고 급 문 제 해 결

Chapter 6 : String

Chapter 8 : Stack

4주차 발표 자료

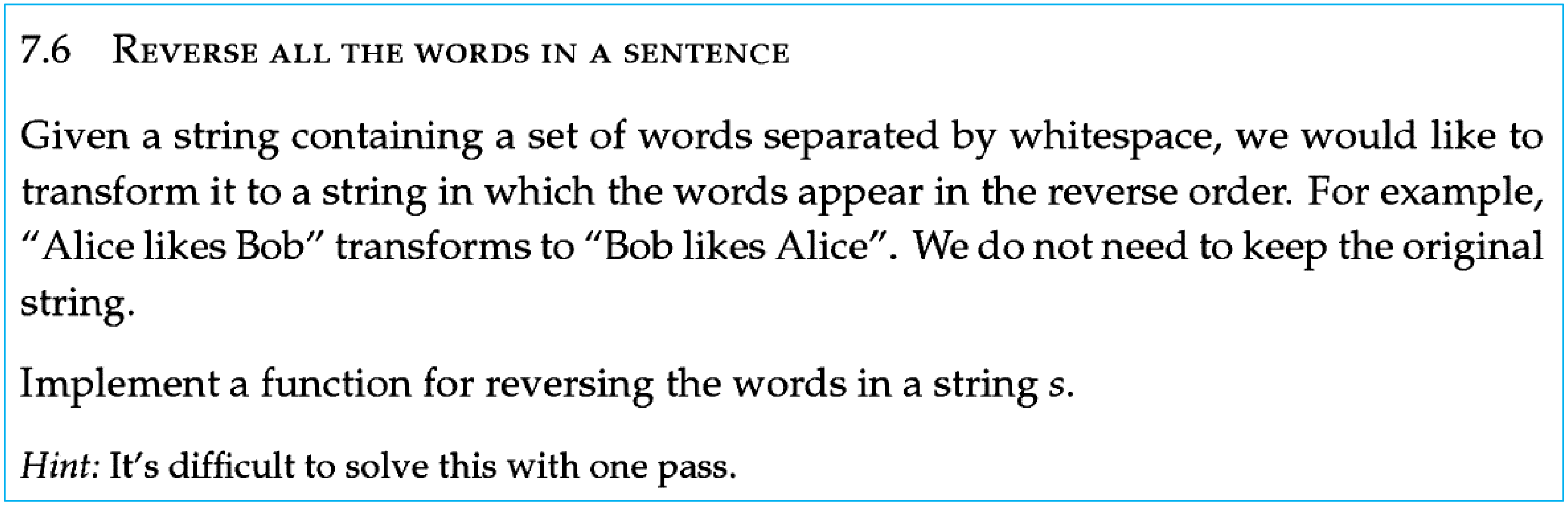
2018117543 한재성

Contents

Chapter 6

String

문장의 모든 단어 뒤집기



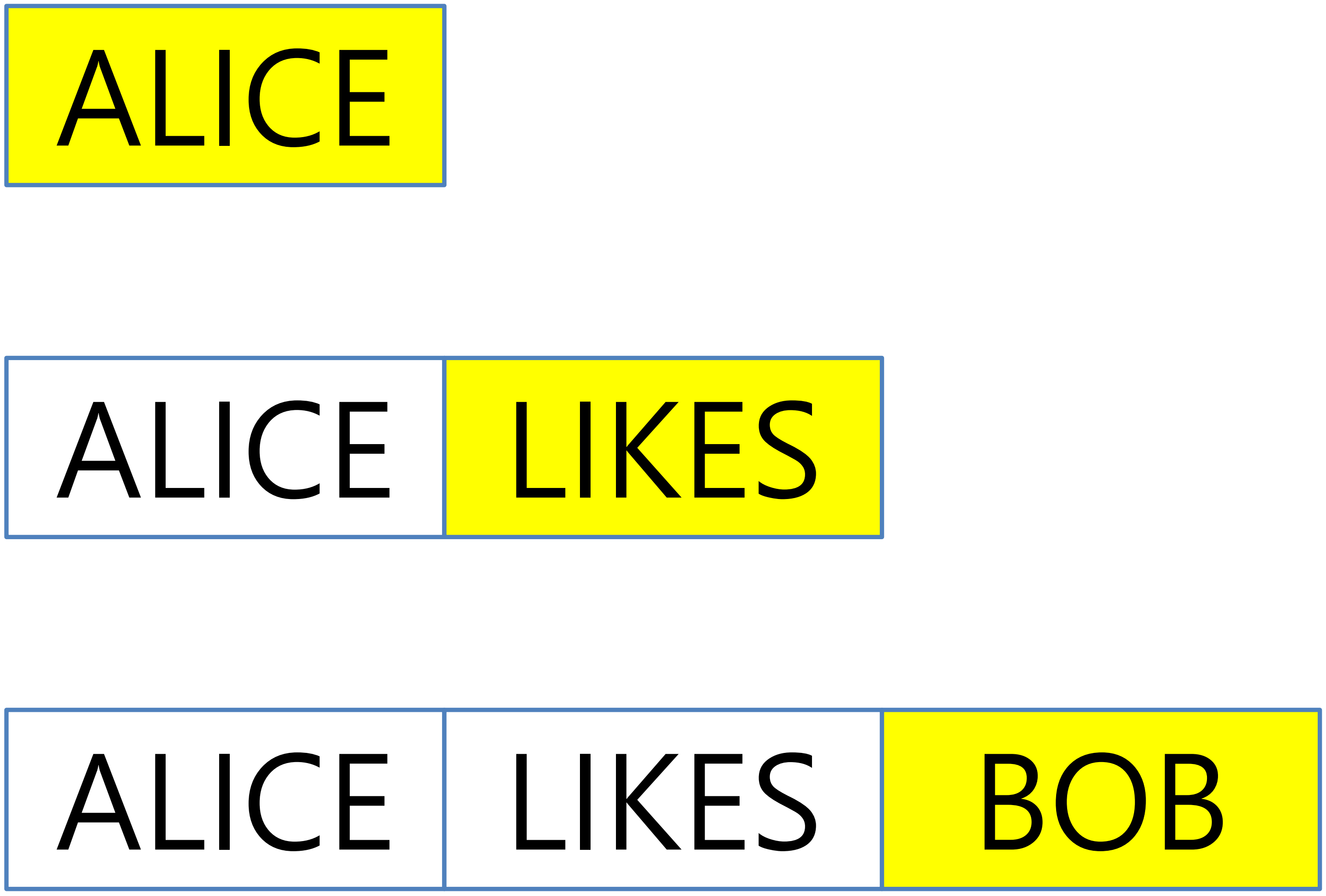
7.6 Reverse all the words in a sentence

Given a string containing a set of words separated by whitespace, we would like to transform it to a string in which the words appear in the reverse order. For example, “Alice likes Bob” transforms to “Bob likes Alice”. We do not need to keep the original string.

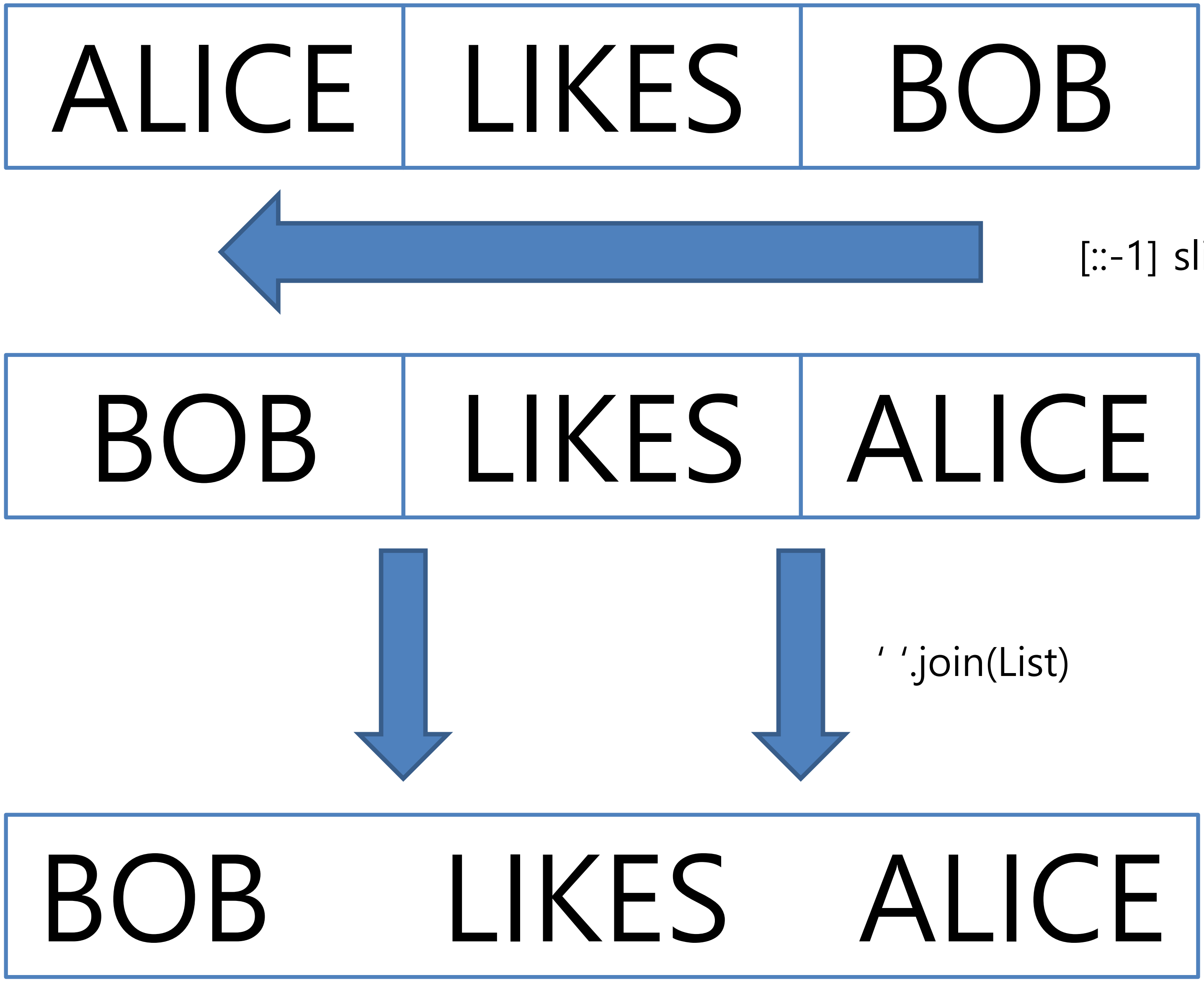
Implement a function for reversing the words in a string s.

