStreamReader(System.in));
 5      String sirCitit;
 6      int tip, limita;
 7      Pachet[] pacheteSelectate;
 8
 9      for (int no=0; no<5; no++) {
10        System.out.print("Pachet nou, de tipul (intreg): ");
11        sirCitit = inConsola.readLine();
12        tip = Integer.parseInt(sirCitit);
13        System.out.print("Continut: ");
14        sirCitit = inConsola.readLine();
15        if (adaugarePachet(tip, sirCitit))
16                System.out.println("Pachet creat cu succes!\n");
17        else    System.out.println("Pachetul nu a putut fi creat!\n");
18      }
19
20      System.out.println("Afisarea pachetelor: ");
21
22      afisarePachete(tablouPachete, numarPachete);
23
24      System.out.println("\nLimita de lungime pentru selectie pachete: ");
25      sirCitit = inConsola.readLine();
26      limita = Integer.parseInt(sirCitit);
27
28      pacheteSelectate = selectiePachete(limita);
29
30      System.out.println("\nAfisarea pachetelor selectate: \n");
31
32      afisarePachete(pacheteSelectate, pacheteSelectate.length);
33    }
34  }
```