

파이썬 프로그래밍

11차시

비트이동연산자
연산자 우선순위
프로젝트 Lab



⚠ 학습개요

- ... 비트 이동 연산자
- ... 연산자 우선 순위
 - 윤년 검사
- ... 프로젝트 Lab
 - 할인율에 따른 가격
 - 문자열의 회문(palindrome) 검사

⚠ 학습목표

- ... 비트 이동 연산자를 사용할 수 있다.
- ... 연산자 우선 순위를 알 수 있다.
- ... 윤년 계산을 할 수 있다.
- ... 할인율에 따른 가격을 출력하는 프로그래밍을 할 수 있다.
- ... 문자열의 회문(palindrome) 검사를 수행하는 프로그래밍을 할 수 있다.

Chapter 1.

비트 이동 연산자

P Y T H O N P R O G R A M M I N G

⚠ 비트 이동 연산자 >>와 <<

✦ 비트 단위로 뒤 피연산자인 지정된 횟수만큼 이동시키는 연산자



[그림11-1] 8비트에서 비트 이동 연산 수행의 이해

⚠ 비트 이동 연산자 >>와 <<

```
a = 0b00010111
print('10진수 {0:3d}, 2진수 {0:08b}'.format(a))
print('10진수 {0:3d}, 2진수 {0:08b}'.format(a >> 1))
print('10진수 {0:3d}, 2진수 {0:08b}'.format(a // 2))
print('10진수 {0:3d}, 2진수 {0:08b}'.format(a >> 2))
print('10진수 {0:3d}, 2진수 {0:08b}'.format(a // 2**2))
```

실행
결과

| | | | |
|------|-----|-----|----------|
| 10진수 | 23, | 2진수 | 00010111 |
| 10진수 | 11, | 2진수 | 00001011 |
| 10진수 | 11, | 2진수 | 00001011 |
| 10진수 | 5, | 2진수 | 00000101 |
| 10진수 | 5, | 2진수 | 00000101 |

Chapter 2.

연산자 우선 순위

P Y T H O N P R O G R A M M I N G

⚠ 연산자 우선순위

✦ 지수, 단항, 산술, 비트이동, 비트논리, 관계, 비교(멤버십 검사), 논리 연산 순서

| 연산자 | 설명 |
|--|--|
| (expressions...), [expressions...], {key: value...}, {expressions...} | 결합(binding)또는 괄호 친 표현식, 리스트, 딕셔너리, 집합 표현 |
| x[index], x[index: index], x(arguments...), x.attribute | 서브스크립션, 슬라이싱, 호출, 어트리뷰트 참조 |
| await x | 어웨이트 표현식 |
| ** | 거듭제곱 |
| +x, -x, ~x | 양, 음, 비트 NOT |
| +, - | 덧셈과 뺄셈 |
| <<, >> | 이동(시프트) |
| & | 비트 AND |
| ^ | 비트 XOR |

⚠ 연산자 우선순위

✦ 지수, 단항, 산술, 비트이동, 비트논리, 관계, 비교(멤버십 검사), 논리 연산 순서

| 연산자 | 설명 |
|--|-------------------------|
| | 비트 OR |
| in, not in, is, not is, <, <=, >, >=, !=, == | 멤버십 검사와 아이덴티티 검사, 관계 연산 |
| not x | 논리 NOT |
| and | 논리 AND |
| or | 논리 OR |
| if-else | 조건 표현식 |
| lambda | 람다 표현식 |
| := | 대입 표현식 |

⚠ 다양한 연산과 연산자 우선순위를 고려한 윤년 검사

✦ 다음 두 가지 조건 중 하나를 만족하면 윤년

- 4로 나뉘지고 100으로는 나뉘지지 않는 해(연도)
- 400으로 나뉘지는 해(연도)

