**< 수와 문자열 >**

* **비트 연산자**

1. 비트 논리 연산자가 아닌 것을 모두 고르시오.

**① and**

**② or**

**③ not**

④ &

⑤ |

**⑥ !**

**⑦ >>**

**⑧ <<**

⑨ ^

⑩ ~

1. 다음 비트 연산자의 연산 결과를 16진수로 예측하세요. (손으로 풀어보세요!)

* 0xFF00 & 0x00FF (**0x0000 (and 연산)**)
* 0xFF00 | 0x00FF (**0xFFFF (or 연산)**)
* 0xFF00 ^ 0x00FF (**0xFFFF (xor 연산)**)

1. 다음 비트 시프트 연산자의 활용 결과를 예측하세요. (손으로 풀어보세요!)

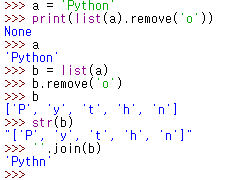
a = 100; a = a << 100; a = a >> 100; print(a)

: **왼쪽으로 100칸 만큼 비트를 이동했다가 다시 오른쪽으로 100칸 만큼 비트를 이동했기 때문에 원래 값인 100이 출력된다.**

* **리스트 메소드들**

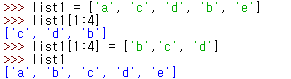
1. 리스트 메소드를 이용해 다음 문자열로부터 모음 ‘o’를 제거한 후, 다시 제거 결과(‘Pythn’)를 출력하세요.

a = ‘Python’



1. list1 = ['a', 'c', 'd', 'b', 'e']라는 리스트를 list1 = ['a', 'b', 'c', 'd', 'e']로 만들어 보세요.

5-1) 인덱스를 이용해 값을 바꾸기 (하나씩 or 한번에 슬라이싱)



5-2) 리스트 메소드 이용

>>> list1 = ['a', 'c', 'd', 'b', 'e']

>>> list1

['a', 'c', 'd', 'b', 'e']

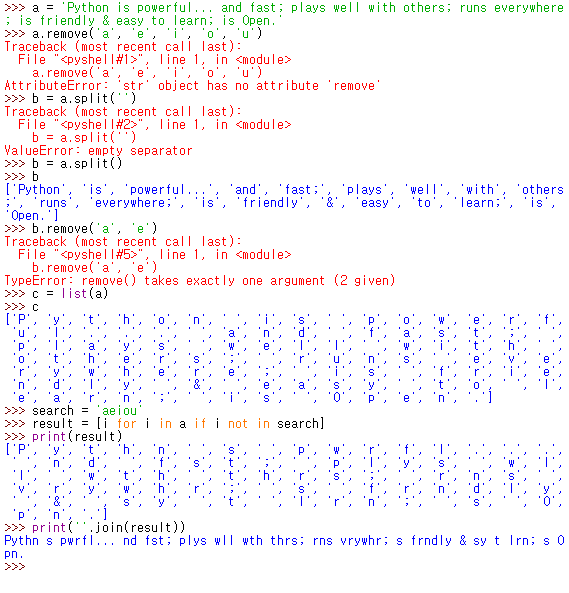
>>> list1.sort()

>>> list1

['a', 'b', 'c', 'd', 'e']

(심화) 리스트 메소드를 이용해 다음 문장(문자열)으로부터 모음('aeiou')을 모두 제거하세요.

'Python is powerful... and fast; plays well with others; runs everywhere; is friendly & easy to learn; is Open.'



****