Laborator – Moștenirea în C#

Obiective

După completarea acestui laborator veți dobândi următoarele cunoștințe:

- Definirea și folosirea interfețelor, claselor abstracte și a celor concrete
- Implementarea unei interfețe într-o clasă concretă
- Folosirea cuvintelor cheie **virtual** și **override**
- Definiți o clasă abstractă și să o folosiți în ierarhia de clase
- Creați clase sigilate pentru a preveni moștenirea

Condiţii prealabile

Înainte de a realiza acest laborator trebuie să fiţi familiarizaţi cu următoarele concepte:

- Crearea claselor în C#
- Definirea metodelor pentru clase

➢ Exerciţiul 1- Crearea unei clase ce implementează o structură de date de tip coadă

În acest exercițiu, veti scrie o clasă numită **EnumerableQueue** ce implementează o structură de tip coadă. În plus, va trebui ca această clasă să moștenească interfața **IEnumerable**, interfață ce permite instrucțiunii **foreach** să itereze pe membrii unei colecții.

Pentru a implementa o coadă:

- Definiți următorii membrii privați ai clasei **EnumerableQueue**:
  - protected object[] queueArray
  - protected int queueLength
  - protected int maxQueueLength
- Definiți doi constructori pentru clasa **EnumerableQueue**, primul ce nu primește nici un parametru și care inițializează variabilele, instanțiind vectorul **queueArray** cu dimensiunea 50; al doilea primește ca parametru un întreg, specificând dimensiunea vectorului
- Adăugați membru o metodă publică, virtuală, **Enqueue** ce primește ca parametru un obiect pe care îl adaugă cozii dacă mai este loc. Dacă nu, se aruncă o excepție de tipul **InvalidOperationException**
- Adăugați apoi metoda publică, virtuală **Dequeue** ce scoate primul element din coadă și îl trimite ca valoare întoarsă; dacă este vidă coada, se aruncă o exceptie de tipul **InvalidOperationException**

Pentru a implementa interfața IEnumerable:

- Declarați o metodă publică **GetEnumerator()** ce întoarce un obiect de tipul **IEnumerator**. Această metodă face parte din interfața **IEnumerator** și trebuie implementată de către clasa **EnumerableQueue**.
- După cum observați, această metodă trebuie să întoarcă un obiect de tip **IEnumerator.** Pentru a putea face acest lucru, va trebui întâi scrisă o clasă care implementează această interfață. Declarați clasa **QueueEnumerator** ce implementează interfața **IEnumerator.** Clasa poate fi îmbricată și privată, accesibilă doar clasei **EnumerableQueue**.
- Constructorului clasei **QueueEnumerator** trimiteți-i vectorul de obiecte din coadă și numărul de obiecte din vector. Va trebui să implementați metodele interfeței, **MoveNext** și **Reset,** precum și proprietatea readonly **Current**. Pentru a declara o proprietate de tipul obiect readonly ce întoarce un obiect dintr-un vector, folosiți următorul cod:

```
public object Current
{
   get { return queue[index]; }
}
```

- Creați o instanță a clasei **QueueEnumerator** pe care o trimiteți ca valoare de întoarcere metodei **GetEnumerator**. Este nevoie de cast explicit la această operație?

Testarea clasei pe care ați scris-o:

- Definiți o altă clasă, de test, în care scrieți o metodă Main. Aici, creați un obiect de tipul **EnumerableQueue** și introduceți în el o serie de numere întregi (remember boxing?)
- Folosiți instrucțiunea **foreach** pentru a parcurge toate elementele din coadă. Afișați-le.
- Salvați ce ați lucrat
- Compilați și rulați programul