Hint: It’s difficult to solve this with one pass.

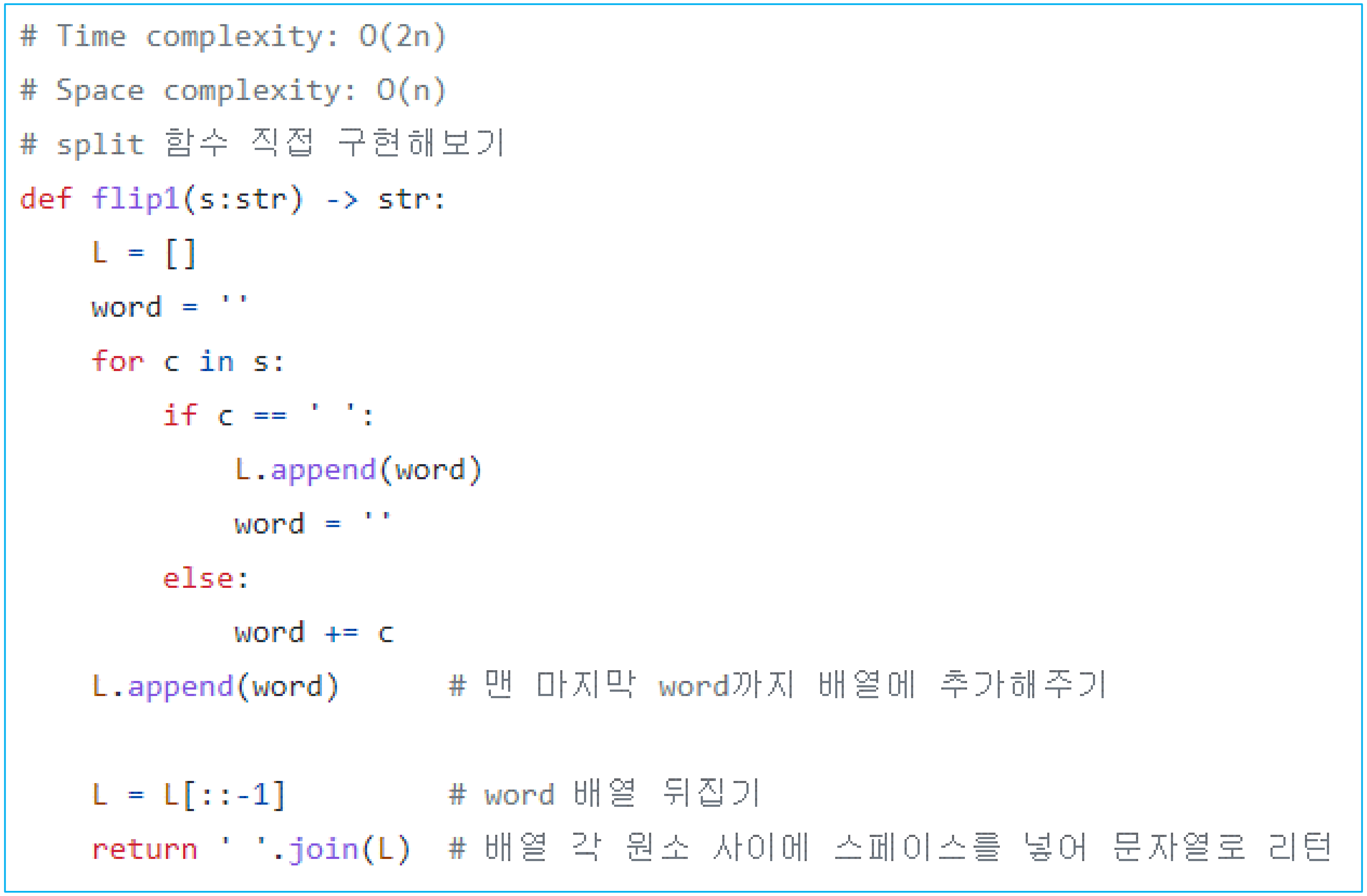
split() 구현



split() 구현


ALICE
LIKES
BOB
[::-1] slicing
BOB
LIKES
ALICE
' '.join(List)
BOB LIKES ALICE

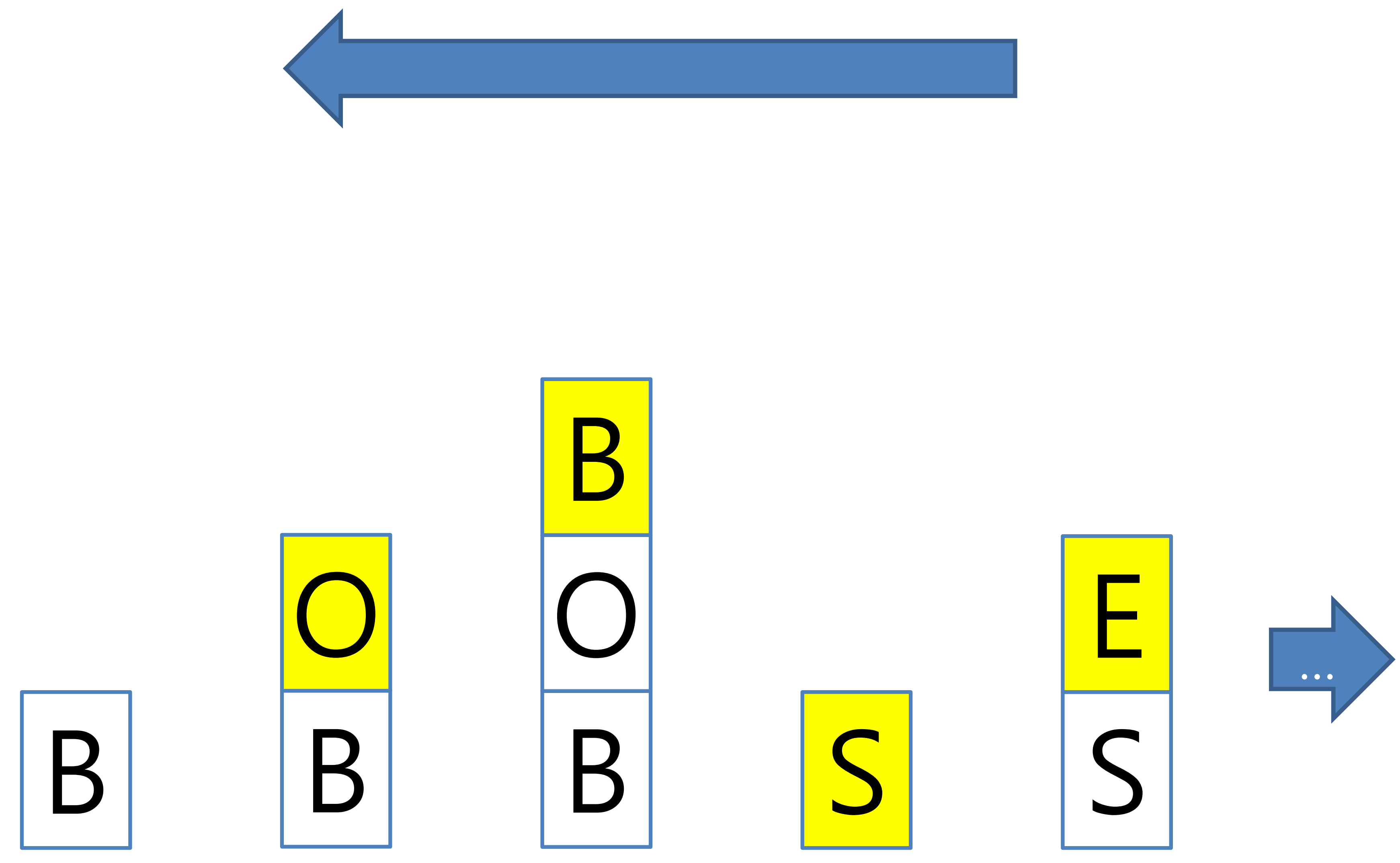
split() 구현



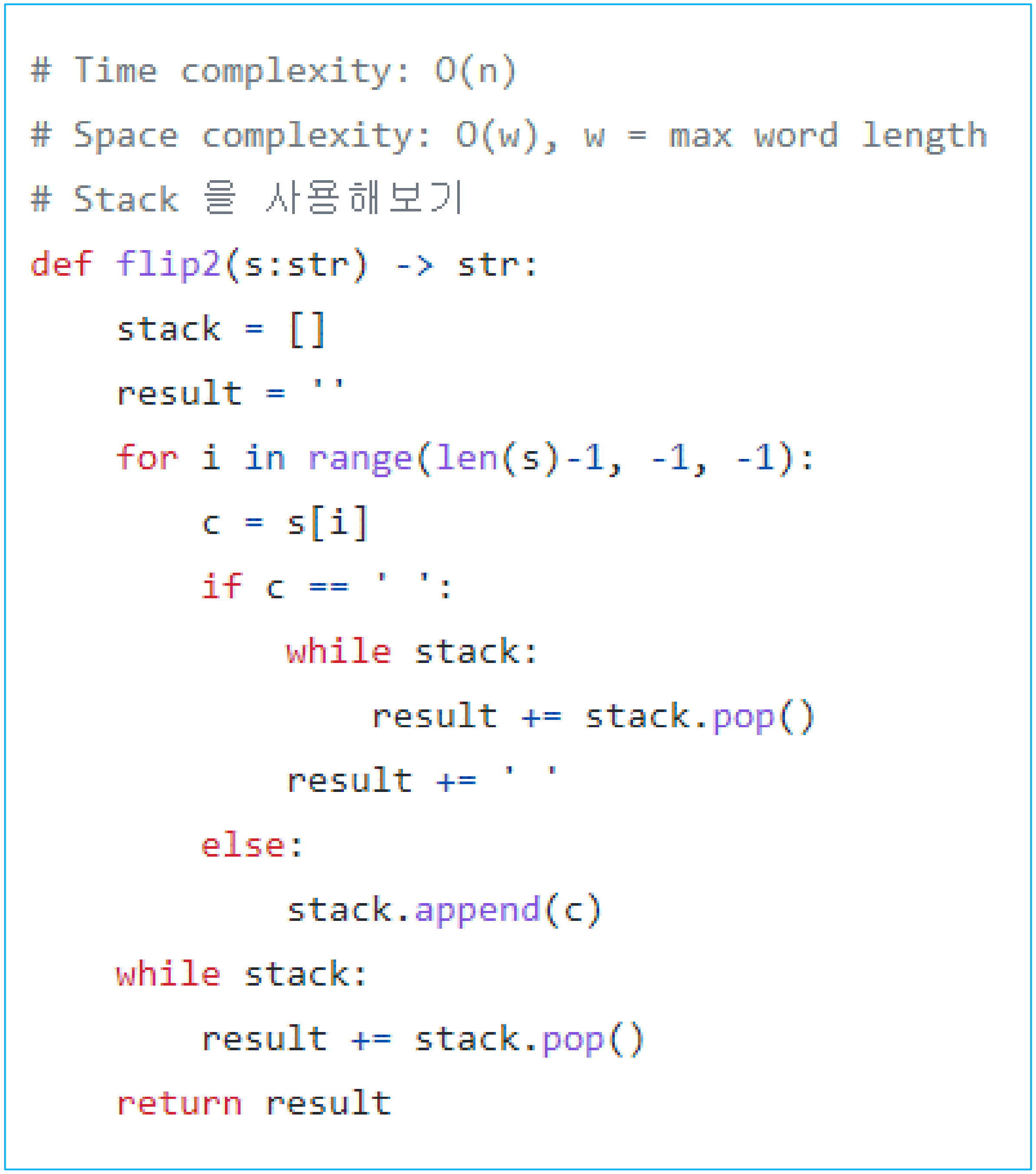
```python
# Time complexity: O(2n)
# Space complexity: O(n)
# split 함수 직접 구현해보기
def flip1(s:str) -> str:
    L = []
    word = ''
    for c in s:
        if c == ' ':
            L.append(word)
            word = ''
        else:
            word += c
    L.append(word)      # 맨 마지막 word까지 배열에 추가해주기

    L = L[::-1]         # word 배열 뒤집기
    return ' '.join(L)  # 배열 각 원소 사이에 스페이스를 넣어 문자열로 리턴
```

