Дизајн класе

Windows Runtime и .NET Framework садрже огроман број класа.
Неке од њих су већ коришћене попут Console.
Класе омогућавају механизам за моделовање ентитета који се манипулишу са апликацијама.
Ентитет може бити одређена ствар (купац) или нека апстракција (трансакција).
Део процеса дизајнирања система је одређивање енитета који су важни за процесе које систем имплементира.
Затим зе врши анализа у потрази за информацијама које ови ентитети садрже и за операцијама које треба да изврше.
Чувају се информације које класа садржи у виду поља (fields) и користе методе за реализацију операција које класа изводи.

Разумевање класификације

Класа је коренска реч израза класификација.
Када се дизајнира класа, систематски се уређују информације и понашања у потребан ентитет и то је део процеса класификације.
Пример: сви аутомобили имају иста понашања (изводе кочење, зауставе се, убрзавају...) и заједничке атрибуте (имају кочнице, волан, мотор...).
Људи користе реч ауто да опишу један објекат који дели ова заједничка понашања и атрибуте.
Без класификације неке ствари које се подразумевају би биле нејасне и неодређене.
ООП се покушавају класификовати различити концепти који су део проблема и решења проблема преко моделовања класа помоћу програмског језика.

Смисао енкапсулације

Енкапсулација је битна приликом дефинисања класе.
Идеја је да програм који користи класу не мора познавати како класа интерно функционише; програм само креира инстанце класе и позива методе те класе.
Ако методе раде оно што је намењено, програм не мора да зна како се то и реализује.
Класа мора да одржава различите врсте информација о унутрашњим стањима да би реализовала разне методе.
Ове додатне информације о стањима и активностима су сакривене од програма који користи класу.
Зато се понекад енкапсулација назива и сакривање информација.
Енкапсулација има два смисла:

- Комбиновање метода и података унутар класе (подржавање класификације)
- Контролисање приступа методама и подацима (контрола употребе класе)

Дефинисање и употреба класе

Најједноставнија декларација класе:

```
class Krug {}
```

Заглавље декларације класе садржи службену реч class и идентификатор имена класе.
Са службеном речи class се дефинише нова класа.
После службене речи следи идентификатор класе којим се даје име класи и са којим се приступа садржају класе.

Подаци и методе класе се налазе унутар тела класе, између витичастих заграда.
Дати пример не приказује реалну ситуацију у коду јер примеру недостаје структура кода програмског језика C# која мора постојати да би се подржала било која креирана класа.
Реалан пример унутар којег је подржана креирана класа:

```
using System;
namespace Proba
{
    class Program
    {
        static void Main() {}
    }
    class Krug {}
}
```

Када се стартује овакав код не добија се никаква корисна информација, али је структурно ово целовит код и нема повратне информације о некаквој грешци у структури.
Битно је да постоји основна класа унутар које мора бити Main функција/метода а коју ћемо користити да искористимо нове класе и методе креиране изван основне класе (Program) али унутар истог именског простора.
Често се једна класа смешта у један фајл и том фајлу се додељује име класе.

Декларација промењиве типа класе

После декларације класе, могуће је декларисати промењиву тог типа:

```
using System;
namespace Proba
{
    class Program
    {
        static void Main()
        {
            Krug k;
        }
    }
    class Krug {}
}
```

Могуће је и дефинисати методу која узима као параметар нов тип класе:

```
using System;
namespace Proba
{
    class Program
    {
        static void Main()
        {
            Krug k;
        }
        static void Metoda(Krug x) {}
    }
    class Krug {}
}
```

Опциони делови дефиниције класе

У заглављу класе пре службене речи class могу бити атрибути и модификатори класе.
Модификатори класе су: public, internal, abstract, sealed, static, unsafe и partial.
После идентификатора имена класе се могу користити генерички типови параметара и ограничавача, базичне класе и интерфејси.
Унутар витичастих заграда се налазе чланови класе попут methods, properties, events, fields, constructora, overloaded operators, nested types, finalizer.

Приступ декларисаној класи штампањем садржаја

Када би се покушало приступити декларисаној класи:

```
Console.WriteLine(Krug);
```

добија се порука о грешци:
'Krug' is a type, which is not valid in the given context
што је доказ да декларисана класа представља нови тип података а не промењиву.

Задаци за самосталан рад

1. Написати програм којим се декларише класа Ljudi.

2. Написати програм којим се декларише класа Zivotinje а затим декларише промењива тог типа података.

3. Написати програм којим се декларише класа Vozila и дефинише метода Ubrzavanje() која користи нову класу као параметар.

4. Написати програм који декларише класе Ljudi и Zivotinje а затим декларише две промењиве ових типова података.

5. Доказати да декларисана класа Ljudi не представља промењиву.