# CVU Intelligence Program Practica Backend 2021

## 1. Introducere:

**Programul de pregatire se desfășoară pe o perioada de 12 saptamani, fiecare secțiune avand o perioada recomandata de studiu pana la finalizare.**

## 2. C# Fundamentals

**Partea de C# este structurată după cartea C# in a Nutshell pe care o veti primi impreuna cu acest document.**

**In partea din dreapta a tabelelor sunt capitolele din carte care trebuie parcurse pentru a avea cunoștințele necesare rezolvării exercițiilor propuse. Pe langa carte aveti in document si o lista de link-uri recomandate pentru studiu, puteti de asemenea sa invatati si din alte surse decat cele prezentate in document (youtube, pluralsight, udemy, etc.)**

**https://docs.microsoft.com/en-us/dotnet/csharp/**
**https://www.youtube.com/watch?v=SXmVym6L8dw&list=PLAC325451207E3105**
**https://www.tutorialspoint.com/csharp/index.htm**

### 2. 1. Language Basics - 3 zile

| | |
|---|---|
| 1) **Scrie un program care:**<br>a) citește de la tastatura un număr cu minim de 3 cifre dacă numărul are mai puțin de 3 cifre, îi spune utilizatorului acest lucru și îi cere alt număr<br>b) dacă numărul este corect, calculeaza valoarea in oglinda (ex 441 in oglinda este 144)<br>c) verifica și îi spune userului dacă numărul dat în oglinda este pătrat perfect (ex 144 = 12 * 12 - ok) | **Cap 2 - C# Language Basics**<br>• **A First C# Program 11**<br>• **Syntax 14**<br>• **Type Basics 17**<br>• **Numeric Types 26**<br>• **Boolean Type and Operators 33**<br>• **Strings and Characters 35**<br>• **Arrays 38**<br>• **Variables and Parameters 42**<br>• **Expressions and Operators 51**<br>• **Null Operators 55**<br>• **Statements 56**<br>• **Namespaces 65** |
| 2) **Scrie un program care:**<br>a) citește de la tastatura o lista de numere (pot fi si numere reale) despărțite prin spațiu și le salvează într-un array<br>b) parcurge array-ul și le afișează pe cele care nu sunt întregi<br>c) cauta si afiseaza cel mai mic numar fara sa folosesti funcții din .NET (Math.Min) | |

| | |
|---|---|
| 3) **Scrie un program care:**<br><br>a) citește de la tastatura 3 nume de persoane (ex. George Ion Maria)<br><br>b) afișează pe cate o linie ce caracter a apărut în fiecare nume și de cate ori indiferent ca-i cu litera mica sau mare | |

