Енумерације

Пример: Написати програм којим се дани у седмици помоћу енумерација представљају као цели бројеви.

```
using System;
namespace Proba
{
    internal class Program
    {
        enum Dani { PON, UTO, SRE, CET, PET, SUB, NED };
        static void Main(string[] args)
        {
            int sedmicaStart = (int)Dani.PON;
            int sedmicaKraj = (int)Dani.NED;
            Console.WriteLine($"Ponedeljak: {sedmicaStart}");
            Console.WriteLine($"Nedelja: {sedmicaKraj}");
        }
    }
}
```

Пример: упоредити смерове света кроз енумерацију као бајтове и као стрингове

```
using System;
namespace Proba
{
    namespace Novo
    {
        enum orijentacija
        {
            sever = 1,
            jug = 2,
            istok = 3,
            zapad = 4
        }
        internal class Program
        {
            static void Main(string[] args)
            {
                byte smerBajt;
                string smerString;
                orijentacija mojSmer = orijentacija.sever;
                Console.WriteLine($"Okrenuo sam se na: {mojSmer}");
                smerBajt = (byte)mojSmer;
                smerString = Convert.ToString(mojSmer);
                Console.WriteLine($"ekvivalentno u bajtovima:
                          {smerBajt}");
                Console.WriteLine($"ekvivalentno u stringovima:
                          {smerString}");
```

```
            }
        }
    }
}
```

Пример: додати следећи код у Main методу

```
byte smerBajt;
string smerString;
orijentacija mojSmer = orijentacija.sever;
Console.WriteLine($"Okrenuo sam se na: {mojSmer}");
smerBajt = (byte)mojSmer;
smerString = Convert.ToString(mojSmer);
Console.WriteLine($"ekvivalentno u bajtovima: {smerBajt}");
Console.WriteLine($"ekvivalentno u stringovima: {smerString}");
```

Даје:
Okrenuo sam se na: sever
ekvivalentno u bajtovima: 1
ekvivalentno u stringovima: sever

Види се да је код дефинисао енумерациони тип orijentacija и да је дефиниција смештена у другачији именски простор Novo.
Ово је могуће извести пошто се дефиниције не извршавају, тј током времена извршавања програма (at runtime) не пролази се кроз код унутар дефиниције као кроз код апликације.
Извршавање апликације почиње на уобичајен начин и постоји приступ новом типу података пошто се и он налази у истом именском простору.
У додатном коду се извршава конверзија енумерационих вредности у друге типове.
Овде се мора извршити експлицитна конверзија, иако је подтип orjentacije тип byte, и даље се мора кастовати тип byte за конверзију вредности mojSmer у byte тип:

```
smerBajt = (byte)mojSmer;
```

За добијање стринг вредности од енумерационе вредности:

```
smerString = Convert.ToString(mojSmer);
```

За конверзију стринг вредности у енумерациону вредност користи се команда Enum.Parse():

```
(tipEnumeracije)Enum.Parse(typeof(tipEnumeracije), vrednostEnumeracije - String);
```

Оператор typeof враћа тип његовог операнда.

Задаци за самосталан рад

1. Написати програм који коришћењем енумерација представља дане у седмици као бројеве (понедељак 1, уторак 2, среда 3, четвртак 10, петак 11, субота 12, недеља 20).

2. Коришћењем енумерација и switch структуре, написати програм којим корисник уноси свој ниво познавања програмског језика C# у бројчаном облику (1, 2, 3) па се на конзоли исписује одговарајућа порука (nizak, srednji, visoki) типа стринга.

3. Коришћењем енумерација и if - else if - else структуре, написати програм којим корисник уноси свој ниво познавања програмског језика C# (nizak, srednji, visoki) па се на конзоли исписује одговарајућа порука и ниво у бројчаном облику.