⚠ 다양한 연산과 연산자 우선순위를 고려한 윤년 검사

[코딩실습] 표준 입력의 연도가 윤년인지 검사해 출력

난이도 응용

```
1. year = int(input('윤년을 검사할 연도 입력 >>'))
2. print('입력할 연도: %d' % year)
3. cond1 = year % 4 == 0
4. cond2 = year % 100 != 0
5. cond3 = year % 400 == 0
6. result1 = cond1 and cond2 or cond3
7. print('개별 검사 {} and {} or {}: {}'.format(cond1, cond2, cond3, result1))
8. result2 = year % 4 == 0 and year % 100 != 0 or year % 400 == 0
9. print('통합 검사: %s' % result2)
```

결과

윤년을 검사할 연도 입력 >> 2020
입력한 연도: 2020
개별 검사 True and True or False: True
통합 검사: True

윤년을 검사할 연도 입력 >> 2021
입력한 연도: 2021
개별 검사 False and True or False: False
통합 검사: False

Chapter 3.

프로젝트 Lab

P Y T H O N P R O G R A M M I N G

⚠ 프로젝트 Lab 1

할인율에 따른 할인 가격

난이도 응용

다음 표와 같은 할인 조건인 경우에 총 가격을 입력 받아 원 가격, 할인된 가격, 할인율, 할인액을 출력하는 프로그램을 작성하자.

| 조건 | 총액 할인율 |
|------------------------|--------|
| 10,000원 이상, 20,000원 미만 | 1% |
| 20,000원 이상, 40,000원 미만 | 2% |
| 40,000원 이상 | 4% |

문제 이해 (Understanding)

총 가격에 따른 할인율을 계산하는 것이 가장 중요하다.
위 조건을 관계 연산식으로 만들고 이 식으로 할인율을 계산한다.

⚠ 프로젝트 Lab 1

할인율에 따른 할인 가격

난이도 응용

다음 표와 같은 할인 조건인 경우에 총 가격을 입력 받아 원 가격, 할인된 가격, 할인율, 할인액을 출력하는 프로그램을 작성하자.

설계 (Design)

알고리즘(Algorithm)

- ① 총 가격(price)을 입력 받는다.
 - 변수 price에 입력,
함수 input()과 int() 사용
- ② 관계 연산식의 결과인 True와 False가 산술 연산에서 1과 0으로 사용되는 것을 활용한다.
 - $(10000 \leq \text{price} < 200000)$
*할인율/100 등을 사용
- ③ 할인액과 할인된 가격을 계산해 출력한다.

표준 입출력 샘플

총 가격(원 가격)입력 >> 50000
원 가격: 50000 할인된 가격: 48000.0
할인율: 0.04 할인액: 2000.0

⚠ 프로젝트 Lab 1

+ 구현(Implementation)

```
1. price = int(input('총 가격(원 가격)입력 >> '))
2.
3. rate1 = (10000 <= price < 20000) * 1 / 100
4. rate2 = (20000 <= price < 40000) * 2 / 100
5. rate3 = (40000 <= price) * 4 / 100
6. rate = rate1 + rate2 + rate3
7.
8. discount = price * rate
9. discPrice = price - discount
10. print('원 가격:' price, '할인된 가격:', discPrice)
11. print('할인율:' rate, '할인액:', discount)
```

⚠ 프로젝트 Lab 1

+ 구현(Implementation)

테스트와
실행 결과
(Testing &
Simulation)

총 가격(원 가격) >> 9000
원 가격: 9000 할인된 가격: 9000.0
할인율: 0.0 할인액: 0.0

총 가격(원 가격) >> 30000
원 가격: 30000 할인된 가격: 29400.0
할인율: 0.02 할인액: 600.0

공유(Share)

할인 조건은 구간과 할인율이 달라질 수 있다. 위 구간과 할인율을 변화 시킨 후 프로그램을 수정해 보자.
코드와 비교해 보고 결과가 맞는지 확인해 보자.