## 2.2. OOP Basics - 7 zile

| | |
|---|---|
| **1. Creeaza un proiect care:**<br><br>a. are următoarele clase: Animal, Carnivor, Erbivor, Omnivor, Mancare, Carne, Planta<br><br>b. clasa Mâncare este abstractă și are următoarele câmpuri publice:<br><br>i. greutate: decimal<br>ii. energie: decimal - valori intre 0 si 0.05<br><br>c. clasele Carne si Planta extind Mancare<br><br>d. clasa Animal este abstractă și are următoarele:<br><br>i. campuri vizibile și mostenite:<br><br>1. nume: string<br>poate fi modificat din exterior<br><br>2. greutate: decimal<br>nu poate fi modificat din exterior<br><br>3. dimensiune: struct { lungime: decimal, latime: decimal, inaltime: decimal }<br>nu poate fi modificat din exterior<br><br>4. viteza: decimal - metri/sec<br>nu poate fi modificat din exterior<br><br>ii. câmpuri ascunse și mostenite:<br><br>1. stomac: array sau lista de tip Mancare<br>contine ce a mancat animalul<br><br>iii. Metode<br>1. constructor(nume, greutate, dimensiune, viteza)<br>primește atributele animalului și le seteaza corespunzător | **Cap 3 - Creating Types in C#**<br><br>• **Classes 73**<br>• **Inheritance 88**<br>• **The object Type 97**<br>• **Structs 101**<br>• **Access Modifiers 102**<br>• **Interfaces 104**<br>• **Enums 109**<br>• **Nested Types 113**<br>• **Generics 114**<br><br>**Cap 4 - Advanced C#**<br><br>• **Delegates 127**<br>• **Events 136**<br>• **Lambda Expressions 143**<br>• **Anonymous Methods 147**<br>• **try Statements and Exceptions 148**<br>• **Enumeration and Iterators 156**<br>• **Nullable Types 162**<br>• **Operator Overloading 168**<br>• **Extension Methods 171**<br>• **Anonymous Types 174**<br>• **Dynamic Binding 175**<br>• **Attributes 183**<br>• **Caller Info Attributes (C# 5) 185**<br>• **Unsafe Code and Pointers 187**<br>• **Preprocessor Directives 190**<br>• **XML Documentation 193**<br><br>**Cap 6 - Framework Fundamentals Capitolul nu este pus la intamplare, conține tipuri de date din .NET ce se regasesc in diferite funcționalități, tipuri ce vor fi folosite in continuare.**<br><br>• **String and Text Handling 213**<br>• **Dates and Times 226** |

2. void Mananca(Mancare m)
   metoda care primește o instanță de mancare, adaugă mancarea în stomac daca mancarea are greutate pana la 1 / 8 din greutatea animalului și afiseaza la consola " mananca " in acest caz

3. double Energie()
   metoda abstractă care întoarce nivelul de energie al animalului (procent)

4. void Alearga(decimal distanta)
   metoda care calculeaza si afișează in cat timp (secunde) parcurge animalul distanța specificata (in metri) dupa formula timp = distanta / (viteza / energie) " parcurge in secunde"

e. clasele Carnivor, Erbivor si Omnivor extind și implementează Animal.

   i. Carnivor
      1. double Energie()
         nivelul de energie se obține după formula: 0.2 - 1/5 * media greutate mancare + suma energie mancare

   ii. Erbivor
      1. double Energie()
         nivelul de energie se obține după formula: 0.5 + 1/3 * media greutate mancare + suma energie mancare

   iii. Omnivor
      1. double Energie()
         nivelul de energie se obține după formula: 0.35 + coef greutate * media greutate mancare + suma energie mancare (dacă mancare e planta coef greutate = 1/2 altfel este - 1/2)

**2. În proiectul creat anterior, scrie o clasă Program cu o funcție statica void Main care face următoarele:**

a. instantiaza un lup (carnivor), oaie (erbivor) si un urs (omnivor)

b. instantiaza o salata (planta), sunca (carne)

c. da-le sa manance animalelor în felul următor: lupul - 2x sunca, oaia - 3x salata, ursul - 1x sunca + 3x salata

d. pune animalele sa alerge 200 de metri

e. in clasa animal adauga un contor (un camp de tip int) care reține numărul de animale instantiate, contor ce poate fi accesat fără sa mă leg de o instanta anume si afiseaza la consola numărul de animale instantiate (ex. Animal.numar)

f. adaptează codul astfel incat un animal sa accepte doar tipul de mancare corespunzător tipului sau (animalele carnivore sa accepte doar carne,

erbivorele doar plante iar omnivorele ambele)

g. suprascrie metoda ToString() astfel incat atunci cand afisezi un animal în consola (Console.WriteLine(new Erbivor(...))) să afișeze următoarele pe linii separate:
   i. Tip animal: <tip>
   ii. Nume: <nume>
   iii. Greutate <greutate> kg
   iv. Dimensiuni <L x l x h>
   v. Viteza <viteza> m/s

h. în clasa Program adaugă următoarea metoda statică:

