ReadMe file -> Preprocess application

We are going to create a command line application. The number of the application is: preprocess.c

Put at the top the application all details about the creator.

Empty line: line that contains less than 1 character (no spaces, tab);
We are going to consider that comment just the lines with // and ending at the end of the line. In this coursework we are not going to consider /* ...APPLICATION. */

Preprocess applications produce an output file which has the same name as the .c input but with a .o file extension.
We have to read from a file and:
This output file contains a preprocesses version of the input file contaning the same line of program code but modified according to the following specification:

-Print out the **number of non-empty lines** (lines that contains at least 1 character that is different from newline, space, or tab)
-A line that begins with #include "filename.h" should be replaced with the entire contents of the file called filename.h (our application should remote the quoted when processing the filename). That file could not be in the same directory.
-A line that begins with #define constant-name value shold be processes as follow: all the constant_should be replaced with value.
the applicatiom should understand also two commands.

-i filename //(filename of the .c inpout file to process)
-c indicate that comments should be left in the output file insted of being removed.
sss

Es: preprocess -i myprog.c
preprocess -i myprog.c -c

We have to create the file output with the .o exentions and if it does not exist, we have to create it.

Variables: all lowercase
Constant : all uppercase

The code file required to build your application.
A makefile to allow building of your application
A readme file explaining how to use the preprocess application, as weel as how to build it from the source, and the tool chain used for building (Microsoft Compiler, GCC, etc..)

Make file

A make file is a file used to build our application/s. It contains all the necessary instructions in a single file, rather tjam typing them in the command line each time. To use a male file with
Microsoft tools, we have to create a file called makefile (without any extension). This file comntains the necessary instructions. We use a tool called nmake to perform the build operation of us. nmale looks at the makefile anddetermines
which particular build we want to perform.

It is important to know that our application is called preprocessor.

Now, we are going to analize the the makefile created for the our application.

```
preprocessor:
	cl preprocessor.c

all:
	cl preprocessor.c

clean:
	del *.obj
	del *.exe
```

A clean configuration deletes all the files generate during our builds. This can help us to tidy you to tiny up temporary fils that we really don't need to stored. We can add as well "del *.asm" in the clean configuration but for our application it is not necessary.

Tool chain used

For develop our application we have used:
-Microsoft Visual studio 2017 --> https://visualstudio.microsoft.com/downloads/?rr=https%3A%2F%2Fwww.google.co.uk%2F
-Microsoft clang compiler --> https://en.wikipedia.org/wiki/Clang
-Visual studio developer prompt -> https://docs.microsoft.com/en-us/dotnet/framework/tools/developer-command-prompt-for-vs
-gitlab -> https://about.gitlab.com/

Online resources

Programming foundamentals workbook (Created by Dr Kevin Chalmers and Mainted by Dr. Simon Powers)
stackoverflow -> C forum
tutorialpoint -> Programming in C

Build and use the application

Compile from developer Command Promt For VS 2017:

```
H:\>cl mainP.c
Microsoft (R) C/C++ Optimizing Compiler Version 19.15.26726 for x86
Copyright (C) Microsoft Corporation.  All rights reserved.

mainP.c
Microsoft (R) Incremental Linker Version 14.15.26726.0
Copyright (C) Microsoft Corporation.  All rights reserved.

/out:mainP.exe
mainP.obj
```

Analize the math_functions.c file print in the math_functions.o file also the comments:

```
H:\>mainP preprocess -i math_functions.c -c
Input file found: math_functions.c
Comments list required
Total no of lines: 59
Total no of comment line: 20
```

Analize the string_functions.c file print in the string _functions.o without include the comments:

```
H:\>mainP preprocess -i string_functions.c
Input file found: string_functions.c
Comments list is not required
Total no of lines: 41
Total no of comment line: 9
```