실행오류



프로젝트 Lab 2

할인율에 따른 할인 가격

난이도 응용

표준 입력으로 3개의 단어를 콤마로 구분해 입력 받자. 3개의 단어를 추출해 각 단어와 역순의 단어를 출력한다. 그리고 그 단어가 회문(palindrome)인지를 판단하는 논리 값도 출력하는 프로그램을 작성하자.

- 회문 또는 팰린드롬(palindrome)은 거꾸로 읽어도 제대로 읽는 것과 같은 문장이나 낱말을 말한다. 예를 들면 level, 기러기 등이다.

문제 이해 (Understanding)

표준 입력에서 콤마로 구분된 단어 토큰을 분리해 낸다.
분리된 단어를 부분 문자열 참조 방법을 사용해 역순으로 출력한다.
분리된 단어가 회문인지를 부분 문자열 참조 방법으로 검사해 논리 값을 출력한다.



프로젝트 Lab 2

할인율에 따른 할인 가격

난이도 응용

표준 입력으로 3개의 단어를 콤마로 구분해 입력 받자. 3개의 단어를 추출해 각 단어와 역순의 단어를 출력한다. 그리고 그 단어가 회문(palindrome)인지를 판단하는 논리 값도 출력하는 프로그램을 작성하자.

- 회문 또는 팰린드롬(palindrome)은 거꾸로 읽어도 제대로 읽는 것과 같은 문장이나 낱말을 말한다. 예를 들면 level, 기러기 등이다.

설계 (Design)

알고리즘(Algorithm)

- ① 3개의 단어를 콤마로 구분해 입력 받는다.
 - 콤마 다음에 빈 공백을 넣을 것을 고려해 먼저 메소드 `replace()`로 공백을 제거한 후, 다시 메소드 `split(',')`으로 3개의 단어를 빼냄
- ② `word`에 저장된 단어를 역순으로 출력한다.
 - `word[::-1]`
- ③ 단어와 역순의 단어가 동일한지를 검사한다.
 - `word == word[::-1]`

표준 입출력 샘플

콤마로 구분된 단어 4개 입력 >> 여보보여,
파이썬, level

단어: 여보보여, 역순: 여보보여, 회문: True

단어: 파이썬, 역순: 썬이파, 회문: False

단어: level, 역순: level, 회문: True

⚠ 프로젝트 Lab 2

+ 구현(Implementation)

```
1. # 단어를 추출해 회문인지 검사
2. str = input('콤마로 구분된 단어 3개 입력 >> ')
3. str = str.replace(' ', '')
4. w1, w2, w3 = str.split(',')
5.
6. print('단어:{}, 역순:{}'.format(w1, w1[::-1]), (w1 == w1[::-1]))
7. print('단어:{}, 역순:{}'.format(w2, w2[::-1]), (w2 == w2[::-1]))
8. print('단어:{}, 역순:{}'.format(w3, w3[::-1]), (w3 == w3[::-1]))
```

⚠ 프로젝트 Lab 2

+ 구현(Implementation)

테스트와
실행 결과
(Testing &
Simulation)

콤마로 구분된 단어 3개 입력 >> 기러기, 파이썬, level
단어: 기러기, 역순: 기러기, 회문: True
단어: 파이썬, 역순: 뉼이파, 회문: False
단어: level, 역순: level, 회문: True

공유(Share)

실행에서 콤마 2개로 3개의 단어가 입력돼야 한다. 그렇지 않으면 실행 오류가 발생한다.
친구의 코드와 비교해 보고 구분자인 콤마를 빼고 공백으로 분리해 입력 받을 수 있는
프로그램으로 수정해 보자.

실행오류

SUMMARY



동양미래대학교
DONGYANG MIRAE UNIVERSITY

⚠ 비트 이동 연산자

⚠ 연산자 우선 순위

... 윤년 검사

SUMMARY



⚠ 프로젝트 Lab

- ... 할인율에 따른 가격
- ... 문자열의 회문(palindrome) 검사