

**기초프로그래밍 강의 (Python3)**

# **Workbook**

김 광 교수 (한동대학교 전산전자공학부)  
[kkim@handong.edu](mailto:kkim@handong.edu)

# 서 문

이 워크북은 컴퓨터 프로그래밍을 배우는 데에 있어서 기초가 되는 논리들을 이해하고 이를 활용한 프로그램을 제작할 수 있도록 훈련하기 위해 제작된 교재이다. 이를 위해 기초부터 응용까지 단계별로 영역을 나누어 각 단계마다 필요한 논리를 간단한 예제를 통해 설명한다. 그리고 각 단계마다 반복적인 실습문제들을 제시하여 풀게 함으로써 프로그래밍 논리를 차근차근 습득해 나갈 수 있도록 돋고 있다.

본 교재에서는 Python 3 언어를 기준으로 설명하고 있지만, 여기에서 다루는 프로그래밍 논리 연습은 컴퓨터 프로그래밍을 공부하는 학생들에게 특정 프로그래밍 언어에 국한되지 않는 공통적인 프로그래밍 논리를 스스로 제작할 수 있도록 하는데 도움이 될 것이다. 따라서 앞으로 어떤 언어를 사용하더라도 본 교재에서 습득한 프로그램 해결 능력을 활용하면 빠르게 개발 능력을 키워 갈 수 있을 것이다.

프로그래밍 논리 제작 수준에 따라 영역 1부터 영역 3까지 나누었으며, 각 영역에는 3~4개의 단계로 나누어 각 단계마다 필요한 프로그래밍 논리 문제들을 예제와 함께 간단히 설명한다. 그리고 각 단계마다 제시되는 실습 문제들은 주어진 문제와 조건, 사용해야 하는 변수나 함수의 이름과 형식을 준수하고, 제시된 실행 예시 화면을 참고하여 문제를 해결하기 바란다. 본 교재는 일반적인 Python 언어에서 다루는 구체적인 문법들을 설명하고 있지는 않다. 그렇기 때문에 본 교재의 실습 문제를 풀 때에는 다른 Python 교재나 인터넷의 관련 사이트 정보를 충분히 활용하기 바란다.

그리고 실습문제를 풀 때에는 제시된 변수만을 사용해서 풀어보기를 권한다. 임의로 변수를 추가해서 만들기보다는 주어진 조건을 최대한 지키면서 문제를 해결할 수 있다면 훨씬 유익한 공부가 될 것으로 기대한다.

\*프로그램 제작 도구 :

<https://www.jetbrains.com/pycharm-edu/download/#section=windows-version>

초판 작성 : 2014년 2월

파이썬 3.x 내용 반영 : 2016년 7월

저자.

# 목 차

## 영역 1. 프로그래밍 제어구조 기초

[Step A] 변수의 사용과 데이터 입출력 .....	5
[Step B] 단순 조건문 사용하기 .....	12
[Step C] 복합 조건문 사용하기 .....	24
[Step D] 반복문 사용하기 .....	37

## 영역 2 : 프로그래밍 제어구조 응용

[Step E] 복합 반복문 사용하기 .....	52
[Step F] 리스트 사용하기 .....	60
[Step G] 조건과 반복을 활용하는 응용 예제 해결하기 .....	72
[Step H] 파이썬에서 함수 사용하기 .....	81
[Step H'] 딕셔너리 사용하기 .....	97

## 영역 3 : 함수 만들어 사용하기

[Step I] 파라미터 또는 리턴값만 있는 함수 만들기 .....	100
[Step J] 파라미터와 리턴 값이 모두 있는 함수 만들기 .....	110
[Step K] 파일 사용하기 .....	123

## 실습문제 목록

## 영역 1. 프로그래밍 제어구조 기초

영역 1에서는 프로그래밍의 가장 기본이 되는 변수의 사용과 입출력 처리 방법, 조건문과 반복문을 통한 제어구조의 기초를 연습한다. 영역 1은 다음과 같이 4개의 단계로 구성된다.