스택 사용하기

ALICE LIKES BOB

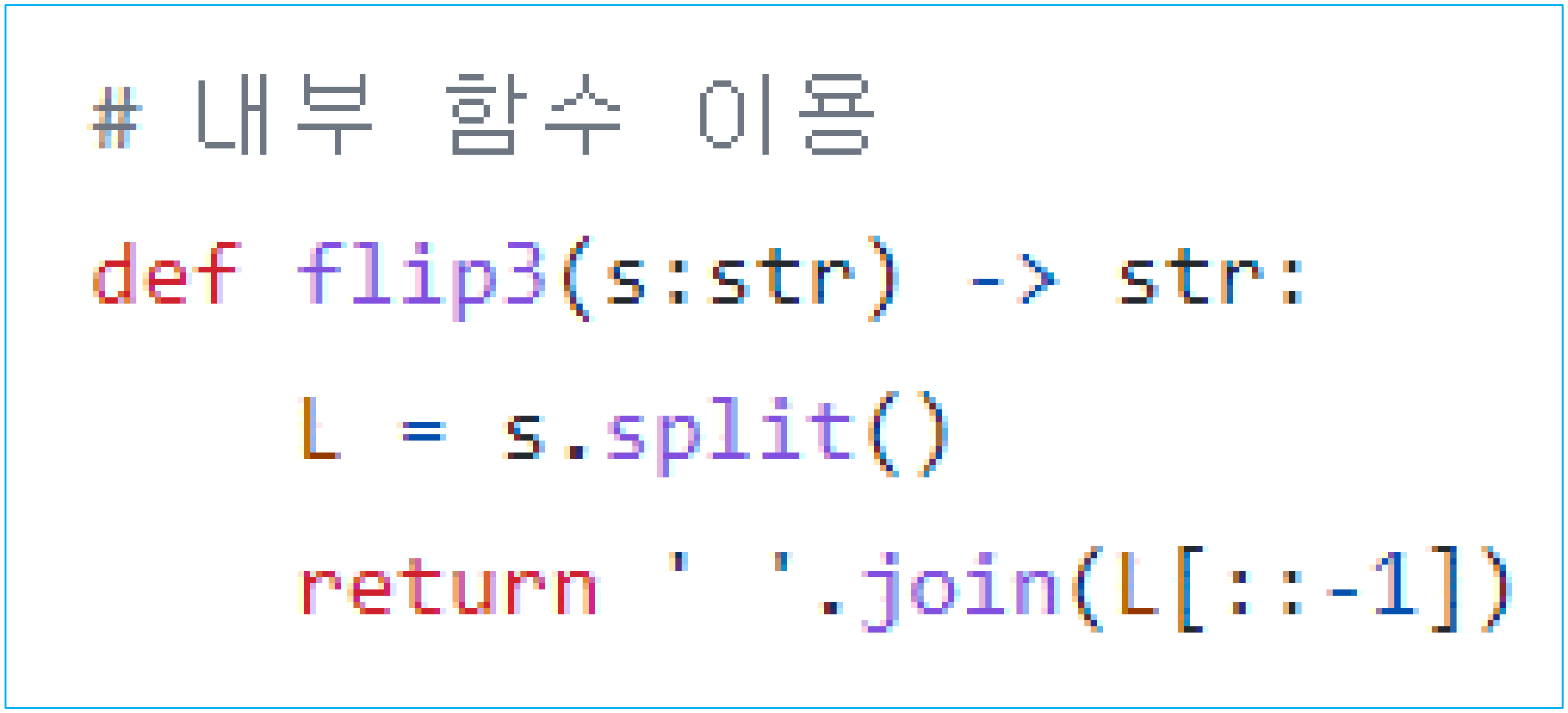


스택 사용하기



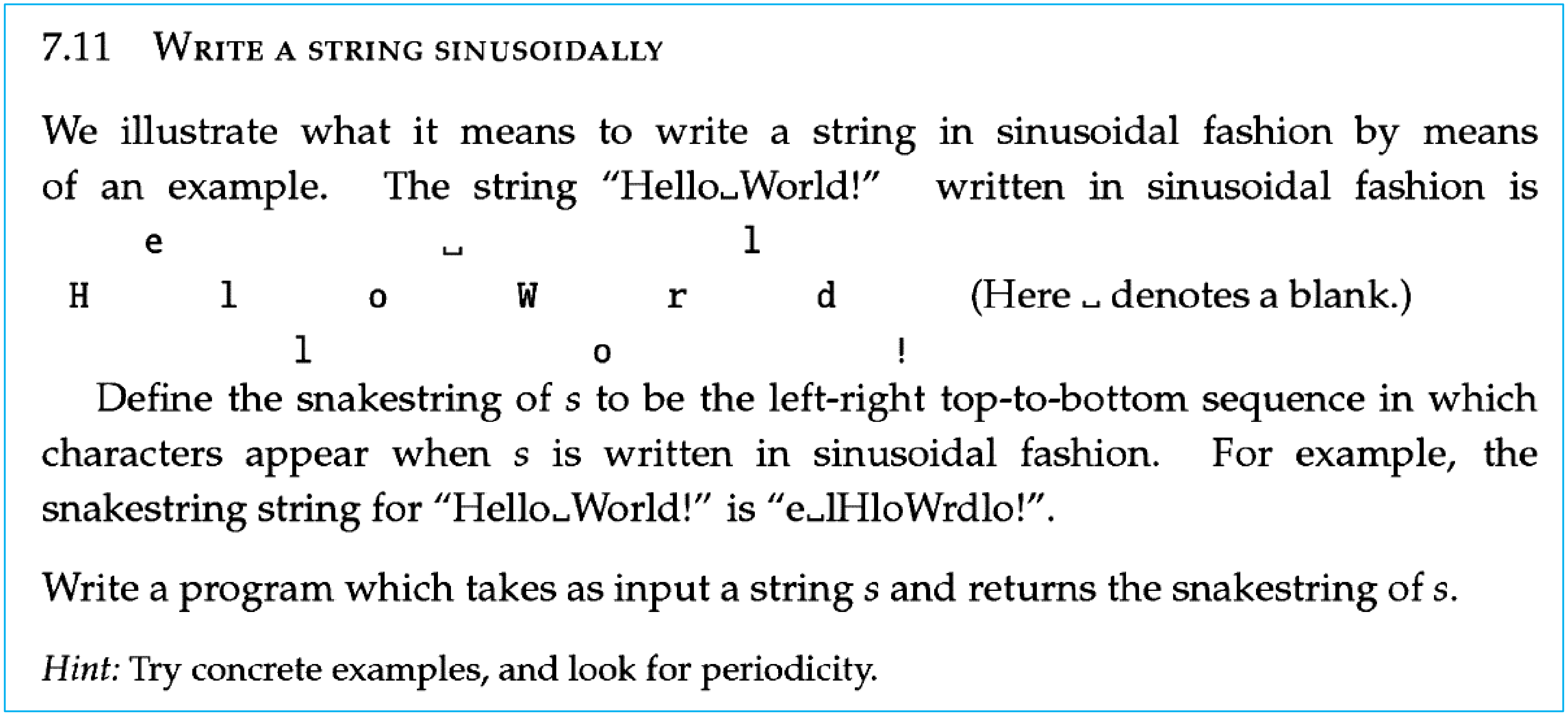
```python
# Time complexity: O(n)
# Space complexity: O(w), w = max word length
# Stack 을 사용해보기
def flip2(s:str) -> str:
    stack = []
    result = ''
    for i in range(len(s)-1, -1, -1):
        c = s[i]
        if c == ' ':
            while stack:
                result += stack.pop()
            result += ' '
        else:
            stack.append(c)
    while stack:
        result += stack.pop()
    return result
```

내부 함수 이용하기



```python
# 내부 함수 이용
def flip3(s:str) -> str:
    L = s.split()
    return ' '.join(L[::-1])
```

사인 곡선 형태로 문자열 작성



7.11 WRITE A STRING SINUSOIDALLY

