The task is to implement a simple dictionary-based translator. The translator should be able to read a dictionary file which contains a few thousand lines, each containing two words with the same meaning in two languages. Each of the two columns in the dictionary file contains files of one language. Using the dictionary, a translation of another text file should be created, which contains several sentences, each on a separate line. The translation should be printed on the standard output while keeping the structure of the input file. The translator does not have to take into account any grammar rules but translates by substituting one word for another using the dictionary. The name of the dictionary file should be the first command line argument, and the file's name to be translated is the second argument. The third command line argument is either 1 or 2 and indicates the dictionary's source language, i.e., the column of the dictionary language used in the file to be translated.
For the sake of simplicity, assume that the words are at most 100 characters and that the file to be translated contains less than 10,000 words.
You have to zip and upload your solution to the brute system.

Suggested, but optional walkthrough:

1) Download and unzip the assignment files (e.g. by "`wget https://cw.fel.cvut.cz/wiki/_media/courses/be5b99cpl/lectures/test_cpl_2024-X.zip`"). Check their structure and create a compilable `main.c`, which accepts three command line arguments or produces an error message to standard error output. **(1 point)**

2) Create a suitable data structure to contain the dictionary and read in the contents of the dictionary file. Check if the contents are read correctly. **(2 points)**

3) Read in the source file, parse its first line to individual words, find them in the dictionary and print their translation into the output file. You can use the **english_example.txt** and **tagalog_example.txt** to check the functionality - their contents should mean the same. **(2 points)**

4) Extend the functionality so that you can process the entire file line-by-line. Create the translation of **tagalog_text.txt (2 points)**

5) Add the capability to identify the language used. If the fourth command line argument is '0', the language autodetection should be used. **(1 point)**

6) Place the functions performing the translation operations in another module with a header file and use the main.c only for the main function. **(1 point)**

7) Write a Makefile that compiles and links the module and the main program, and functions have comments regarding their arguments and what the function does. Correct any compiler warnings when compiling with "`-Wall -std=c99 -pedantic`" flags. **(1 point)**

8) Make your programme work for arbitrary word and input file size. (**+ 2 extra points**)