Step A : 변수의 사용과 데이터 입출력

Step B : 단순 조건문 사용하기

Step C : 복합 조건문 사용하기

Step D : 반복문 사용하기

## [Step A] 변수의 사용과 데이터 입출력

### ○ 변수의 사용

프로그램을 제작할 때 어떤 종류이건 데이터나 값을 다루기 위해서는 변수를 사용해야 한다. 변수를 사용할 때에는 다루어야 하는 값이 숫자(정수와 실수)인지 문자인지에 따라 변수의 형식을 잘 구별해서 결정해야 하므로 프로그래밍 언어에서 사용가능한 변수 형식을 미리 숙지하고 있어야 한다. 파이썬 언어의 경우 주로 사용하는 데이터 형식은 다음과 같다.

데이터 형식	설명
숫자	정수, 실수, 복소수 등의 숫자를 다룬다.
문자열	1개 이상의 글자로 이뤄진 문장을 다룬다. 따옴표로 둘러쌓아 표현한다.
튜플	여러 개의 원소를 포함하는 모음을 다룬다. 소괄호로 둘러쌓아 표현한다.
리스트	여러 개의 원소를 포함하는 모음을 다룬다. 대괄호로 둘러쌓아 표현한다.
딕셔너리	키와 값의 쌍으로 된 데이터들을 포함하는 모음을 다룬다. 중괄호로 둘러쌓아 표현한다.

그리고 변수의 이름을 정해주어야 하는데 영문과 숫자를 적당히 사용하되 변수의 용도를 쉽게 파악할 수 있도록 의미 있는 이름으로 정하는 것이 좋다. 다음과 같은 이름은 좋은 변수 이름이다.

apple_count	# 사과 개수
keyChar	# 입력받은 문자

프로그램에서 변수를 다루는 가장 기본적인 방법은 변수에 값을 할당하는 것이다. 예를 들어 1개에 1,000원씩 판매하는 사과 15개의 가격을 계산하라고 한다면 누구나 1,000원 곱하기 15개인 15,000 원을 계산할 수 있을 것이다. 그렇다면 이 계산을 컴퓨터에게 시켜본다면 어떻게 해야 할까? 적절하게 이름붙인 변수들에게 값을 할당해주어야 한다. 즉 사과 개수에는 15를 할당하고, 전체 사과의 가격은 사과 개수에 1,000원을 곱한 값을 할당하는 것이다.

apple_count = 15	# 사과 개수에 15를 할당한다.
price = apple_count * 1000	# 전체 가격에는 사과 개수와 1000을 곱한 값을 할당한다.

하나만 더 생각해보자. 만일 사과 1개의 가격이 1,200원으로 변경된다면 어떻게 해야 할까? 다음과 같이 프로그램을 변경하면 쉽게 해결할 수 있어 보인다.

price = apple_count * 1200	# 전체 가격에는 사과 개수와 1200을 곱한 값을 할당한다.
----------------------------	------------------------------------

하지만 만약 사과 1개의 가격이 앞으로도 변동될 가능성이 있을 때에는 다음과 같이 사과 1개당 단가를 별도의 변수 `unit_price`를 만들어 사용하는 것이 더 좋은 방법이다.

```
unit_price = 1200          # 단가에 1200을 할당한다.  
price = apple_count * unit_price  # 전체 가격에는 사과 개수와 단가를 곱한 값을 할당한다.
```

## ○ 데이터 입출력

프로그램에서 변수에 값을 할당하기 위해서 사용자로부터 입력을 받아야 하는 경우가 있다. 그리고 사용자에게 프로그램의 진행 과정이나 결과를 모니터 화면을 통해 출력해주어야 할 필요가 있다. 이런 경우에 데이터 입력, 출력 기능을 사용하게 된다.

