Dash - word_puzzle

word_puzzle

Summary: *this document is the subject for the dash @ 42Tokyo.*

Contents

Chapter I

Foreword

Try your hand at some dynamic programming!

Chapter II

Objective

Create the fastest `word_puzzle.c`.
All functions allowed!

Chapter III

Instructions

- If your program doesn't compile, it's a 0.
- Evaluation will be done on 42 Tokyo's Mac.
- This dash is a solo project.
- Turn in your code inside the turn-in repository.

Chapter IV

Exercice 00 : word_puzzle

	Exercise 00
word_puzzle	
Turn-in directory : *ex*00/	
Files to turn in : `word_puzzle.c`	
Allowed functions : `*`	

- Given N words, determine whether the word puzzle is solvable or not.
- The word puzzle is considered solved when the words are arranged in a sequence, such that every word begins with the same letter as the previous word ends.
- For example, the word "dash" can be followed by the word "happy", but not the other way around.
- A word is defined as a string of lowercase characters, with length L. $(1 \leq L \leq 100)$
- Your function should accept 2 variables as input.
  - `N` - Number of words in `words`. $(2 \leq N \leq 100000)$
  - `words` - An array of `N` strings.
- Your function should return a 1 if the word puzzle is solvable, and 0 if it is not.
- Example:
  Input: N = 2, words = {"dash", "hard"}
  Output: 1
  Input: N = 3, words = {"dash", "too", "hard"}
  Output: 0
- Your function must be declared as follows:

```
int                 word_puzzle(size_t N, char **words);
```