# Subiectul02 - Rezolvare

## 1

```
public class Employee: IComparable<Employee> {
  // cod, nume, data nasterii, telefon, adresa, salariu;
  public string Code { get; set; }
  public string Name { get; set; }
  public DateTime DateOfBirth { get; set; }
  public string Phone { get; set; }
  public string Address { get; set; }
  public int Salary { get; set; }

  public Employee() {}
  public Employee(string code, string name, DateTime date, string phone, string
address, int salary)   {
    Code = code;
    Name = name;
    DateOfBirth = date;
    Phone = phone;
    Address = address;
    Salary = salary;
  }
}
```

## 2.

```
public override string ToString() {
  return $ "{Code} ; {Name} ; {DateOfBirth} ; {Phone} ; {Address} ; {Salary}";
}
```

## 3.

```
 public class Employee: IComparable<Employee> {
  // ...
  public int CompareTo(Employee other) {
    if (this.Salary < other.Salary)
      return -1;
    if (this.Salary == other.Salary)
      return 0;
    return 1;
  }
}
```

## 4.

```
public class Employer {
  // lista angajatilor (de tip Employee), nume si adresa.
  public string Name { get; set; }
  public string Address { get; set; }
  public List<Employee> EmployeeList { get; set; }
  public Employer() {
    EmployeeList = new List<Employee>();
  }
  public Employer(string name, string address, List < Employee > list) {
    Name = name;
    Address = address;
    EmployeeList = list;
  }
}
```

## 5.

### a.

```
// In clasa Employee
public override bool Equals(object obj) {
  if (obj == null)
    return false;
  if (!(obj is Employee))
    return false;
  Employee emp = obj as Employee;
  return Code == emp.Code &&
    Name == emp.Name &&
    DateOfBirth == emp.DateOfBirth &&
    Phone == emp.Phone &&
    Address == emp.Address &&
    Salary == emp.Salary;
}
public override int GetHashCode() {
  return base.GetHashCode();
}

// In clasa Employer
public void AddEmployee(Employee e) {
  bool isAdded = false;
  foreach(var emp in EmployeeList)
  if (emp.Equals(e)) {
    isAdded = true;
    break;
  }
  if (!isAdded)
    EmployeeList.Add(e);
}
```

**b.**

```
// Dependente folosite
using Newtonsoft.Json;
using System.IO;

// In App.config
<appSettings>
  <add key="DiskPath" value="D:\"/>
</appSettings>

// In clasa Employer
public void SaveOnDisk() {
  string json = JsonConvert.SerializeObject(this);
  string path =
System.Configuration.ConfigurationManager.AppSettings["DiskPath"];
  File.WriteAllText($ @ "{path}\{Name}.json", json);
}
```

## 6.

```
public delegate bool Func(Employee emp);

// In clasa Employer
public List<Employee> Filter(Func func) {
  return EmployeeList.Where(x => func(x)).ToList();
}
```

## 7.

```
static void Main(string[] args) {
  List<Employee> list = new List<Employee>();

  list.Add(new Employee("c01", "Andrei P.", new DateTime(1980, 11, 30), "",
"Pitesti", 2200));
  list.Add(new Employee("c02", "George P.", new DateTime(1990, 10, 28), "",
"Giurgiu", 1200));
  list.Add(new Employee("c03", "Mihai S.", new DateTime(1990, 7, 25), "0710",
"Gruiu, Cateasca", 1800));
  list.Add(new Employee("c04", "Costache G.", new DateTime(1986, 10, 21), "0720",
"Pitesti", 3600));
  list.Add(new Employee("c05", "Mincu M.", new DateTime(2000, 8, 11), "0730",
"Pitesti", 4000));

  Employer employer = new Employer("Nitescu D.", "Pitesti", list);
  Func f1 = (Employee e) => {
    int now = int.Parse(DateTime.Now.ToString("yyyyMMdd"));
    int dob = int.Parse(e.DateOfBirth.ToString("yyyyMMdd"));
    int age = (now - dob) / 10000;
    return e.Salary >= 2000 && age < 30;
  };

  Func f2 = (Employee e) => {
    bool adr = e.Address.Split(',').Where(x =>
x.Contains("Pitesti")).ToList().Count != 0;
    return adr && e.Phone.Length != 0;
  };

  Console.WriteLine("Filter 1");
  foreach(var e in employer.Filter(f1))
    Console.WriteLine(e);

  Console.WriteLine("\nFilter 2");
  foreach(var e in employer.Filter(f2))
    Console.WriteLine(e);

  Console.ReadKey();
}
```