   i. Animal CreeazaAnimal(enum tip animal, nume, greutate, dimensiune, viteza)
   metoda instantiaza în funcție de tipul de animal primit clasa corespunzatoare cu restul de argumente primite enum TipAnimal { Lup, Urs, Oaie, Veverita, Pisica, Vaca }
   enum TipAnimal { Lup, Urs, Oaie, Veverita, Pisica, Vaca }

i. in functia Main:

   i. creeaza o lista goala de animale
   ii. creeaza o bucla de 10 iterații
   iii. în bucla trebuie sa:
      1. generezi un tip de animal random (folosește ce-ti ofera .net - Random)
      2. apelează funcția Program.Creează pentru a genera un animal (numele va fi "Animal ", greutate + dimensiune + viteza generează folosind random)
      3. adaugă animalul în lista de animale
      4. itereaza din nou prin lista de animale, verifica din ce familie face parte animalul si da-i 1xbucata de mancare corespunzătoare

j. Afișează următoarele statistici
   i. <numar animale> animale mancanca mancare
   ii. <numar animale> animale mancanca carne
   iii. <numar animale> animale mancanca plante

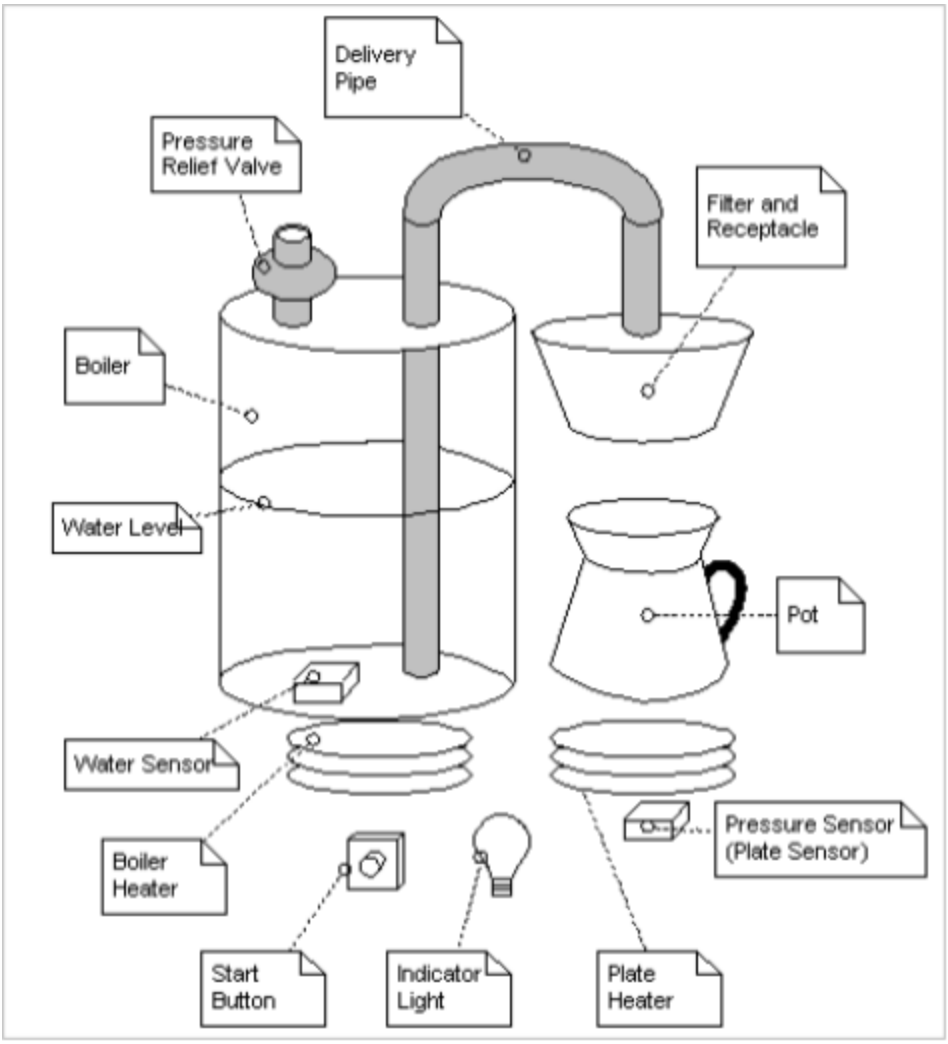
**Cap 14 - Concurrency and Asynchrony**

- **Introduction 563**
- **Threading 564**
- **Tasks 581**
- **Principles of Asynchrony 589**
- **Asynchronous Functions in C# 594**
- **Asynchronous Patterns 610**
- **Obsolete Patterns 618**

**Cap 19 - Reflection and Metadata**

- **Reflecting and Activating Types 790**
- **Reflecting and Invoking Members 797**
- **Reflecting Assemblies 810**
- **Working with Attributes 812**
- **Dynamic Code Generation 818**
- **Emitting Assemblies and Types 825**
- **Emitting Type Members 828**
- **Emitting Generic Methods and Types 834**
- **Awkward Emission Targets 836**

## 2.3. Your first program architecture - 10 zile




**Plecând de la poza de mai sus creeaza un proiect unde trebuie sa proiectezi un sistem/ierarhie de clase ce alcătuiesc un Espressor.**

**Folosește toate conceptele de POO cunoscute pentru a creea ierarhia cu următoarele considerente:**

**Trebuie sa existe o clasa principala numita Espressor care e compusă din diferite componente (ramane la atitudinea ta ce componente faci si folosesti) si care expune metode (user-friendly) publice prin care sa interactionezi cu tot sistemul (ex. scoate cana, adauga cana, adauga apa samd)**

**modul