



DASH 00 - 42 MALAGA

word_puzzle

Summary: This document is the subject for a dash created by 42 Málaga.

Version: 1.00

Contents

I	A word about dashes	2
II	Introduction	3
III	General instructions	4
IV	Mandatory part	5
V	Submission	7

Chapter I

A word about dashes

The dashes are a disruptive way to gamify the 42 Cursus. You will have a certain amount of time to finish and submit this challenge. Mind that, in order to win some points, your code must be fully functional and must follow all the rules.

Chapter II

Introduction

In this dash, you will need to create a program that checks whether it is possible to solve a puzzle.

Chapter III


General instructions

Please, read thoroughly all the instructions.

- This subject is the one and only trustable source. Don't trust any rumor.
- Your program **must** be written in C.
- Obviously, your program **must respect the Norm**, otherwise, it's a 0.
- If your program doesn't compile, it's a 0.
- Be careful about the access rights of your files and folders.
- Your assignments **WON'T** be evaluated by your peers, but by the DASHinette.
- This dash is a solo project, meaning that even if you work with your peers as a team, everyone must submit the code.
- You have a question? Ask your left neighbor. Otherwise, try your luck with your right neighbor.
- You must read the examples thoroughly. They can reveal requirements that are not obvious in the assignment's description.
- By Thor, by Odin! Use your brain!!!

Chapter IV

Mandatory part

	Exercise
	word_puzzle
	Turn-in directory : <i>ex/</i>
	Files to turn in : word_puzzle.c
	Allowed functions : *

- The program must receive as arguments a list of **N** words and determine whether the puzzle has a solution.
- We will consider that the puzzle **has a solution** if the words can be arranged consecutively such that each word starts with the letter that the previous word ends with.
- For example, "dash" and "happy" would be arranged correctly.
- A word will be considered as a **lowercase string**, with a length of L. ($1 \leq L \leq 100$)
- You must create a function called "**ft_word_puzzle()**". This function will receive the following arguments:
 - N - Number of words. ($2 \leq N \leq 100000$)
 - words - An array of N strings.
- Your function must return NULL if there is no solution, and the ordered list if there is a solution.
- If there is no solution, your program must display the message "**Error: No solution found**" on the terminal. Otherwise, it must display the solution, printing each word in order **followed by a newline**.

- If your program receives no arguments, you must display the message "**Error: No arguments given**".
- If one of the arguments is not considered a word, or it is malformed, your program must display the message "**Error: Arguments are not correct.**".
- Example:

```
%> ./a.out "dash" "yeehaw" "happy" | cat -e
dash$
happy$
yeehaw$
%> ./a.out "pikachu" "morpeko" "chikorita" | cat -e
Error: No solution found$
%> ./a.out "j4me2" | cat -e
Error: Arguments are not correct$
```

- Your function must be declared as follows:

```
ft_word_puzzle(size_t N, char **words)
```

Chapter V

Submission

- You must upload all the files requested to the turn-in repo.



No Peer evaluation for this dash, once you submit it, the DASHinette and the Staff team will take care of the rest.