We illustrate what it means to write a string in sinusoidal fashion by means of an example. The string "Hello␣World!" written in sinusoidal fashion is

```
  e       ␣       l
H   l   o   W   r   d        (Here ␣ denotes a blank.)
      l       o       !
```

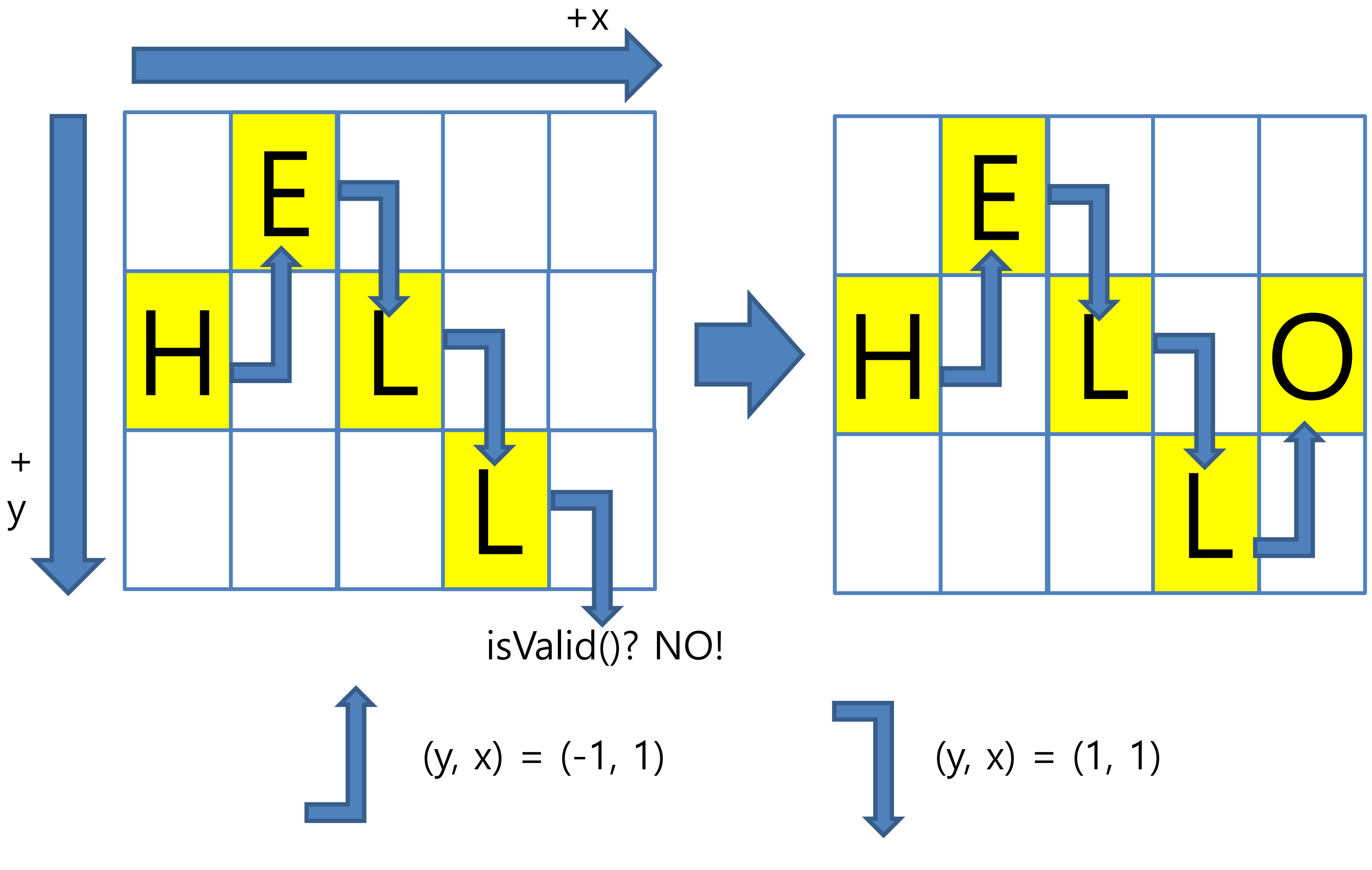
Define the snakestring of *s* to be the left-right top-to-bottom sequence in which characters appear when *s* is written in sinusoidal fashion. For example, the snakestring string for "Hello␣World!" is "e␣lHloWrdlo!".

Write a program which takes as input a string *s* and returns the snakestring of *s*.

Hint: Try concrete examples, and look for periodicity.

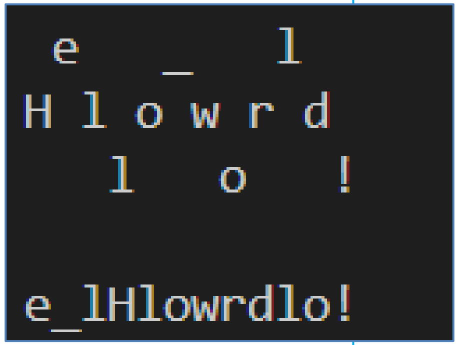
배열을 사용해서 구현

HELLO WORLD!

