[숙제07심화] 옛한글 입력 방법 추가

언어와 컴퓨터

2018년 11월 2일 금요일 13시까지

- 소스코드 스크립트 `hangul.py` 및 보고서 `hw07adv_00000.pdf` 파일을 `hw07adv_00000.zip` 파일로 압축하여 제출하라.
- `hangul.py` 파일에 기존에 만든 함수가 들어 있으면 앞으로 한글 처리 모듈로 활용하기 좋다. 없어도 괜찮다.
  - [숙제05] `ishangul()`, `hasbatchim()`, `get_ga()`
  - 10월 17일자 10강 실습 코드 `jamo_name()`, `conjoin()`
  - [숙제07] `decompose()`, `compose()`
- 보고서는 문제 해결 방법, 코드 설명, 테스트 실행 결과 등을 포함하여 작성하라.

1 옛한글 입력 방법 추가하기

1.1 목표

이 과제의 목표는 '**ㅂㅅㄱㅜㄹ**'이나 '**ㅂㅅ굴**'을 받아 '**ᄲᅮᆯ**'[1]을 돌려줄 수 있도록 10강 실습 코드의 함수 `conjoin()`을 수정하는 것이다.

1.2 배경

10강 실습 코드에서는 옛한글을 입력하기 위해 `'Z'` ↦ `'PANSIOS'`, `'A'` ↦ `'ARAEA'` 등 영문 대문자 하나를 옛한글 자모의 이름에 대응시키는 방법을 사용하였다.

```
>>> jamo_name('ㅈ')
'CIEUC'
```

```
>>> jamo_name('ㅏ')
'A'
```

```
>>> jamo_name('Z')
'PANSIOS'
```

```
>>> jamo_name('A')
'ARAEA'
```

옛한글에는 ㅿ나 ㆍ 등과 같이 현대 한글에 대응하는 것이 없는 자모 외에도, ㅃ처럼 서로 다른 자음을 나란히 붙여 쓰는 합용 병서가 많이 있다. 이 경우에도 `'T'` ↦ `'PIEUP-SIOS-TIKEUT'` 등을 추가하여 함수 `jamo_name()`을 수정하면 되지만, 사람의 입장에서는 각각의 자모를 차례로 입력하는 것이 자연스럽다. 예를 들어 '**ᄲᅮᆯ**'이라는 음절을 만들고 싶다면 키보드에서 ㅂ, ㅅ, ㄱ, ㅜ, ㄹ을 차례로 입력하는 것이다.

1.3 방법

'**ㅂㅅ굴**'을 '**ᄲᅮᆯ**'로 바꾸는 과정을 다섯 단계로 생각할 수 있다.[2]

(예 1) '**ㅂㅅ굴**' → '**ㅂㅅㄱㅜㄹ**' → ('**ㅂㅅㄱ**', '**ㅜ**', '**ㄹ**') → ('**ㅃ**', '**ㅜ**', '**ㄹ**') → '**ᄲᅮᆯ**'

1. 인자를 받아서
2. 자모의 문자열로 분해하고
3. (**초성 자음(군)**, **중성 모음(군)**, **종성 자음(군)**) 꼴의 튜플을 얻어서
4. 자음군이나 모음군을 합용 병서로 바꾼 뒤
5. 초성, 중성, 종성을 연결하여 반환한다.

[1] 폰트에 따라 '**ᄲᅮᆯ**'로 표시될 수도 있다.
[2] 더 효과적인 방법이 떠오르면 절차를 바꾸어도 괜찮다.

jamo_name()을 정의할 때 'H' ↦ 'YEORINHIEUH' 규칙을 추가하면 아래와 같은 것도 가능하다.

(예 2) 'ㅎAㄹH' → 'ㅎAㄹH' → ('ㅎ','A','ㄹH') → ('ㅎ','ㆍ','ㅀ') → '퓧'

(1→2)는 [숙제07]에서 만든 decompose() 함수, (4→5)는 문자열 메서드 ''.join()을 이용하면 된다. 이 과제에서 해결해야 하는 문제는 (2→3)과 (3→4)이다. 각 단계에서 한 개 이상의 함수를 새로 만들 수 있다.

(2→3) 힌트 문자열에서 모음(군)을 찾아내면 앞뒤의 초성과 종성이 자동으로 결정된다.

(3→4) 조건 및 힌트 'ㅂㅅㄱ' ↦ 'ㅃ' 등을 처리할 때, 가능한 경우를 딕셔너리에 하나하나 만들어 넣기보다는 일반화할 수 있는 방법을 생각하라. 딕셔너리는 ㅿ, ㆍ, ㆆ 등 현대 한글로 표현할 수 없는 경우로 한정하라. 합용 병서 자모[3]의 유니코드 이름을 살펴보고 일정한 규칙을 찾아내면 좋다.

```
>>> from unicodedata import lookup
>>> lookup('HANGUL LETTER SIOS-TIKEUT')
'ㅶ'
>>> lookup('HANGUL CHOSEONG PIEUP-SIOS-KIYEOK')
'ㅃ'
>>> lookup('HANGUL JUNGSEONG YA-O')
'ㅛ'
>>> lookup('HANGUL JONGSEONG RIEUL-YEORINHIEUH')
'ퟶ'
```

시험 기간에 무리하지 말고 할 수 있는 데까지 해 보라. 이 과제에서는 예외를 고려하지 말고 처리 가능한 인자만 받는다고 가정하라.

1.4 테스트

conjoin() 함수로 아래와 같은 것이 가능하다면 과제를 훌륭하게 완성한 것이므로 자축하자. 가능하지 않더라도 고생이 많았으므로 맛있는 것을 먹자.

```
>>> conjoin('ㅍㅂA')
'퍕'
```

```
>>> conjoin('ㅇAㅣ')
'읭'
```

```
>>> conjoin('ㅅ쎤Z')
'쎐'
```

잘 알려진 문장으로 테스트해 볼 수도 있다.

```
if __name__ == '__main__':
    hm = f"제{conjoin('ㅂㄷ')}들시러펴디몯{conjoin('ㅎAㄹH')}노미하니라"
    db = f"고{conjoin('ㅅAㄴ')}수리{conjoin('ㅂㅅ굴')}{conjoin('ㄱA')}티{conjoin('ㄷA')}닐"
    print(hm)
    print(db)
```

```
>>> print(hm) # 훈민정음
제쁘들시러펴디몯홇노미하니라
```

'자기 뜻을 실어 펴지 못할 사람이 많다'

```
>>> print(db) # 두시언해
고손수리뜔ㄱ티둒닐
```

'향기로운 술이 꿀 같이 다니'

conjoin() 수정을 끝내지 못한 경우 중간 단계에서 만든 함수를 테스트해 보라.

[3]합용 병서의 정의는 자음에만 해당하지만, 유니코드에서는 모음도 같은 방식으로 조합되어 있으므로 합용 병서라고 치자.