



DASH 00 - 42 MALAGA

word_puzzle

Summary: Este documento es el subject de un dash creado por 42 Málaga.

Version: 1.00

Contents

I	Unas palabras sobre los dashes	2
II	Introducción	3
III	Instrucciones generales	4
IV	Parte obligatoria	5
V	Submission	7

Chapter I

Unas palabras sobre los dashes

Los dashes son una forma innovadora de gamificar el Cursus de 42. Tendrás una cantidad limitada de tiempo para completar y enviar este desafío. Ten en cuenta que, para ganar puntos, tu código debe ser completamente funcional y cumplir con todas las reglas.

Chapter II

Introducción

En este dash, tendrás que crear un programa que compruebe si es posible la resolución de un puzzle.

Chapter III


Instrucciones generales

Por favor, lee detenidamente todas las instrucciones.

- Este documento es la única fuente confiable. No confíes en rumores.
- Tu programa **debe** estar escrito en C.
- Obviamente, tu programa **debe respetar la Norma**, de lo contrario, será un 0.
- Si tu programa no compila, será un 0.
- Ten cuidado con los permisos de acceso de tus archivos y carpetas.
- Todo lo que entregues **NO** será evaluado por tus compañeros, sino por la DASHinette.
- Este dash es un proyecto individual, lo que significa que, incluso si trabajas con tus compañeros en equipo, cada uno debe enviar su propio código.
- ¿Tienes una pregunta? Pregunta a la persona de tu izquierda. Si no, prueba suerte la de la derecha.
- Debes leer los ejemplos minuciosamente. Pueden revelar requisitos que no son obvios en la descripción de la asignación.
- ¡Por Thor, por Odín! ¡Usa tu cerebro!

Chapter IV

Parte obligatoria

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

- El programa deberá recibir por argumentos una lista de **N** palabras y determinar si el puzzle tiene o no solución.
- Consideraremos que el puzzle **tiene solución** cuando las palabras puedan ordenarse de forma consecutiva, de manera que cada palabra empiece con la letra con la que finaliza la palabra anterior.
- Por ejemplo, "dash" y "happy" estarían ordenadas de manera correcta.
- Consideraremos palabra una **cadena de caracteres en minúscula**, de una longitud L . ($1 \leq L \leq 100$)
- Deberás crear una función "**ft_word_puzzle()**". Esta función recibirá por parámetros los siguientes argumentos:
 - N - Número de palabras. ($2 \leq N \leq 100000$)
 - words - Un array de N strings.
- Tu función deberá devolver NULL si no hay una solución y la lista ordenada en caso de que hubiese solución.
- En caso de no tener solución, tu programa deberá mostrar en la terminal el mensaje "**Error: No solution found**". En caso contrario, deberá mostrar la solución, imprimiendo cada palabra de forma ordenada **seguidas de un salto de línea**.

- Si tu programa no recibe argumentos, deberás mostrar el siguiente mensaje: **"Error: No arguments given"**.
- Si uno de los argumentos recibidos no es una palabra válida, deberás mostrar el mensaje **"Error: Arguments are not correct."**.
- Ejemplo:

```
%> ./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$
```

- Tu función deberá de declararse de la siguiente forma:

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

Chapter V

Submission

- Debes subir todos los archivos solicitados al repositorio.



No habrá evaluación entre pares para este dash. Una vez que lo envíes, la DASHinette y el Staff se encargarán del resto.