Atleast -- in each same round

Question

```
1 nicky 10
1 Smithy 8
1 Smithy 2
2 nicky 10
2 Smithy 10
```

```
3 nicky 10
3 Smithy 4
3 Smithy 6
```

Task:

Satisfactory: How many strikes (shots of 10 points) are scored after the first strike of Smithy? You have to define a method for reading from the file and you have to create an algorithm which is based on the general algorithm of the algorithmic patterns.

In the sample file, the answer is 1.

Excellent: Was there strike in each round? You have to create a class with methods first(), next(), current(), and end() for enumerating for each round if there was a strike in it. Other public methods cannot be created for this class.

In the sample file the output is yes.

Code

Program.cs

```
using System;
using TextFile;
```

```
namespace test1_sample2
{
    internal class Program
    {
        static void Main(string[] args)
        {
            bool fileError = true;
            do
            {
                try
                {
                    Console.WriteLine("Enter file name: ");
                    string name = Console.ReadLine();
                    Infile t = new Infile(name);

                    t.First();
                    int count = 0;
                    int round = 0;
                    while (!t.End())
                    {
                        round = int.Parse(t.Current().round);
                        t.Next();
                    }
                    //Console.WriteLine($"{round}");
                    //Console.WriteLine($"{count}");
                    if (t.Current().count ⩾ round)
                    {
                        Console.WriteLine(t.Current());
                    }
                    else
                    {
                        Console.Write("no");
                    }
                    fileError = false;
                }
                catch (FileNotFoundException)
                {
                    Console.WriteLine("File not found. Trye again.\n\n");
                }
            }
            while (fileError);
        }
    }
}
```

Infile.cs

```
using System;
```

```
using test1_sample2;
using TextFile;

namespace test1_sample2
{
    public class Game //read()
    {
        public string round;
        public string name;
        public int score;

        Game(string round, string name, int score)
        {
            this.round = round;
            this.name = name;
            this.score = score;
        }

        public static void Read(ref Infile.Status st, ref Game e, ref
TextFileReader x)
        {
            e = new Game("", "", 0);
            x.ReadString(out e.round);
            x.ReadString(out e.name);
            bool l = x.ReadInt(out e.score);
            st = l ? Infile.Status.norm : Infile.Status.abnorm;
        }

    }

    public class Score // ToString()
    {
        public string round;
        public int number;
        public int count;
        public override string ToString()
        {
            return "yes";
        }
    }


    public class Infile
    {
        public enum Status { abnorm, norm };

        private Status st;
        private Game e;
        private TextFileReader x;
        private Score curr;
```

```
        private bool end;
        //private int count = 0;
        //private bool strike = false;

        public Infile(string name)
        {
            x = new TextFileReader(name);
            curr = new Score();
        }

        public void First()
        {
            Game.Read(ref st, ref e, ref x);
            Next();
        }

        public void Next()
        {
            end = st == Status.abnorm;
            if (!end)
            {
                curr.round = e.round;
                curr.number = 0;
                while(st == Status.norm && e.round == curr.round)
                {
                    curr.number = e.score;
                    if (curr.number == 10)
                    {
                        curr.count += 1;
                    }
                    Game.Read(ref st, ref e, ref x);
                }
            }
        }

        public Score Current()
        {
            return curr;
        }

        public bool End()
        {
            return end;
        }
    }
}


/*
curr.number = e.score;
```

```
if (curr.number == 0)
{
    //st = Status.abnorm;
    end = true;
    Console.WriteLine("no");
    break;
}
else
{
    Game.Read(ref st, ref e, ref x);
}
*/
```