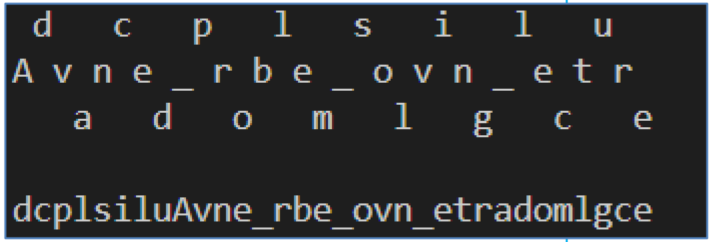


배열을 사용해서 구현

```python
def print_as_sine(s:str) -> None:
    L = [[' '] * len(s) for _ in range(3)]
    dirs = {1:(-1, 1), -1:(1, 1)}   # y, x
    pos = (1, 0)    # Starting position
    next_dir = 1

    def isValid(y, x) -> bool:
        if 0 <= y < len(L) and 0 <= x < len(L[0]):
            return True
        return False

    for i in range(len(s)):
        y, x = pos
        L[y][x] = s[i] if s[i] != ' ' else '_'
        next_y, next_x = y + dirs[next_dir][0], x + dirs[next_dir][1]
        while i < len(s) - 1 and not isValid(next_y, next_x):
            next_dir *= -1
            next_y, next_x = y + dirs[next_dir][0], x + dirs[next_dir][1]
        pos = (next_y, next_x)
```



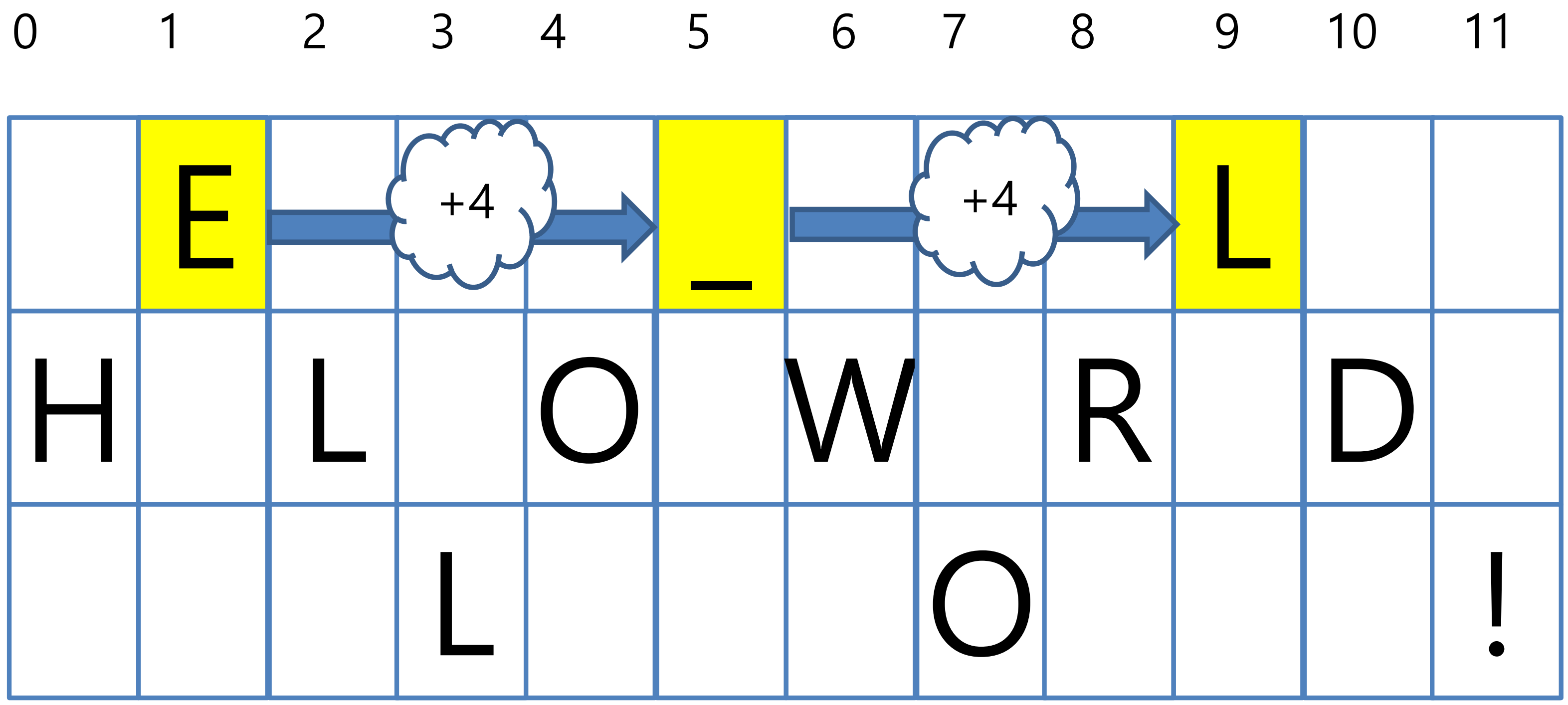
```
  e       _       l
H   l   o   w   r   d
      l       o       !

e_lHlowrdlo!
```



```
  d       c       p       l       s       i       l       u
A   v   n   e   _   r   b   e   _   o   v   n   _   e   t   r
      a       d       o       m       l       g       c       e

dcplsiluAvne_rbe_ovn_etradomlgce
```

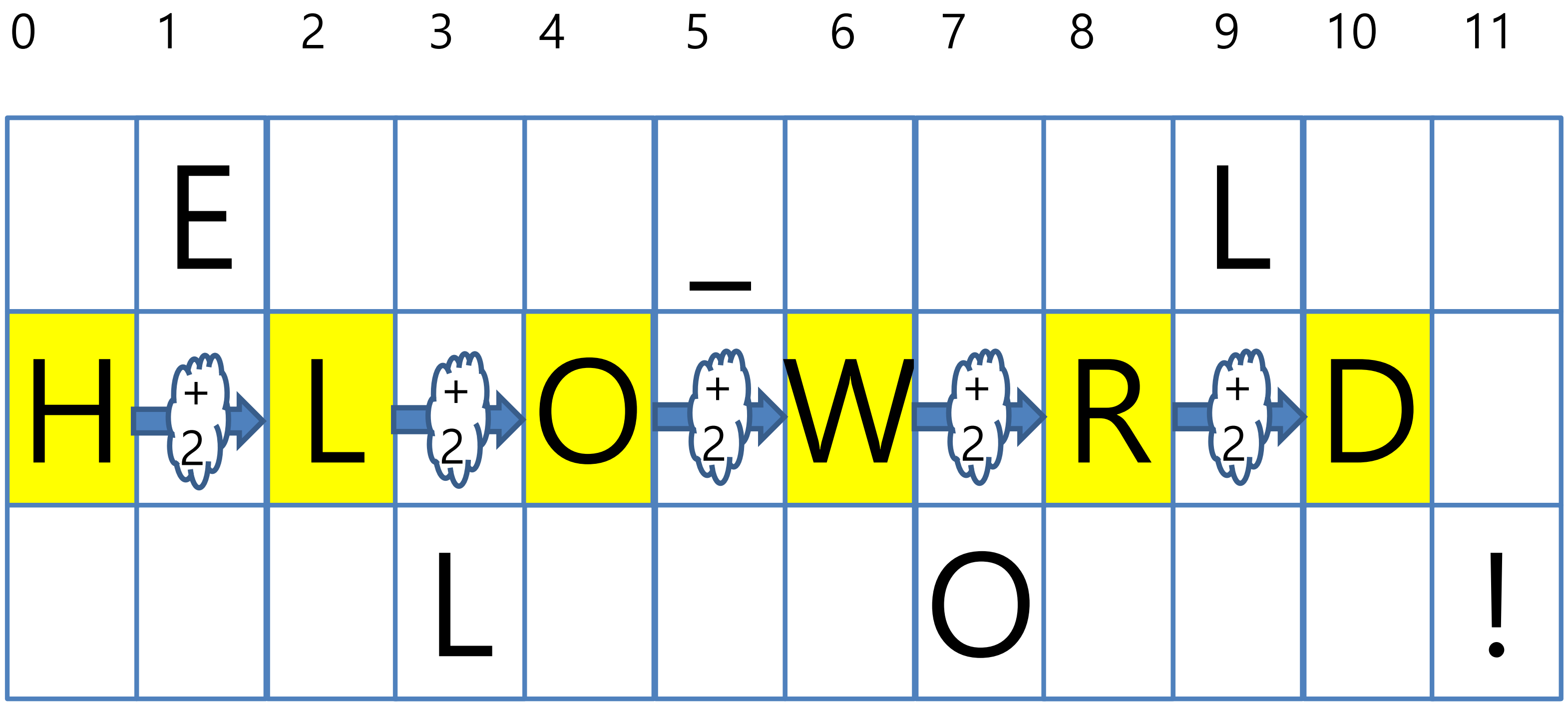
주기함수의 성질

HELLO WORLD!