în care scrii componentele/sistemul trebuie sa fie cat mai aproape de limbajul natural**

**scheletul sistemului trebuie sa se apropie de cel al unui Expressor fizic**

**in entrypoint ul aplicației trebuie sa instantiezi Expresorul si sa expui o modalitate de interactionare cu userul la consola (trebuie expuse opțiunile intr-o bucla ce se termina atunci cand userul nu mai doreste nimic de la Expresor**

## 2.4. Delegates & Events - 5 zile

Creeaza un program pentru a tine evidenta unor produse de forma:

Considera urmatoarele clase:

Reducere:

- Nume - string
- Data: DateTime
- Aplica - delegat generic care primeste un Produs si nu intoarce nimic dar este important ca reducerea sa se aplice relativ la produs si nu global (adica indiferent de produs)

Producator:

- Nume: string
- Reduceri: List

Produs:

- Id: guid - identificare unica in sistem
- Nume: string
- Pret: Pret
- Stoc: intreg > 0
- Producator: Producator

Pret:

- Curs: static Dictionary<Moneda,decimal>
- Valoare: decimal
- Moneda: Moneda
- decimal ValoareCurs(Moneda moneda) - intoarce valoarea in functie de curs pentru moneda primita

enumeratie Moneda: LEU, EUR, USD

Catalog: List<Produs> (catalogul unui vanzator)

- PerioadaStart: DateTime de tip nullable
- PerioadaStop: DateTime de tip nullable
- Reduceri: List

Client:

- Inbox: string[10] (privat)
- Email: string
- Moneda: Moneda
- ProduseFavorite: List boolean
- Notifica(string mesaj)

- daca mesajul este mare de 60 de caractere intoarce false si nu adauga mesajul in inbox, altfel true si adauga in inbox
- daca numarul de mesaje depaseste numarul maxim (10) arunca o exceptie (OutOfMemoryException)

1. In metoda de intrare in program:
    a. instantiaza o lista de producatori, fiecare cu o lista de reduceri cu diferite perioade (considera si perioade fara unul din capete - doar start sau stop)
    b. instantiaza un catalog (refoloseste producatori din lista creeata anterior)
    c. stabileste cursul valutar din clasa Pret la cursul actual pentru cele 3 monede
    d. instantiaza o lista de clienti (proprietatea ProduseFavorite trebuie sa contina id-uri care se pot regasi in catalog sau nu)
    e. atentie: sunt recomandate listele de intializare pana si pentru reduceri care pot fi date ca liste de expresii lambda

