String II

สตริง หรือ สายอักขระ

1. immutability
2. in operator
3. Traversal with while/for loop
4. String and built in function
5. String constant
6. Practice

Immutability [1/2]

- Python strings *cannot* be changed — they are immutable
  - ไม่สามารถเปลี่ยนแปลงค่าของ String ที่มีอยู่แล้วได้

```
>>> word = 'Hello Python'
>>> word[0] = 'J'
    ...
TypeError: 'str' object does not support item assignment
>>> word[2:] = 'py'
    ...
TypeError: 'str' object does not support item assignment
```

- ถ้าต้องการ String ที่ต่างจากเดิมให้สร้าง String ใหม่แทน

```
>>> 'J' + word[1:]
'Jello Python'
>>> word[:2] + 'py’
'Hepy'
>>> word
'Hello Python'
```

Immutability [2/2]

Example

```
word='Hello Python'
print(word)          #Hello Python

#word[0]='J'        #error
#word[2:]='Py'      #error

word='J'+word[1:]
print(word)         #Jello Python

word=word[:2]+'Py'
print(word)         #JePy
```

in operator

- Operator `in` ใช้ตรวจสอบว่า String แรก เป็น *Substring* ของ String ที่สอง หรือไม่
- ผลที่ได้จากการตรวจสอบ คือ `True` / `False`
- หากผลคือ `True` หมายความว่า String แรก เป็น *Substring* ของ String ที่สอง

```
>>> 'y' in 'Python'
True
>>> 'z' not in 'Python'
True
>>> 'no' in 'Python'
False
```

Traversal with while loop

- ในการเขียนโปรแกรม ในหลายๆ กรณี เราจำเป็นต้องเข้าถึงอักขระใน String ทีละตัว (Traversal)
- การใช้ `while` loop

```
02 country = "canada"
03 index = 0
04
05 while index < len(country):
06     letter = country[index]
07     print(letter)
08     index = index + 1
```

Traversal with for loop

- การใช้ for loop
  - for loop with indexes:

```
11 s = "canada"
12 length = len(s)
13 for i in range(length):
14     print(i, s[i])
15
```

  - for loop without indexes

```
17 s = "canada"
18 for c in s:          # similar to for i in range(n)
19     print(c)
```

String-related built-in functions

- `bin(x)`
  - เปลี่ยนเลขจำนวนเต็ม *x* เป็น Binary String

```
>>> bin(4)
‘0b10'
```

- `hex(x)`
  - เปลี่ยนเลขจำนวนเต็ม *x* เป็น Hexadecimal String

```
>>> hex(18)
'0x12'
```

- `oct(x)`
  - เปลี่ยนเลขจำนวนเต็ม *x* เป็น Octal String

```
>>> oct(9)
'0o11'
```

String-related built-in functions

- `ord(c)`
  - คืนค่ารหัส Unicode ของ String ความยาวหนึ่งอักขระ *c*

```
>>> ord('A')
65
```

- `chr(i)`
  - คืนค่า String แทนอักขระที่มีรหัส Unicode เป็นจำนวนเต็ม i

```
>>> chr(65)
'A'
>>> chr(97)
'a'
```

String constants

```
>>> import string
>>> string.digits
'0123456789'
```

string.ascii_letters	'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'
string.ascii_lowercase	'abcdefghijklmnopqrstuvwxyz'
string.ascii_uppercase	'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
string.digits	'0123456789'
string.hexdigits	'0123456789abcdefABCDEF'
string.octdigits	'01234567'
string.punctuation	'!"#$%&¥'()*+,-./:;<=>?@[¥¥]^_`{|}~'
string.printable	digits + letters + punctuation + whitespace
string.whitespace	space + tab + linefeed + return + formfeed + vertical tab (on most systems)

Practice

แบบฝึกหัดข้อ1 ให้เขียนฟังก์ชันที่ชื่อว่า check_vowel ที่รับพารามิเตอร์เป็นข้อความ และคืนค่าเป็นข้อความ

โดยฟังก์ชันจะตรวจสอบตัวอักษรแต่ละตัวหากพบว่าเป็นสระภาษาอังกฤษ จะตัดสระภาษาอังกฤษออก

เช่น

```
Input string: Have a nice day
Hv  nc dy
Input string: HELLO python
HLL pythn
```

template

```
def check_vowel(s):
    ….
    ….
    return result

st=input(“Input string: ”)
ans=check_vowel(st)
print(ans)
```

Practice

แบบฝึกหัดข้อ2 ให้เขียนฟังก์ชันที่ชื่อว่า count_char ที่รับพารามิเตอร์ 2 ตัว ตัวแรกคือข้อความ ตัวที่2คือตัวอักษร และฟังก์ชันนี้จะคืนค่าจำนวนตัวอักษรที่พบในข้อความ

เช่น

```
Input string: Have a nice day
Input character: a
3
```

template

```
def count_char(s,ch):
    ….
    ….
    return result

st=input(“Input string: ”)
ch=input(“Input character: ”)
ans= count_char(st,ch)
print(ans)
```