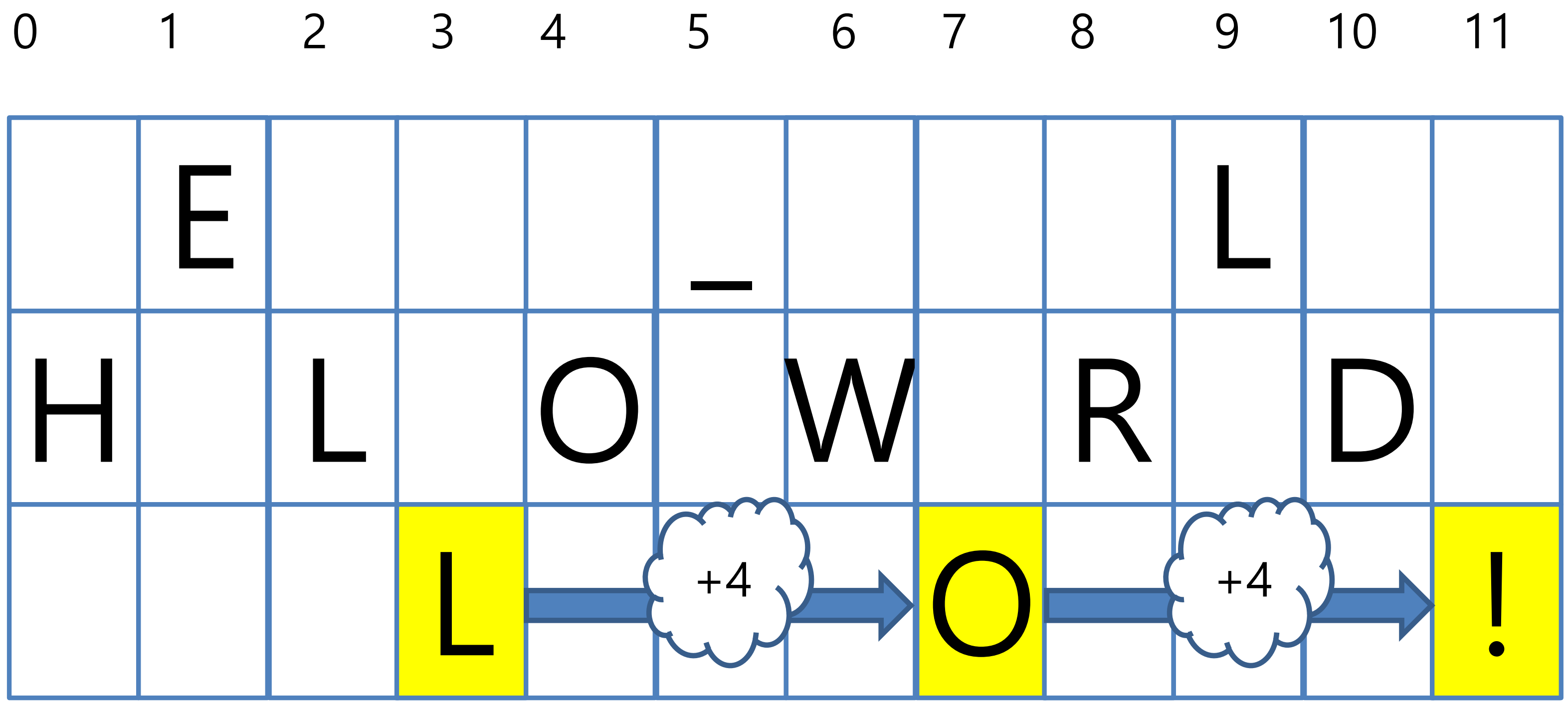
주기함수의 성질

HELLO WORLD!

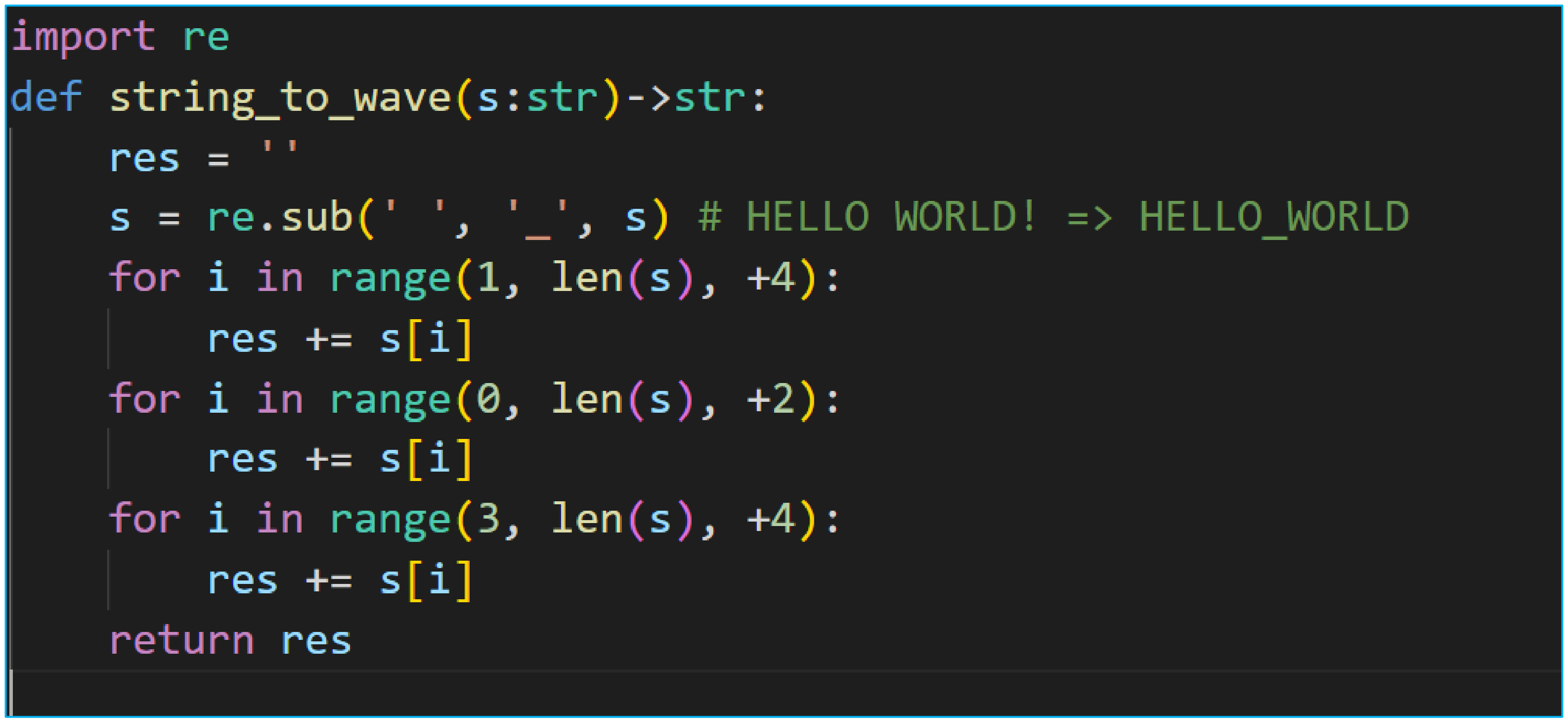


주기함수의 성질

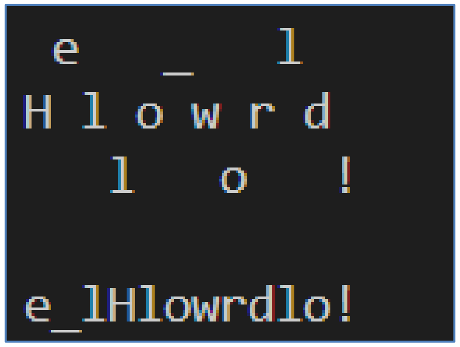
HELLO WORLD!



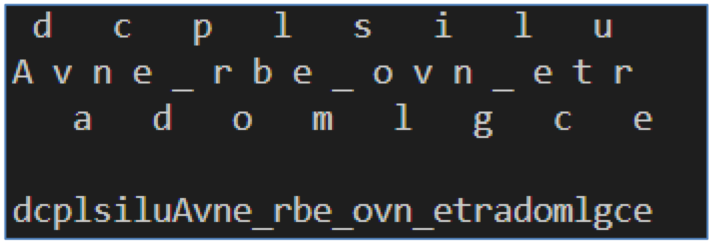
주기함수의 성질



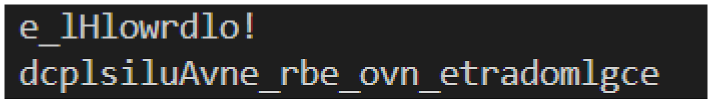
```python
import re
def string_to_wave(s:str)->str:
    res = ''
    s = re.sub(' ', '_', s) # HELLO WORLD! => HELLO_WORLD
    for i in range(1, len(s), +4):
        res += s[i]
    for i in range(0, len(s), +2):
        res += s[i]
    for i in range(3, len(s), +4):
        res += s[i]
    return res
```



```
 e   _   l
H l o w r d
   l   o   !

e_lHlowrdlo!
```



```
 d   c   p   l   s   i   l   u
A v n e _ r b e _ o v n _ e t r
   a   d   o   m   l   g   c   e

dcplsiluAvne_rbe_ovn_etradomlgce
```

