AARHUS
UNIVERSITET
INGENIØRHØJSKOLEN

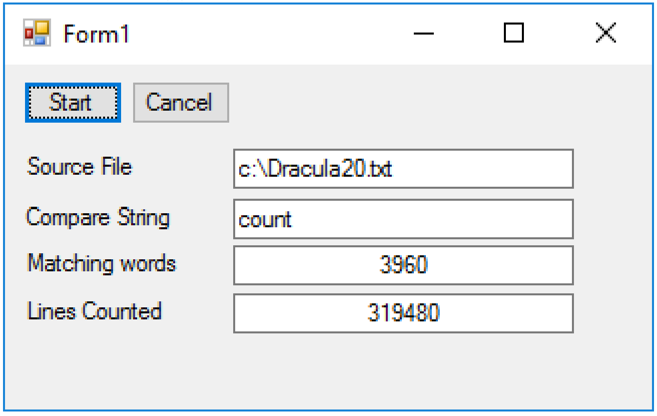
Exercise: Threading and Windows Forms

In these exercises, you will work with C# BackgroundWorker in Windows Forms to create software with concurrency.

Exercise 1:
Go to the web page: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/threading/walkthrough-multithreading-with-the-backgroundworker-component
And complete the walkthrough on the page.

A text file with the book Dracula is available on Blackboard. The text of the book is repeated 20 times, so you will have a chance to use the Cancel button.

Your finished application should look somewhat like this:



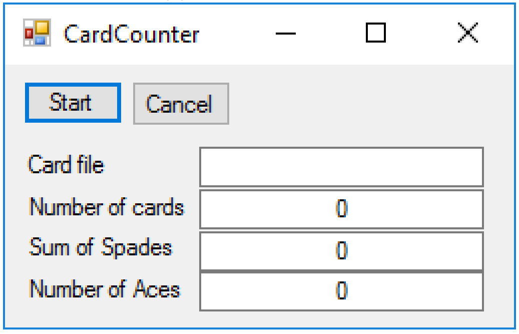
Source File	c:\Dracula20.txt
Compare String	count
Matching words	3960
Lines Counted	319480

Exercise 2:
A .zip file with 3 files containing card tuples are provided on Blackboard.
Each tuple is a line in the file in the form “SPADE, 10” or “HEART, 8”.
Pick **one** file to use for this exercise.

Write a Card Counting Windows Forms application, with a BackgroundWorker, which answers the following questions:

- How many cards are there?
- What is the total sum of all SPADES?
- How many aces (1’s) are there?

Your finished application should look somewhat like this:



Card file	
Number of cards	0
Sum of Spades	0
Number of Aces	0

Exercise 3:
Let the user select the source file, by adding an OpenFileDialog to the word count application.

Exercise 4:
Modify your card counting application, so you can select all three files.
Use a single BackgroundWorker to process the files one at the time.

Exercise 5:
Modify your card counting application, so all files are processed in parallel, i.e. each file is processed by a separate BackgroundWorker.
Verify that the count is still correct.

Exercise 6:
Modify your card counting application, so you can select a folder and process all .txt files in the folder.

Advanced exercise 1:
It has become increasingly difficult to sell the console chat program you wrote in the previous exercise.
Youngsters have stopped using the console… they want nice and shiny Windows Forms programs instead.
And all the old people have either bought your program or are using Linux.

So…
Your mission, *should you choose to accept it*, is to create a Windows Forms chat program.
When your program starts, it shall connect to a chat program on one of your fellow students computers.
The text you enter shall be sent to the other computer and displayed in a nice chat window.
Likewise, text entered on the other computer shall be sent to your computer and displayed in a nice chat window.

Hint: You will probably still need a *TCPListener* and *TCPClient* for your application. And multiple threads of course.