## Output

```
Filter 1
c05 ; Mincu M. ; 11.08.2000 00:00:00 ; 0730 ; Pitesti ; 4000

Filter 2
c04 ; Costache G. ; 21.10.1986 00:00:00 ; 0720 ; Pitesti ; 3600
c05 ; Mincu M. ; 11.08.2000 00:00:00 ; 0730 ; Pitesti ; 4000
```

## Full Code

**Link**: https://dotnetfiddle.net/v2VJs0

```
using Newtonsoft.Json;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleAppExamen {
  public class Employee: IComparable<Employee> {
    // cod, nume, data nasterii, telefon, adresa, salariu;
    public string Code { get; set; }
    public string Name { get; set; }
    public DateTime DateOfBirth { get; set; }
    public string Phone { get; set; }
    public string Address { get; set; }
    public int Salary { get; set; }

    public Employee() { }
    public Employee(string code, string name, DateTime date, string phone, string
address, int salary) {
      Code = code;
      Name = name;
      DateOfBirth = date;
      Phone = phone;
      Address = address;
      Salary = salary;
    }

    public int CompareTo(Employee other) {
      if (this.Salary < other.Salary)
        return -1;
      if (this.Salary == other.Salary)
        return 0;
      return 1;
    }

    public override string ToString() {
      return $"{Code} ; {Name} ; {DateOfBirth} ; {Phone} ; {Address} ; {Salary}";
    }

    public override bool Equals(object obj) {
      if (obj == null)
        return false;
      if (!(obj is Employee))
        return false;
      Employee emp = obj as Employee;
      return Code == emp.Code &&
        Name == emp.Name &&
        DateOfBirth == emp.DateOfBirth &&
        Phone == emp.Phone &&
        Address == emp.Address &&
        Salary == emp.Salary;
    }

    public override int GetHashCode() {
      return base.GetHashCode();
    }
  }

  public class Employer {
    // lista angajatilor (de tip Employee), nume si adresa.
    public string Name { get; set; }
    public string Address { get; set; }
    public List<Employee> EmployeeList { get; set; }

    public Employer() {
      EmployeeList = new List<Employee> ();
    }

    public Employer(string name, string address, List < Employee > list) {
      Name = name;
      Address = address;
      EmployeeList = list;
    }
    public void AddEmployee(Employee e) {
      bool isAdded = false;
      foreach(var emp in EmployeeList)
      if (emp.Equals(e)) {
        isAdded = true;
        break;
      }
      if (!isAdded)
        EmployeeList.Add(e);
    }

    public void SaveOnDisk() {
      string json = JsonConvert.SerializeObject(this);
      string path =
System.Configuration.ConfigurationManager.AppSettings["DiskPath"];
      File.WriteAllText($@"{path}\{Name}.json", json);
    }

    public List<Employee> Filter(Func func) {
      return EmployeeList.Where(x => func(x)).ToList();
    }
  }

  public delegate bool Func(Employee emp);

  class Program {
    static void Main(string[] args) {
      List<Employee> list = new List<Employee>();

      list.Add(new Employee("c01", "Andrei P.", new DateTime(1980, 11, 30), "",
"Pitesti", 2200));
      list.Add(new Employee("c02", "George P.", new DateTime(1990, 10, 28), "",
"Giurgiu", 1200));
      list.Add(new Employee("c03", "Mihai S.", new DateTime(1990, 7, 25),
"0710...", "Gruiu, Cateasca", 1800));
      list.Add(new Employee("c04", "Costache G.", new DateTime(1986, 10, 21),
"0720", "Pitesti", 3600));
      list.Add(new Employee("c05", "Mincu M.", new DateTime(2000, 8, 11), "0730",
"Pitesti", 4000));

      Employer employer = new Employer("Nitescu D.", "Pitesti", list);

      Func f1 = (Employee e) => {
        int now = int.Parse(DateTime.Now.ToString("yyyyMMdd"));
        int dob = int.Parse(e.DateOfBirth.ToString("yyyyMMdd"));
        int age = (now - dob) / 10000;
        return e.Salary >= 2000 && age < 30;
      };

      Func f2 = (Employee e) => {
        bool adr = e.Address.Split(',').Where(x =>
x.Contains("Pitesti")).ToList().Count != 0;
        return adr && e.Phone.Length != 0;
      };

      Console.WriteLine("Filter 1");
      foreach(var e in employer.Filter(f1))
      Console.WriteLine(e);
```

```
            Console.WriteLine("\nFilter 2");
            foreach(var e in employer.Filter(f2))
            Console.WriteLine(e);

            Console.ReadKey();
        }
    }
}
```