2. In clasa Pret adauga 2 evenimente
    a. eveniment la care se pot intregistra subscriberi ce vor primi atunci cand se schimba pretul, vechiul pret si noul pret
    b. eveniment la care se pot intregistra subscriberi ce vor primi atunci cand se schimba stoc-ul,vechiul stoc si noul stoc

3. Scrie urmatoarele metode pentru clasa Catalog:
    a. o metoda publica prin care clientii se pot abona la respectiv-ul catalog
    atunci cand se aboneaza un client la catalog trebuie legat direct la evenimentul de schimbare a pretului pentru produsele care-l intereseaza printr-o expresie lambda intermediara care genereaza un mesaj de forma "Pret-ul produsului s-a schimbat de la la " (in functie de moneda clientului) si il trimite clientului

    b. o metoda publica prin care clientii se pot dezabona de la respectivul catalog (in cazul in care un client se dezaboneaza si nu se regaseste in lista de abonati, o exceptie trebuie aruncata)
    atunci cand un client se dezaboneaza trebuie scos de la notificari

    c. o metoda privata care este apelata cand se modifica stocul unui produs si notifica Clientii (daca poate daca nu trebuie sa prinda exceptia si sa treaca la urmatorul client) abonati prin metoda corespunzatoare cu mesajul "Produsul este din nou in stoc!" in urmatoarele conditii:
        i. daca stoc-ul este pe 0 si devine mai mare ca 0
        ii. daca id-ul produsului se regaseste printre favoritele clientului

    d. atentie: este o diferenta intre punctul a. si c. (extra: http://www.dofactory.com/net/observer-design-pattern)

4. Scrie o extensie pentru DateTime care permite sa verifici daca data se incadreaza intr-un range (2 DateTime-uri).
    a. itereaza prin lista de produse si pentru fiecare produs aplica toate reducerile producatorului care se incadreaza in perioada catalogului

b. indifierent de aplicarea sau nu a reducerii, produs-ul trebuie intors ca urmatorul element din iterator si d-abia dupa trebuie facuta trecerea la urmatoarea iteratie (hint: exista un keyword dedicat, metoda se va folosi impreuna cu metoda urmatoare si vrem sa sincronizam iteratiile astfel incat pentru fiecare produs sa aplicam mai intai reducerile producatorului dupa care reducerile vanzatorului inainte de a trece la urmatorul produs)

5. In clasa Catalog scrie o metoda privata AplicaReduceriProducator care:
   a. daca este specificat un delegat trebuie obtinuta reducerea specificata prin invocarea delegatului, altfel vom lua in calcul toate reducerile vanzatorului
   b. apeleaza metoda AplicaReduceriProducator iar pe colectia intoarsa, pentru fiecare element aplica reducerea sau reducerile vanzatorului in functie de ce s-a intamplat la pasul .a
   c. pentru produsele cu stoc-ul 0 si pretul redus sub 10 euro stocul va fi incrementat cu 100 de bucati

6. In clasa Catalog scrie o metoda AplicaReduceri care primeste optional un delegat generic ce permite selectarea unei reduceri din cele disponibile in catalog si trebuie sa faca urmatoarele:
   a. daca este specificat un delegat trebuie obtinuta reducerea specificata prin
   b. invocarea delegatului, altfel vom lua in calcul toate reducerile vanzatorului
   c. apeleaza metoda AplicaReduceriProducator iar pe colectia intoarsa, pentru fiecare element aplica reducerea sau reducerile vanzatorului in functie de ce s-a intamplat la pasul .a
   d. pentru produsele cu stoc-ul 0 si pretul redus sub 10 euro stocul va fi incrementat cu 100 de bucati

7. In metoda de intrare in program apeleaza metoda AplicaReduceri fara parametru cand aplicatia e compilata pentru deploy si cu o reducere selectata cand aplicatia e compilata pentru debug.

