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

namespace TrashRecycling
{
        class DegradationTimeException : Exception
        {
        }
}
```

---

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

namespace TrashRecycling
{
        interface IBurnable
        {
        float Burn();
        }
}
```

---

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

namespace TrashRecycling
{
        interface IRecyclable
        {
        void Recycle(byte p);
        }
}
```

---

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

namespace TrashRecycling
{
	class Landfill
	{
	List<Trash> trashList = new List<Trash>();

	public void Add(Trash newTrash)
	{
	if (newTrash != null)
		trashList.Add(newTrash);
	}

	public Trash this[int index]
	{
	get
	{
		return trashList[index];
	}
	}

	public int Count
	{
	get { return trashList.Count; }
	}
	}
}
```

---

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

namespace TrashRecycling
{
	class PercentageException:Exception
	{
	}
```

```
}

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

namespace TrashRecycling
{
	class Program
	{
	static void Main(string[] args)
	{
	Landfill landfill = new Landfill();

	try
	{
		StreamReader reader = new StreamReader("data.txt");
		while (!reader.EndOfStream)
		{
		string[] szavak = reader.ReadLine().Split(';');
		switch (szavak[0])
		{
		case "recyclable":
			landfill.Add(new RecycableTrash(szavak[1], float.Parse(szavak[2]),
float.Parse(szavak[3]), byte.Parse(szavak[4])));
			// PARSEOLÁSOK
			break;
		case "burnable":
			landfill.Add(new RegularTrash(szavak[1], float.Parse(szavak[2]),
float.Parse(szavak[3]), float.Parse(szavak[4])));
			// PARSEOLÁSOK
			break;
		}
		}
		reader.Close();
	}
	catch (ZeroWeightException) { Console.WriteLine("Nulla a szemét tömege, ami
nagyon jó!"); }
	catch (DegradationTimeException) { Console.WriteLine("A lebomlási idő nem lehet
negatív!"); }
	catch (PercentageException) { Console.WriteLine("A százalék értéke csak 0 és 100
között lehet!"); }
```

```
        catch (Exception e) { Console.WriteLine("Egyéb hiba:\n{0}", e.StackTrace); }

        float totalweight = 0;
        for (int i = 0; i < landfill.Count; i++)
        {
                totalweight += landfill[i].Weight;
        }
        for (int i = 0; i < landfill.Count; i++)
        {
                if (landfill[i] is RecycableTrash)
                {
                (landfill[i] as RecycableTrash).Recycle(20);
                }
        }
        // RECYCLING PÁR ALKALOMMAL
        float max = -1;
        int counter = 0;
        for (int i = 0; i < landfill.Count; i++)
        {
                if (landfill[i].DegradationTime > max)
                {
                max = landfill[i].DegradationTime;
                counter = i;
                }
        }
        Console.WriteLine("A leghosszabb lebomlási idővel az alábbi hulladék
rendelkezik:\n{0}", landfill[counter]);
        float burntTrash = 0;
        for (int i = 0; i < landfill.Count; i++)
        {
                if (landfill[i] is RegularTrash)
                {
                RegularTrash a = (landfill[i] as RegularTrash);
                burntTrash += a.Burn();
                }
        }
        Console.WriteLine("Összesen {0} tonna CO2 szabadul fel, ha az összes szemetet
elégetjük.\n", burntTrash);

        Console.WriteLine("A hulladék összmennyisége az újrafelhasználás és égetés előtt:
{0} tonna\n", totalweight);

        float weightAfterBurn = 0;
        for (int i = 0; i < landfill.Count; i++)
        {
        weightAfterBurn += landfill[i].Weight;
```

```
        }
        Console.WriteLine("A hulladék mennyisége újrafelhasználás és égetés után: {0}
tonna\n", weightAfterBurn);
        // KIIRATÁSOK
        Console.ReadKey();
        }
        }
}

---------------------------------------
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TrashRecycling
{
        class RecycableTrash:Trash, IRecyclable
        {
        byte airPollutionDecrease;
        public byte AirPollutionDecrease
        {
        get { return airPollutionDecrease; }
        private set
        {
                if (value < 0 || value > 100)
                throw new PercentageException();
                airPollutionDecrease = value;
        }
        }

        public void Recycle(byte p)
        {
        if (Weight == 0)
                throw new ZeroWeightException();
        Weight = p * 0.01f*Weight;
        }

        public RecycableTrash(string name, float weight, float degradationTime, byte
airPollutionDecrease):base(name,weight,degradationTime)
        {
        AirPollutionDecrease = airPollutionDecrease;
        }

        public override string ToString()
```

```
        {
        return string.Format("Hulladék neve: {0}\nHulladék tömege: {1}t\nHulladék lebomlási ideje: {2}év\nLégszennyezés csökkentés újrahasznosítással: {3}%\n", Name, Weight, DegradationTime, AirPollutionDecrease);
        }
        }
}
----------------------------
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TrashRecycling
{
        class RegularTrash:Trash, IBurnable
        {
        float carbonicDioxid;
        public float CarbonicDioxid
        {
        get { return carbonicDioxid; }
        private set
        {
                if (value <= 0)
                throw new Exception();
                carbonicDioxid = value;
        }
        }

        public float Burn()
        {
                float burn = Weight * CarbonicDioxid;
                this.weight=0;
                return burn;
        }

        public RegularTrash(string name, float weight, float degradationTime, float carbonicDioxid)
        : base(name,weight,degradationTime)
        {
        CarbonicDioxid = carbonicDioxid;
        }

        public override string ToString()
        {
```

```
        return string.Format("Hulladék neve: {0}\nHulladék tömege: {1}t\nHulladék lebomlási ideje: {2}év\n1 tonna szemét elégéséből származó CO2 mennyiség: {3}\n", Name, Weight, DegradationTime, CarbonicDioxid);
        }
        }
}
-----------------------------------------------
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TrashRecycling
{
        abstract class Trash
        {
        string name;
        public string Name
        {
        get { return name; }
        private set
        {
                if (value == null)
                throw new NullReferenceException();
                name = value;
        }
        }

        protected float weight;
        public float Weight
        {
        get { return weight; }
        protected set
        {
                if (value < 0 || value > weight)
                throw new ZeroWeightException();
                weight -= value;
        }
        }

        private float degradationTime;

        public float DegradationTime
        {
        get { return degradationTime; }
```

```
        private set
        {
                if (value < 0)
                throw new DegradationTimeException();
                degradationTime = value;
        }
        }

        public Trash(string name, float weight, float degradationTime)
        {
        Name = name;
        this.weight = weight;
        DegradationTime = degradationTime;
        }

        }
}
```

---

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

namespace TrashRecycling
{
        class ZeroWeightException:Exception
        {
        }
}
```

---

TXT

```
recyclable;papír;14;0,15;40
recyclable;pamutruha;8,3;4;65
burnable;pelenka;3,2;80;0,3
recyclable;nejlonzacskó;21,1;17;23
burnable;olajos textil;1,2;7;36
recyclable;PET palack;76;1000000;44
```