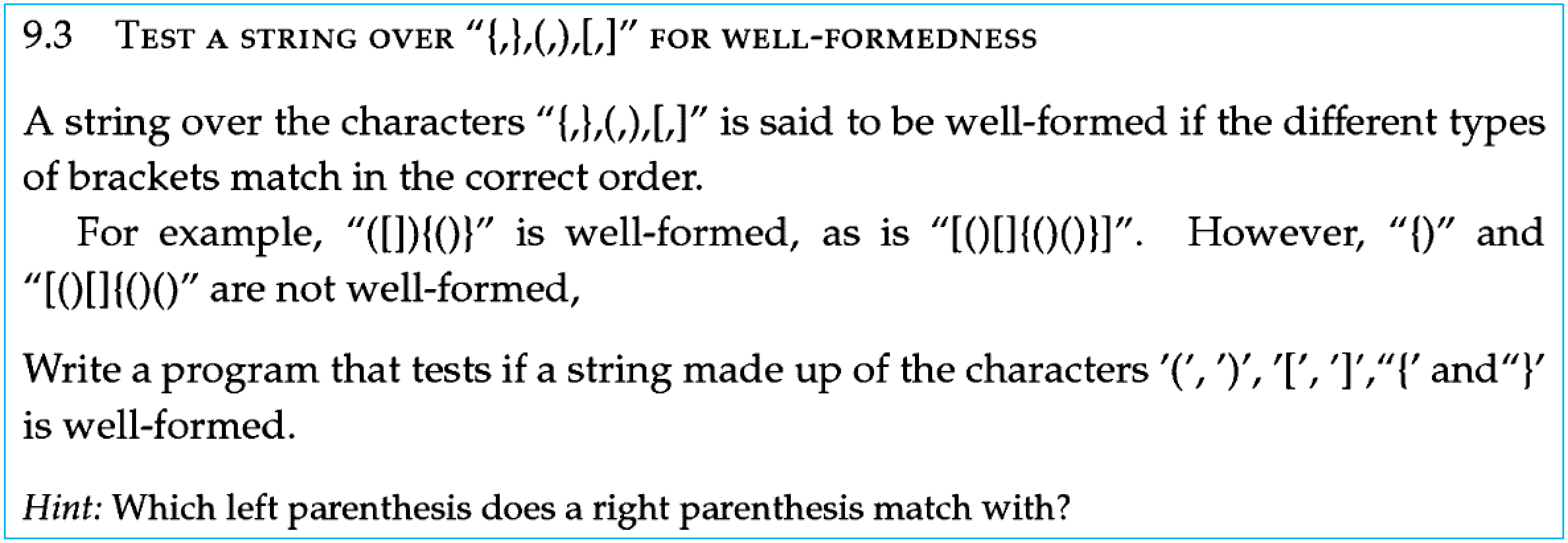


```
e_lHlowrdlo!
dcplsiluAvne_rbe_ovn_etradomlgce
```

Chapter 8

`Stack`

괄호가 짝을 이루는가?



9.3 Test a string over "{,},(,),[,]" for well-formedness

A string over the characters "{,},(,),[,]" is said to be well-formed if the different types of brackets match in the correct order.

For example, "([]){()}" is well-formed, as is "[()[]{()()}]". However, "{)" and "[()[]{()()" are not well-formed,

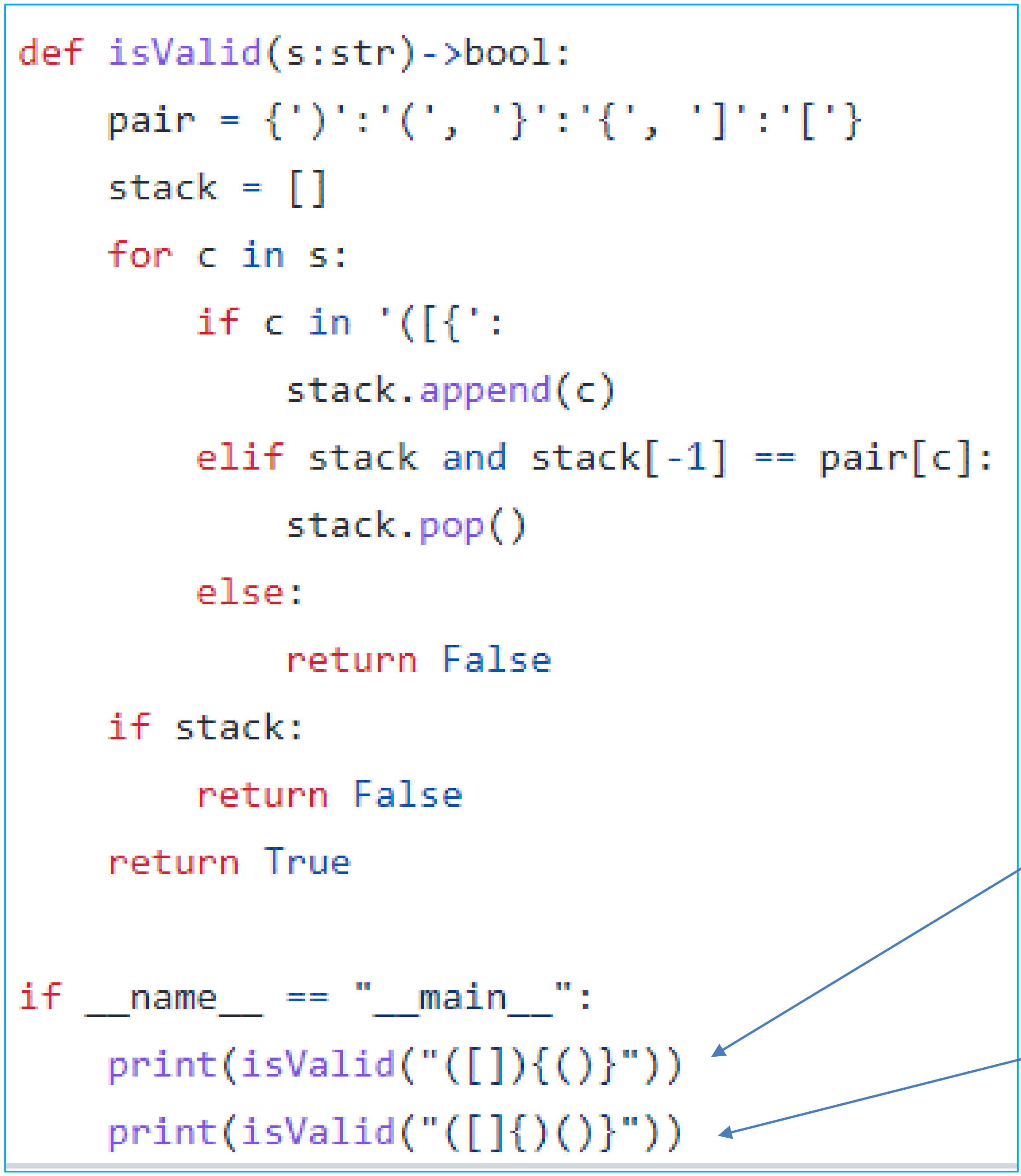
Write a program that tests if a string made up of the characters '(', ')', '[', ']',"{' and"}' is well-formed.

Hint: Which left parenthesis does a right parenthesis match with?

Stack을 사용하기

- 여는 괄호
  - ✓ (, { , [
  - ✓ 고려할 필요 없이 Stack에 push
- 닫는 괄호일때가 관건
  - ✓), },]
  - ✓ 매칭된다면 pop()
  - ✓ Stack.isEmpty() or not matching -> invalid.

Stack을 사용하기

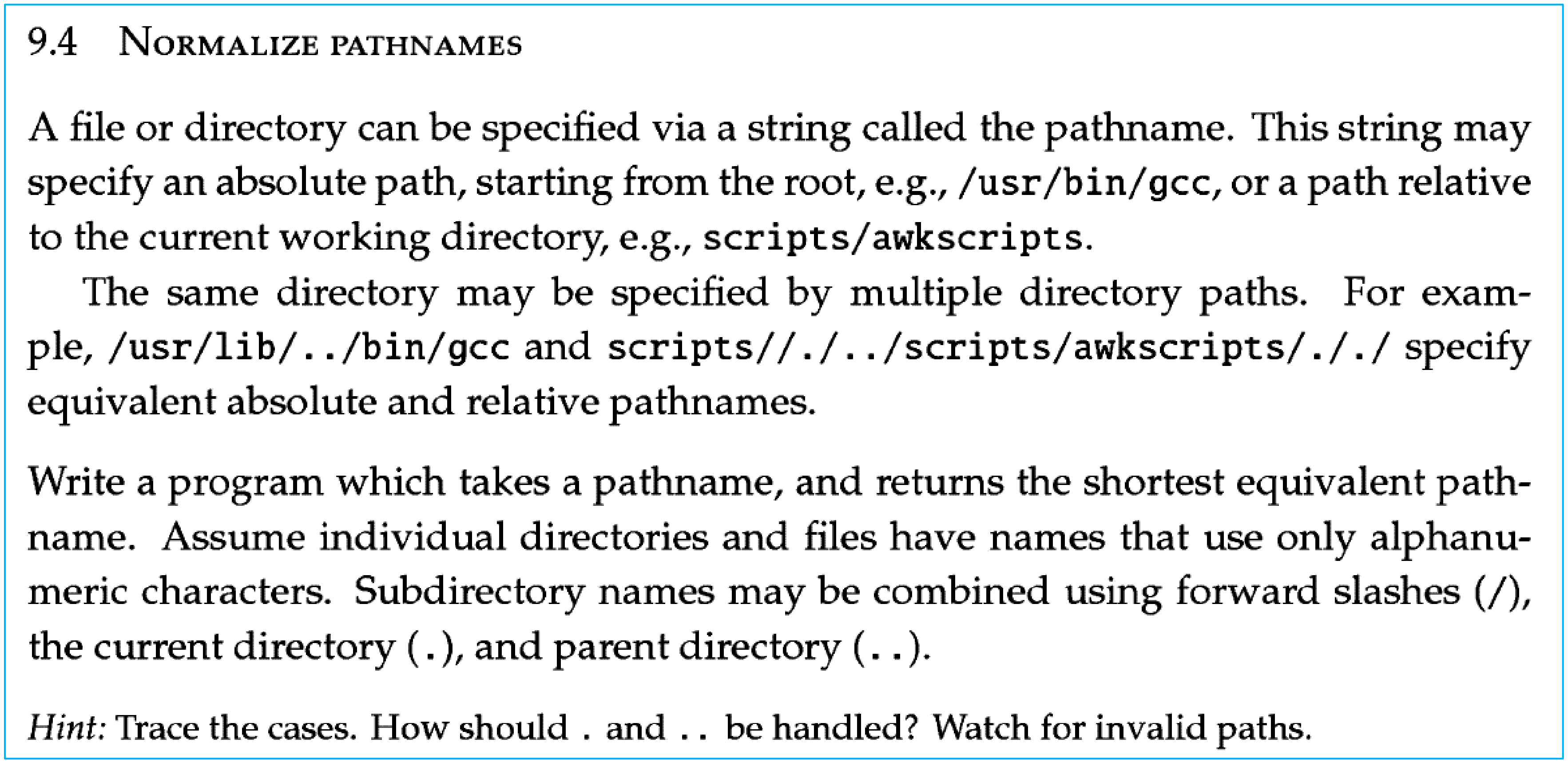


```python
def isValid(s:str)->bool:
    pair = {')':'(', '}':'{', ']':'['}
    stack = []
    for c in s:
        if c in '([{':
            stack.append(c)
        elif stack and stack[-1] == pair[c]:
            stack.pop()
        else:
            return False
    if stack:
        return False
    return True

if __name__ == "__main__":
    print(isValid("([]){()}"))
    print(isValid("([]{)()}"))
```

