Cryptographer

By Sasmita Wiropati

Description

Kamu dengan temanmu sering mengirim file yang berisi array data terenkripsi untuk saling berkomunikasi satu sama lain. Namun, karena kalian berdua tidak jago dalam hal enkripsi dan dekripsi maka kalian hanya mengacu pada ASCII.

Suatu saat kamu mau mengirim file dengan isi berikut ke temanmu:

01001000	01100101	01101100	01101100	01101111
01011000	01000110	01110101	01001101	01110010
01111001	01001010	01101000	01100011	01010101
01110110	01110001	01010011	01101010	01110100
01000111	01100110	01001011	01101101	01011001

Namun kamu ingin memeriksa terlebih dahulu bahwa pesan yang kamu kirim sudah ada di dalam file atau belum. Kamu hanya bisa memeriksa dari arah kiri ke kanan atau atas ke bawah saja karena mempermudah pembacaan ketika didekripsi.

Buatlah program untuk mendekripsi data tersebut, **penggunaan square brackets [] dilarang beserta gunakan pointer untuk mengakses isi dari array kecuali untuk input scanf.**

Input

Sebuah string, word, yang merepresentasi kata yang ingin dicari dan data berjenis string sebanyak 5x5.

Output

Output akan bergantung terhadap dua kondisi:

- Jika ditemukan kata yang dicari:
  "{word} found"

- Jika tidak ditemukan kata yang dicari:
  "Failed to find the specified word: {word}"

Constraint

$1 \leq length(word) \leq 5$

Example

```
Hello
01001000 01100101 01101100 01101100 01101111
01011000 01000110 01110101 01001101 01110010
01111001 01001010 01101000 01100011 01010101
01110110 01110001 01010011 01101010 01110100
01000111 01100110 01001011 01101101 01011001
```

```
Hello found
```

```
Hello
01001000 01011000 01111001 01101100 01000111
01100101 01000110 01110101 01001101 01110010
01101100 01001010 01101000 01100011 01010101
01101100 01110001 01010011 01101010 01110100
01101111 01100110 01001011 01101101 01011001
```

```
Hello found
```

```
Hello
01001000 01011000 01111001 01101100 01000111
01100101 01000110 01110101 01001101 01110010
01101111 01001010 01101000 01100011 01010101
01110110 01110001 01010011 01101010 01110100
01101111 01100110 01001011 01101101 01011001
```

```
Failed to find the specified word: Hello
```

Cryptographer

By Sasmita Wiropati

Description

You and your friend often send files containing encrypted arrays of data to communicate with each other. However, because neither of you are skilled in encryption and decryption, you both only refer to ASCII.

One day, you're about to send this file to your friend:

01001000	01100101	01101100	01101100	01101111
01011000	01000110	01110101	01001101	01110010
01111001	01001010	01101000	01100011	01010101
01110110	01110001	01010011	01101010	01110100
01000111	01100110	01001011	01101101	01011001

However, you want to check whether the message you're sending is already in the file or not before you send it. You can only check from left to right or top to bottom because it makes reading easier when decrypted.

Create a program to decrypt the data, **the use of square brackets [] are forbidden and are replaced by the use of pointers to access the content of an array except for input scanf.**

Input

A string, word, that represents the word you want to search for, and a data with 5x5 size array of strings.

Output

The output will depends on two conditions:

- If you find the word you're searching:
  "{word} found"

- If you don't find the word you're searching:
  "Failed to find the specified word: {word}"

Constraint

$1 \leq length(word) \leq 5$

Example

```
Hello
01001000 01100101 01101100 01101100 01101111
01011000 01000110 01110101 01001101 01110010
01111001 01001010 01101000 01100011 01010101
01110110 01110001 01010011 01101010 01110100
01000111 01100110 01001011 01101101 01011001
```

```
Hello found
```

```
Hello
01001000 01011000 01111001 01101100 01000111
01100101 01000110 01110101 01001101 01110010
01101100 01001010 01101000 01100011 01010101
01101100 01110001 01010011 01101010 01110100
01101111 01100110 01001011 01101101 01011001
```

```
Hello found
```

```
Hello
01001000 01011000 01111001 01101100 01000111
01100101 01000110 01110101 01001101 01110010
01101111 01001010 01101000 01100011 01010101
01110110 01110001 01010011 01101010 01110100
01101111 01100110 01001011 01101101 01011001
```

```
Failed to find the specified word: Hello
```