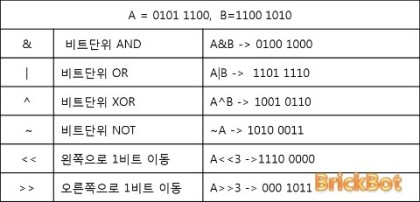
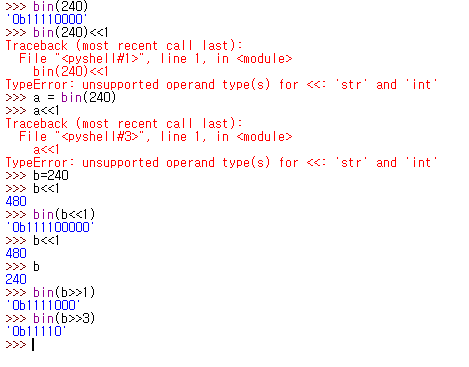
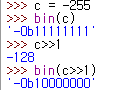
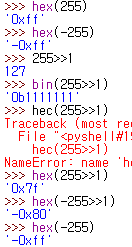
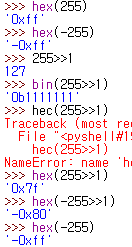
**< 비트, 리스트 >**

1. **비트란**

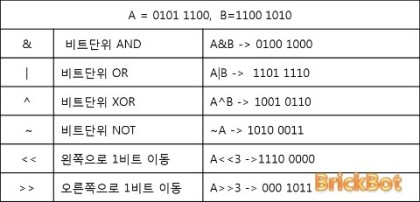
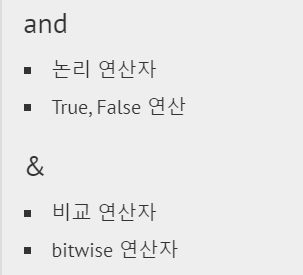
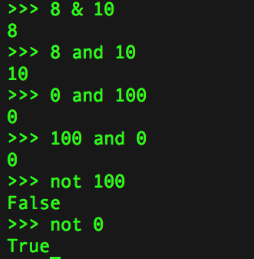
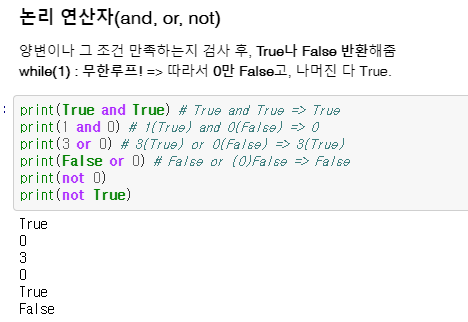
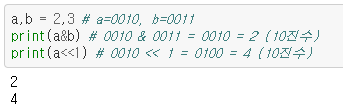
* 컴퓨터 : 0과 1로 구성된 비트 데이터를 다루는 기계
* 파이썬에서는 ‘바이트’ 단위. 1Byte = 8Bit
* 그래서 파이썬에서는 비트 수준에서 연산을 잘 하지 않음.

1. **비트 연산자 - 시프트 연산자**



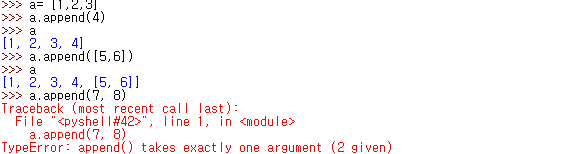
* 
* **\*\*\*여기서 -255가 비트로 표현하면 왜 저렇게 되는지는 pass! 부동소수점 관련 얘기이므로 비트 연산과 관련 없으므로 강의 상황 봐서 추후에 한번에 진행! \*\*\***
* ****🡺 이 두 결과에 대해선 나도 잘ㅠㅠ
* 시프트 연산의 의미 : <<(왼쪽 시프트 연산자)는 2배씩, 반대는 1/2배씩
* 예를 들면, 11(2)는 1x1 + 1x2인데, 110(2)는 0x1 + 1x2 + 1x2x2 = 2x(11(2))! 2배!
* 참고로, hex(), bin() 함수 등은, 출력하게 되면 0x, 0b 등의 접두사가 붙어서 출력되므로, **정수를 문자열**로 변환해주는 기능!
* 
* 
* *Q. p.91 예제에서, hex(a), hex(a<<1), hex(a<<2) 까지는 계속 2의 배수 씩 가다가 왜 hex(a<<5)는 2의 제곱꼴이 아니라 20이 나왔을까??*
* hex는 16진수. 그냥 a<<5라고 했으면 계속 2의 배수 꼴로 나왔을 건데, 16진수로 바꿨기 때문에 hex(a<<3)까지는 hex(8)이므로 9를 안넘어서 걍 2배 꼴로 출력됐을 건데, a<<4부터는 9를 넘어가기 때문에 그렇게 표현 안됨! 외우지 말 것!
* 따라서 hex(a<<5) == hex(32) = 0x20 (나눠보면 몫이 2, 나머지가 0)
* 255>>1 == 127 (255를 2로 나눈 **몫**)

1. **비트 연산자 - 논리 연산자**

* 결과 : True or False 반환함
* 각 비트에 대해서 수행 (각 비트에 해당하는 0, 1은 각각 거짓, 참으로 해서 논리 연산을 하는 것)
* 정확히는, 0만 false, 나머지는 다 true인데 보통 1을 대표값으로 생각
* 
* 
* **\*\*\*일반 논리 연산자(and) vs 비트 논리 연산자(&) 다름!! \*\*\***
* 
* 
* 
* 
* 배타적 논리합 연산자(XOR)
* 

1. **리스트와 리스트 메소드들**

* append() : 무조건 한 인자만 받을 수 있음!



* extend() : 여러 값들을 이어 붙이고 싶을 때. [리스트1] + [리스트2]. **‘+’** 연산자와 같은 역할
* pop() : 리스트의 **마지막** 요소를 뽑아내서 제거
  + 만약, 특정 위치의 요소를 제거하고 싶을 때? 🡺 인덱스를 인자로!
  + ex) pop(2) : 인덱스 2에 위치하는 인자를 제거하라.
* sort() : 정렬해줌. 디폴트는 오름차순인데, 내림차순은 a.sort(reverse = True)
* [ 정리 ]
* 책에 남은 연습문제들 풀고 마무리

****