Subjectul I:

- 1. B
- 2. D
- 3. A
- 4. C
- 5. C

Subjectul II:

```
LinkedList<Automobil> la = new LinkedList<>();
la.add(new Automobil("BMW", "X5", 2000.5, 5000));
la.add(new Automobil("Mercedes", "E Class", 3500, 6000));
la.add(new Automobil("Audi", "A5", 1500, 3000));
la.add(new Automobil("BMW", "X2", 2000.5, 2000));
la.add(new Automobil("BMW", "X6", 2800.75, 4500));
la.add(new Automobil("BMW", "X1", 1600.5, 5000));
la.add(new Automobil("Mercedes", "S Class", 2200.25, 15000));
la.add(new Automobil("Audi", "A6", 2000.5, 4000));
System.out.println("Lista initiala:");
la.stream().forEach(System.out::println);
System.out.println();
System.out.println("Cerinta a)");
la.stream().filter(a -> a.getPret() >= 5000).
sorted(Comparator.comparing(Automobil::getPret).reversed()).forEach(System.out::println);
System.out.println();
System.out.println("Cerinta b)");
la.stream().map(Automobil::getMarca).distinct().forEach(System.out::println);
System.out.println();
```

```
System.out.println("Cerinta c)");
List<Automobil> In = la.stream().filter(a -> a.getCapacitate() >= 2000).
                   filter(a -> a.getCapacitate() <= 3000).collect(Collectors.toList());
In.stream().forEach(System.out::println);
System.out.println();
System.out.println("Cerinta d)");
System.out.println(la.stream().filter(a -> a.getMarca().equals("Audi")).
            collect(Collectors.minBy(Comparator.comparing(Automobil::getPret))));
System.out.println();
      Subjectul III:
      class FirNumarare extends Thread
        private String numeFisier;
        private String cuvantCautat;
        private int nrAparitii;
        public FirNumarare (String numeFisier, String cuvantCautat)
          this.numeFisier = numeFisier;
          this.cuvantCautat = cuvantCautat;
          this.nrAparitii = 0;
        }
        public int getNrAparitii()
          return nrAparitii;
        }
```

```
@Override
  public void run()
  {
      Scanner in = new Scanner(new File(numeFisier));
      while(in.hasNextLine())
         String linie = in.nextLine();
         String []cuvinte = linie.split("[ .,\n]+");
         for(int i = 0; i < cuvinte.length; i++)</pre>
           if(cuvinte[i].equals(cuvantCautat))
             nrAparitii++;
      }
      in.close();
  }
public class NumarareCuvinteFisier
  public static void main(String[] args)
  {
    Scanner in = new Scanner(System.in);
    System.out.println("Cuvantul cautat: ");
    String cuv = in.next();
    FirNumarare f_1 = new FirNumarareCuvinteFisier("exemplu_1.txt", cuv);
    FirNumarare f 2 = new FirNumarareCuvinteFisier("exemplu 2.txt", cuv);
    FirNumarare f 3 = new FirNumarareCuvinteFisier("exemplu 3.txt", cuv);
    f_1.start();
     f_1.join();
     f_2.start();
     f_2.join();
     f_3.start();
     f_3.join();
    int t = f 1.getNrAparitii() + f 2.getNrAparitii() + f 3.getNrAparitii();
    System.out.println("Cuvantul" + cuv + " apare de " + t + " ori in fisierele date!");
  }
}
```

Subjectul IV:

```
try (Connection conn =
DriverManager.getConnection("jdbc:derby://localhost:1527/AngajatiDB", "root", "12345");)
{
      Scanner in = new Scanner(System.in);
      System.out.println("Salariul minim: ");
      float s = in.nextFloat();
      System.out.println("Varsta maxima: ");
      int v = in.nextInt();
      PreparedStatement pst = null;
      ResultSet rs = null;
      pst = conn.prepareStatement("SELECT * FROM Angajati WHERE Salariu > ? AND Varsta
<?");
      pst.setFloat(1, s);
      pst.setInt(2, v);
      rs = pst.executeQuery();
      while (rs.next()) {
            System.out.println(rs.getString("Nume") + " " + rs.getFloat("Salariu") + " " +
                                rs.getInt("Varsta"));
            }
          } catch (SQLException ex) {
            System.out.println("Eroare JDBC: " + ex);
      }
```