Készítsen programot az alábbi feladatra az Ön által tanult programozási nyelven! Az elkészítendő programnak egy DHCP szerver működését kell szimulálnia. A DHCP szerver a 192.168.10.100 – 192.168.10.199-as tartományból osztja az IP-címeket. A feladathoz négy induló állomány tartozik:

- *excluded.csv*: azon IP-címek listája, amelyeket a DHCP szervernek nem szabad kiosztania.
- *reserved.csv*: pontosvesszővel elválasztva tartalmaz MAC cím – IP-cím párokat. A DHCP szerverhez érkező kérés esetén a listában szereplő MAC címhez a hozzá párosított IP-címet kell kiosztani. A listában szereplő IP-címeket más MAC cím nem kaphatja meg.
- *dhcp.csv*: a DHCP szerver aktuális állapotát (bérelt címeket) tartalmazza. A fájlban MAC cím – IP-cím párok vannak, melyik MAC címhez milyen IP-címet osztott ki és tart fent a DHCP szerver.
- *test.csv*: a teszteléshez szükséges állomány: műveletek és azok paramétere szerepelnek benne. Lehetséges műveletek: ○ *request*, paramétere MAC cím. Ebben az esetben a MAC címhez próbál IP- címet rendelni a szerver a következő oldalon látható folyamatábra alapján. Ha az IP-cím kiosztása sikertelen, akkor dobjon saját kivételt a program.
    - ○ *release*: paramétere IP-cím. Ebben az esetben fel kell szabadítani az IP- címet.

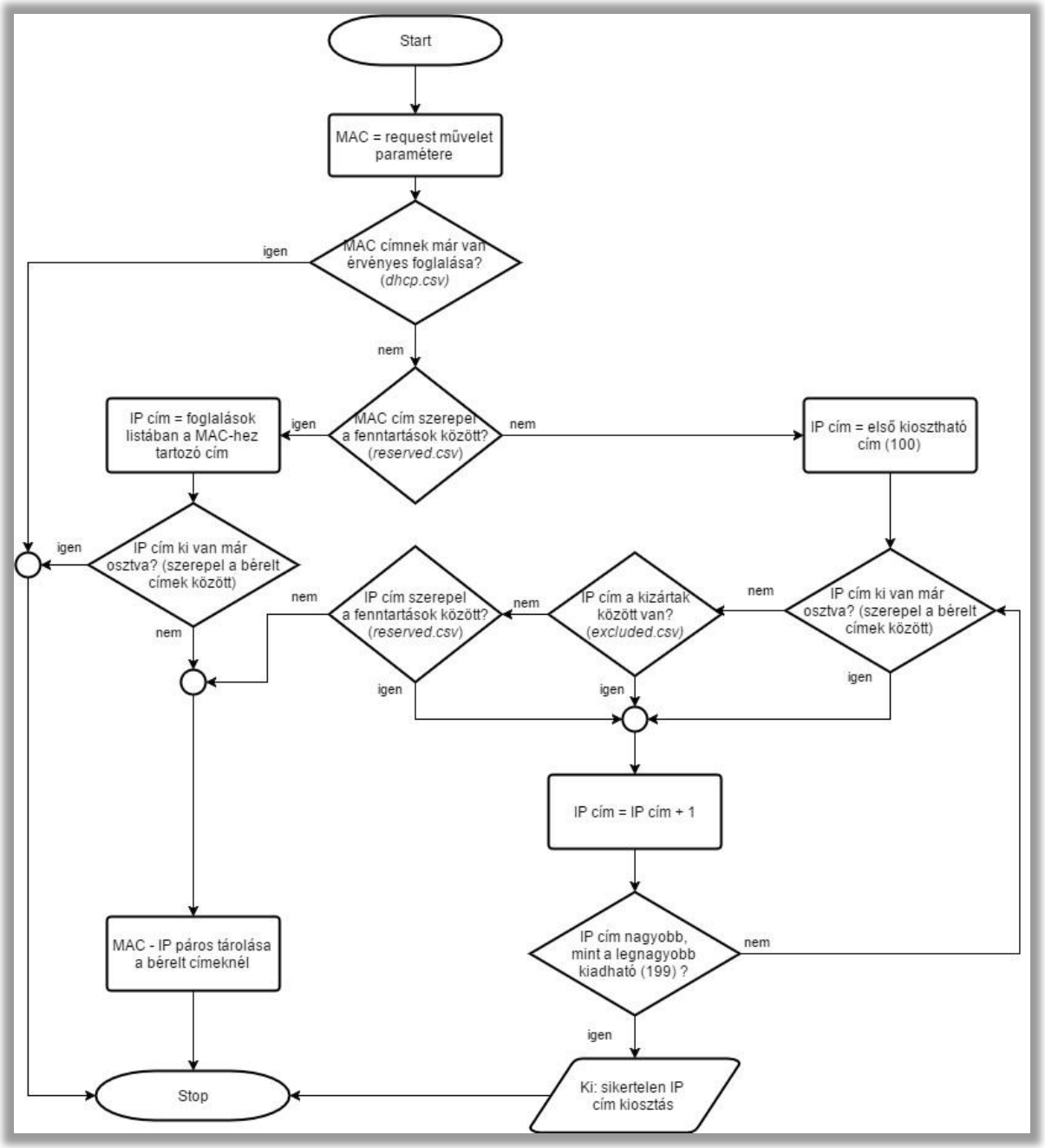
Feladat:

1. **Olvassa be a négy állomány tartalmát, tárolja el a memóriában!**
2. **Hajtsa végre soronként a test.csv állományban lévő műveleteket! A *request* műveletet a következő oldalon található folyamatábra szerint kódolja!**
3. **Írja ki a DHCP szerver állapotát a *dhcp_kesz.csv* állományba a műveletek végrehajtása után!**

Minta:

```
767CF59F54C3;192.168.10.102
8567CBE24317;192.168.10.103
F4397EDE572C;192.168.10.104
0AEC090BDA48;192.168.10.105
18F307FF9F93;192.168.10.108
C6150FBCB564;192.168.10.109
56EC1F4F3529;192.168.10.126
63E614BEB096;192.168.10.110
E31444BB981A;192.168.10.106
3EDF5045DFEF;192.168.10.112
1B6387D6BE8D;192.168.10.111
4DE2F5192E12;192.168.10.116
2891F93D50AA;192.168.10.117
```

Request művelet folyamatábrája:



Megoldás:

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

namespace DHCP
{
    class Program
    {
        static List<int> Excluded;//nem kiosztható IP címek
        static Dictionary<string, int> Reserved;//fentartott címek (MAC;IP csak az utolsó
oktettet tárolom)
        static Dictionary<string, int> Dhcp;//kiosztott címek (MAC;IP csak az utolsó
oktettet tárolom)
        static string halozatcim = "192.168.10.";
        static int elso=100;//kiosztható első cím
        static int utolso=199;//kiosztható utolsó cím
        static void Main(string[] args)
        {
            int i;
            Excluded = new List<int>();//nem kiosztható IP címek listája
```

```
            Reserved = new Dictionary<string, int>();//fentartott címek listája
            Dhcp = new Dictionary<string, int>();//kiosztott címek listája
            foreach (string sor in File.ReadAllLines("excluded.csv", Encoding.UTF8))
            {
                Excluded.Add(int.Parse(sor.Split('.')[3]));//csak az utolsó oktettet
tárolom
            }
            //for (i = 0; i < Excluded.Count; i++) Console.WriteLine("{0} ",Excluded[i]);
            foreach (string sor in File.ReadAllLines("reserved.csv", Encoding.UTF8))
            {
                Reserved.Add(sor.Split(';')[0], int.Parse(sor.Split(';', '.')[4]));
            }
            //foreach(var d in Reserved)
Console.WriteLine("{0};{1}{2}",d.Key,halozatcim,d.Value);//fentartott IP címek
            foreach (string sor in File.ReadAllLines("dhcp.csv", Encoding.UTF8))
            {
                Dhcp.Add(sor.Split(';')[0], int.Parse(sor.Split(';', '.')[4]));
            }
            //foreach (var d in Dhcp) Console.WriteLine("{0};{1}{2}", d.Key, halozatcim,
d.Value);//kiosztott IP címek

            foreach (string sor in File.ReadAllLines("test.csv", Encoding.UTF8))
            {
                string muvelet = sor.Split(';')[0];//ebben az esetben a MAC címhez próbál
IP címet rendelni
                if(muvelet=="request")
                {
                    string Maccim = sor.Split(';')[1];
                    if(!Dhcp.ContainsKey(Maccim))//ha a MAC cím nem szerepel a kiosztott IP
címek között
                    {
                        if(Reserved.ContainsKey(Maccim))//ha a fenntartott címekben
szerepel a MAC cím
                        {
                            int utolsooktett = Reserved[Maccim];
                            if(!Dhcp.ContainsValue(utolsooktett))
                            {
                                Dhcp.Add(Maccim, utolsooktett);
                            }
                        }
                        else
                        {
                            int utolsooktett = elso;//megpróbáljuk először az első címet
kiosztani
                            while(utolsooktett<=utolso)//ciklus amíg el nem érem az utolsó
címet
                            {
                                if(!Dhcp.ContainsValue(utolsooktett))//ha nem szerepel a
kiosztott címek között
                                {
                                    if(!Excluded.Contains(utolsooktett))//ha nem szerepel a
kizárt címek között
                                    {
                                        if(!Reserved.ContainsValue(utolsooktett))//ha nem
szerepel a foglalt címek között
                                        {
                                            Dhcp.Add(Maccim, utolsooktett);//lefoglaljuk az
aktuális MAC címhez az IP címet
                                            break;
                                        }
                                    }
                                }
                                utolsooktett++;//veszem a következő IP címet
                            }
                            if (utolsooktett > utolso)//ha nem sikerült IP címet kiosztani
                                throw (new Exception("Nincs kiadható IP cím!"));
                        }
                    }
                }
```

```
                else if(muvelet=="release")
                {
                    int utolsooktett = int.Parse(sor.Split(';', '.')[4]);
                    if(Dhcp.ContainsValue(utolsooktett))
                    {
                        Dhcp.Remove(Dhcp.First(x => x.Value == utolsooktett).Key);//IP cím
felszabadítása
                    }
                }
            }
            //kiírás fájlba
            StreamWriter fajlbairo = new StreamWriter("dhcp_kesz.csv", false,
Encoding.UTF8);
            foreach(var d in Dhcp)
            {
                fajlbairo.WriteLine("{0};{1}{2}",d.Key,halozatcim,d.Value);
            }
            fajlbairo.Close();


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