8. Itereaza pe lista de client si afiseaza informatii despre fiecare in felul urmator:
   a. Email <email client>
   b. Produse <nume produs 1>, <nume produs 2> etc.
   c. Inbox
      i. <mesaj 1>
      ii. <mesaj 2>
      iii. etc

# 3. MS SQL Fundamentals - 5 zile

## 3.1. Necesar:

**SQL Server express edition (2019 sau mai nou)**
**https://www.microsoft.com/en-us/sql-server/sql-server-downloads**

**SSMS - SQL Server Management Studio**

https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15

### 3.2. Resurse de învățare:

https://www.w3schools.com/sql/

### 3.3. Cunoștințe necesare la finalul cursului:

1. **Baze de date relationale**
2. **Creeare/manipulare baza de date la nivel basic:**
    a. **tabele**
    b. **Indecsi**
    c. **chei primare**
    d. **chei straine**
    e. **relatii intre tabele (1:1, 1:M, M:M)**

### 3.4. Probleme:

1. **Creeaza o baza de date cu tema la alegere care sa contina cel putin 4 tabele si cel putin cate o relatie 1:1, 1:M, M:M. Tabelele trebuie sa aiba toate constrângerile necesare (chei primare, indecsi, chei straine).**

2. **Populeaza cu date baza de date creata anterior.**

3. **Adauga la toate tabelele cate 2 coloane care sa stocheze pentru fiecare intrare data adaugarii si data ultimei modificari.**

4. **Plecand de la baza de date creata anterior, formuleaza minimum 2 enunturi de probleme de interogare a bazei de date si rezolva-le. Fiecare enunt formulat trebuie sa contina cel putin una dintre urmatoarele comenzi:**

    a. **Group By**
    b. **Join**
    c. **Having**

**La final va trebui sa aveți un singur fișier sql continand scriptul care va rula comenzile în ordinea formulării enunțurilor.**

## 4. ORM (Entity Framework Core) - 5 zile

### 4.1. Resurse de învățare:

https://docs.microsoft.com/en-us/ef/core/
https://www.entityframeworktutorial.net/efcore/entity-framework-core.aspx

### 4.2. Probleme:

1. **Creeaza o aplicație de consola in care sa integrezi entity framework core (ultima versiune stabilă) si creeaza utilizand metoda code-first baza de date pe care ai folosit-o la capitolul anterior.**
2. **Implementeaza un set de operatii CRUD peste baza de date prin care utilizatorul poate citi, adauga, modifica sau sterge date folosind aplicatia de consola. Aplicatia trebuie sa fie interactiva.**

## 5. ASP.NET MVC Core + Final Web Application - 25 zile

### 5.1. Resurse de invatare

https://docs.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-5.0

### 5.2. Proiect final

Folosind toate cunoștințele dobandite in capitolele anterioare, creeaza o aplicatie web utilizand tehnologia ASP.NET MVC Core. Tema propusa este o aplicatie de tip blog, insa poti propune si o alta tema.

Aplicația trebuie sa indeplineasca urmatoarele cerinte:

1. Sa folosesti cel putin 3 design patterns in aplicatie.
2. Codul structurat corect, scris curat, reutilizabil unde este cazul si identat
3. Trebuie sa functioneze fluent, fără erori netratate și fără puncte moarte de navigare prin interfața grafica.
4. Utilizati entity framework si metoda code-first pentru design-ul și crearea bazei de date.
5. Parte de identity si user management - Optional, nice to have
6. Toate datele care nu sunt statice in cadrul aplicatiei trebuie sa poata fi manipulate urilizand interfata grafica (modul de administrare pentru fiecare entitate din aplicatie)