먼저 화면에 원하는 문장을 출력하는 구문은 다음과 같다.

```
print("안녕하세요?")      # 화면에 "안녕하세요?"라는 문장을 출력한다.  
print("사과의 개수는 몇 개입니까?")
```

특정 변수의 값을 화면에 출력하는 방법은 다음과 같다. 이 때 주의해야 할 것은 출력하려는 변수를 적절한 위치에 넣어 주어야 한다.

```
print("사과의 개수는 모두", apple_count, "개입니다.")
```

사용자에게 특정 변수에 할당할 값을 입력받는 방법은 다음과 같다.

```
apple_count = int(input("사과의 개수는 모두 몇 개 입니까? "))  
# 사용자가 입력한 값을 apple_count에 정수형식(int)으로 할당한다.
```

그러면 이제 몇 개의 변수를 사용하여 간단한 입출력을 수행하는 프로그램을 만들어보자. 문제는 사과의 단가와 개수를 입력받아 전체 사과의 가격을 계산하는 것이다. 이를 해결하기 위해서 다음과 같은 순서대로 프로그램이 실행되어야 할 것이다.

1. 사과 1개의 가격을 입력받는다.
2. 사과 개수를 입력받는다.
3. 전체사과의 가격을 계산한다.
4. 계산한 가격을 출력한다.

위의 순서에 따라 프로그램 코드를 만들면 다음과 같다.

```

unit_price = int(input("사과 1개의 가격은 얼마입니까? "))
# 사용자가 입력한 값을 unit_price에 할당한다.

apple_count = int(input("사과의 개수는 모두 몇 개 입니까? "))
# 사용자가 입력한 값을 apple_count에 할당한다.

price = apple_count * unit_price # 전체 사과의 가격을 계산한다.

print("전체 사과의 가격은 ", price, "원입니다.") # 전체 사과의 가격을 출력한다.

```

한글이 지원되지 않는 프로그램제작 도구에서 한글을 사용한 파이썬 코드는 프로그램 첫 줄에 문자 코딩 정보를 입력해주어야 한다. 이렇게 만들어진 파이썬 프로그램은 다음과 같다.

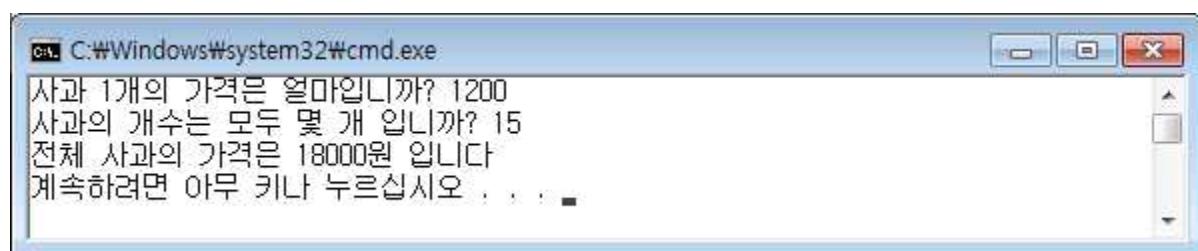
```

# -*- coding: utf-8 -*-
# 예제 ex_A.py

unit_price = int(input("사과 1개의 가격은 얼마입니까? "))
apple_count = int(input("사과의 개수는 모두 몇 개 입니까? "))
price = apple_count * unit_price
print("전체 사과의 가격은 ", price, "원입니다.")

```

다음 화면은 이 프로그램의 실행 결과 예시를 보인 것이다.



## ○ 실습 문제

### [A01] 나이 계산

태어난 년도를 입력받은 후, 이 값을 이용하여 나이를 계산하고 출력하라.

단, 나이 =  $2012 - \text{태어난 년도} + 1$  로 계산한다.

