PA301 – Simple Translation Dictionary

We want to implement a program which will help students learn the vocabulary of foreign languages. The first step, which has been assigned to you, is to build a data structure representing a list of English words and their translation in a foreign language. We'll also need to provide functions allowing us to lookup translations from English-to-Foreign and Foreign-to-English. The main data structure of this program encapsulates some variables specifying the maximal and current number of words in the dictionary along with a pointer to a dynamically allocated array of pointers on a data structure representing a pair of words. A skeleton for this project is provided to you along with the data struct definitions.

Files you will be working with

You will be provided with several files to get you started working on this assignment. Make sure you edit them but do not alter their name or add new files. Here are the files;

tests.c	will contain the functions you will write to run your tests on the other functions you have to implement in this programming assignment.
tools.c	will contain the definition of the functions you will have to write
tools.h	will contain the headers of the functions you will have to write
main.c	will contain already provided code to start your tests.

Business as usual

By now, you should be used to the fact that you need to implement a solution to the requirements expressed in these instructions, develop tests, posts and discuss your tests on the module's forum and refactor here and then to improve the quality of your solution.

The following instructions will assume you're already used to this from the two previous practice assignments and will therefore skip reminders.

Please note that, unlike with previous PAs, you are expected to write your testing functions from scratch. Use peer testing to get help with this and debugging.

Task #1 – Implement and test dictionary_build

This first function to implement in *tools.c* will be invoked every time you need to start working on a new dictionary. You will be doing so in the *tests.c* file as you build a dictionary to test with.

Part of your grade on this implementation will be on how well your function handles error situations; e.g. we ask for a dictionary of negative size, there is no memory available for your *malloc* invocations…

Any time you encounter one of these, or similar, errors, you should return a specific value from your function which is described in the program's source itself. When you test your function with such parameters, you should verify its return value is appropriate.

Task #2 – Implement and test dictionary_free

This 2nd function will be used to de-allocate the dictionary or dictionaries you might have used in your *tests.c* file.

Make sure you dispose of all the memory you allocated for the various data structures involved in each dictionary. Draw on paper a diagram of everything that has been allocated so that you verify you are not leaving any memory leaks.

When you are invoking this function with inappropriate parameters, verify its return value is appropriate based on the documentation in the program itself.

Task #3 – Implement and test dictionary_add

The 3rd function will take a dictionary and two strings as parameters. It will use **strdup** to make a copy of the words provided and insert them as a new element in the dictionary.

Again, detect possible bad parameters; e.g. no words provided; 1 word instead of 2… and verify the return value is appropriate based on the documentation in the program itself.

Task #4 – Implement and test dictionary_translate

This function, as indicated in its comments, will return a **strdup** copy of the translation of one of the two strings passed as parameter. It is expected that one will use this function with a valid dictionary, and specify one of the two strings to be NULL.

- If the English word parameter is NULL, we are providing a string in the foreign language to be translated into English.
- If the foreign word is NULL, we do the opposite.

Passing two non-NULL strings or a NULL dictionary are errors which should be handled by your function returning the value described in the documentation in the program itself.