True

False

경로 압축



9.4 Normalize pathnames

A file or directory can be specified via a string called the pathname. This string may specify an absolute path, starting from the root, e.g., `/usr/bin/gcc`, or a path relative to the current working directory, e.g., `scripts/awkscripts`.

The same directory may be specified by multiple directory paths. For example, `/usr/lib/../bin/gcc` and `scripts//./../scripts/awkscripts/././` specify equivalent absolute and relative pathnames.

Write a program which takes a pathname, and returns the shortest equivalent pathname. Assume individual directories and files have names that use only alphanumeric characters. Subdirectory names may be combined using forward slashes (/), the current directory (`.`), and parent directory (`..`).

Hint: Trace the cases. How should `.` and `..` be handled? Watch for invalid paths.

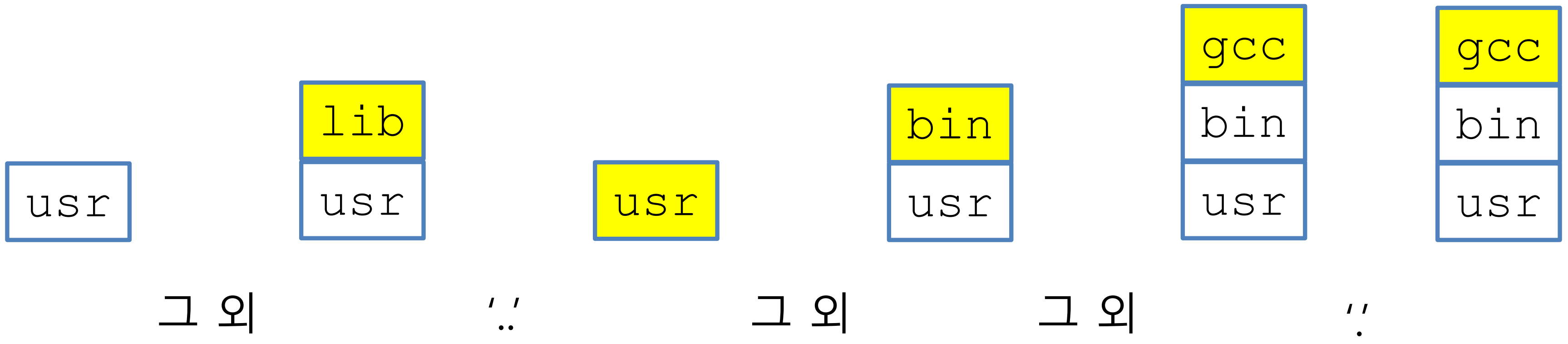
split(), stack 사용

```
'/usr/lib/../bin/gcc/./..//./'
```

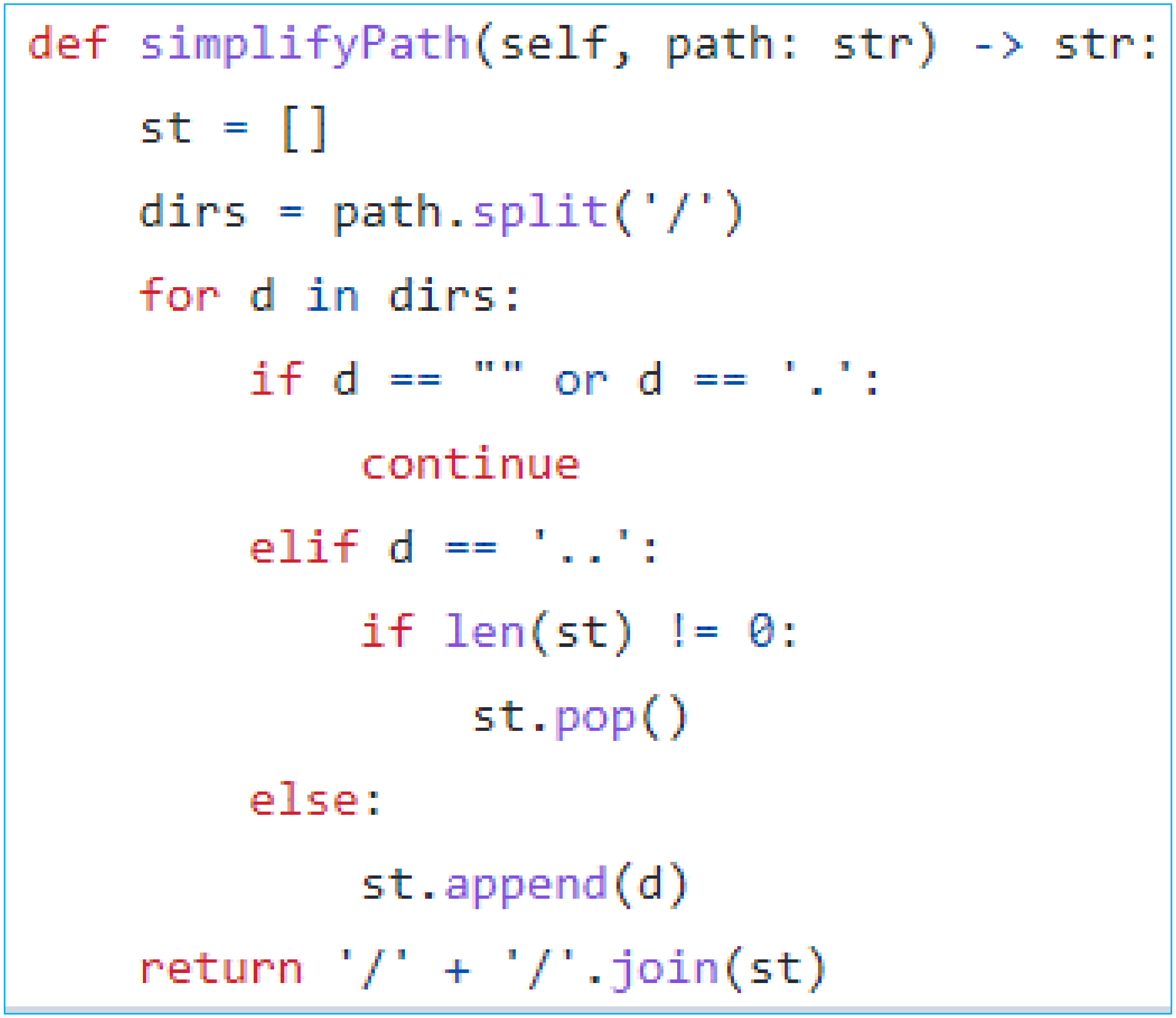
- Split('/')

```
['','usr','lib','..','bin','gcc','.','..','','.','']
```

- '' or '.' → 무시
- '..' → pop()
- 그 외 → push()



split(), stack 사용



```python
def simplifyPath(self, path: str) -> str:
    st = []
    dirs = path.split('/')
    for d in dirs:
        if d == "" or d == '.':
            continue
        elif d == '..':
            if len(st) != 0:
                st.pop()
        else:
            st.append(d)
    return '/' + '/'.join(st)
```

Problems

- **Chapter 6 : `String`**
  - ✓**문제 6.6 – 문장의 모든 단어 뒤집기**
    - **Leetcode #**151. Reverse Words in a String
  - ✓**문제 6.11 – 사인 곡선 형태로 문자열 작성**
    - **Leetcode #**6. Zigzag Conversion
- **Chapter 8 : `Stack`**
  - ✓**문제 8.3 – 괄호가 짝을 이루는가?**
    - Leetcode #20. Valid Parentheses
  - ✓**문제 8.4 – 경로 압축**
    - Leetcode #71. Simplify Path

Q & A