Highest -- in all rounds

Question

One sample file:

```
AAAAAA 1
AAAAAA 5
AAAAAB 4
AAAAAC 3
```

Task:

Satisfactory: How many Excellent (5) grades there were after the first fail (1)? 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: What is the highest number of attempts that a student has taken? You have to create a class with methods first(), next(), current(), and end() for enumerating the number of attempts for each student. Other public methods cannot be created for this class.

In the sample file AAAAAA had 2 attempts, the other students had 1, so the answer is 2.

Code

Program.cs

```
using System.IO.Enumeration;
using System;
using TextFile;

namespace test1_sample4
{
    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 max = 0;
                    while (!t.End())
                    {
                        if (t.Current().attempt > max)
                        {
                            max = t.Current().attempt;
                        }
                        t.Next();
                    }
                    Console.WriteLine($"{max}");
                    fileError = false;
                }
                catch(FileNotFoundException)
                {
                    Console.WriteLine("file not found\n\n");
                }
            }
            while (fileError);
        }
    }
}
```

Infile.cs

```
using System;
using test1_sample4;
using TextFile;

namespace test1_sample4
{

    public class Result
    {
        public string code;
        public int grade;

        Result(string code, int grade)
        {
            this.code = code;
            this.grade = grade;
        }
```

```
        public static void Read(ref Infile.Status st, ref Result e, ref
TextFileReader x)
        {
            e = new Result("", 0);
            x.ReadString(out e.code);
            bool l = x.ReadInt(out e.grade);
            st = l ? Infile.Status.norm : Infile.Status.abnorm;
        }
    }

    public class Attempt
    {
        public string code;
        public int attempt;
    }
    public class Infile
    {
        public enum Status {abnorm, norm};

        private Status st;
        private Result e;
        private TextFileReader x;
        private Attempt curr;
        private bool end;

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

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

        public void Next()
        {
            end = st == Status.abnorm;
            if (!end)
            {
                curr.code = e.code;
                curr.attempt = 0;
                while(st == Status.norm && e.code == curr.code)
                {
                    curr.attempt +=1;
                    Result.Read(ref st, ref e, ref x);
                }
            }
```

```
        }

        public Attempt Current()
        {
            return curr;
        }

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