변수는 다음과 같이 사용하라.

```
birth_year      # 태어난 년도  
age            # 나이
```



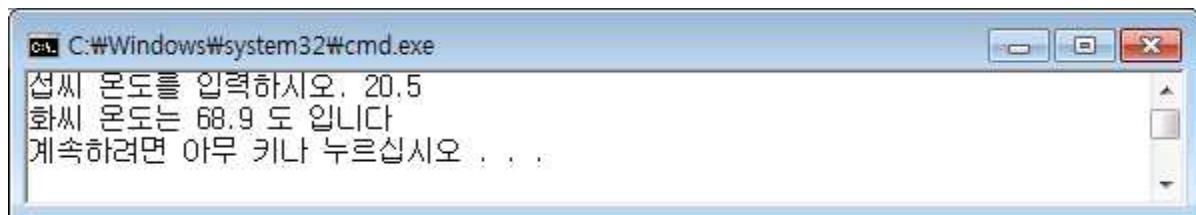
### [A02] 온도 변환

섭씨 온도를 입력받아 이 값을 화씨온도로 변환하여 출력하라.

단, 화씨 온도 = 섭씨 온도 \* 1.8 + 32 로 계산한다.

변수는 다음과 같이 사용하라.

```
c_degree    # 섭씨 온도  
f_degree    # 화씨 온도
```



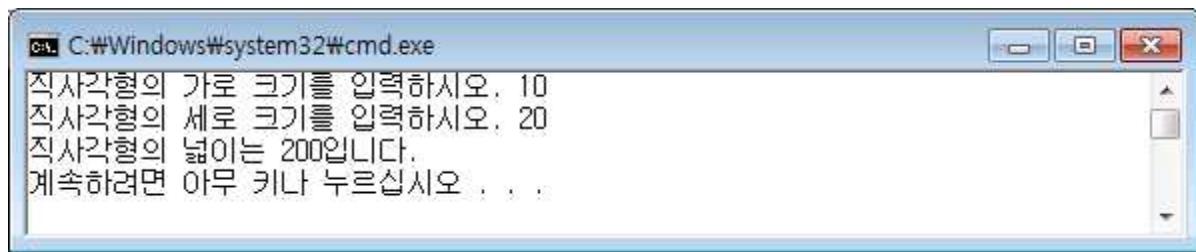
### [A03] 직사각형 넓이 계산

직사각형의 가로크기와 세로크기를 입력받아 이 값을 이용하여 직사각형의 넓이를 계산하고 출력하라.

단, 직사각형의 넓이 = 가로크기 \* 세로크기 로 계산한다.

변수는 다음과 같이 사용하라.

```
width      # 가로크기  
height     # 세로크기  
area       # 직사각형의 넓이
```



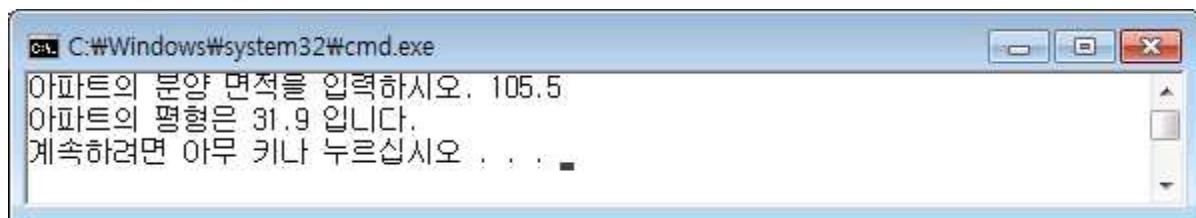
### [A04] 아파트 평형 계산

아파트의 분양 면적을 제곱미터( $m^2$ ) 단위로 입력받아 이 값을 평형 단위의 값으로 변환하여 출력하라.

단, 평형 수 = 제곱미터 / 3.305 로 계산한다.

변수는 다음과 같이 사용하라.

```
m2_area      # 면적 (제곱미터)  
pyung_area    # 면적 (평수)
```



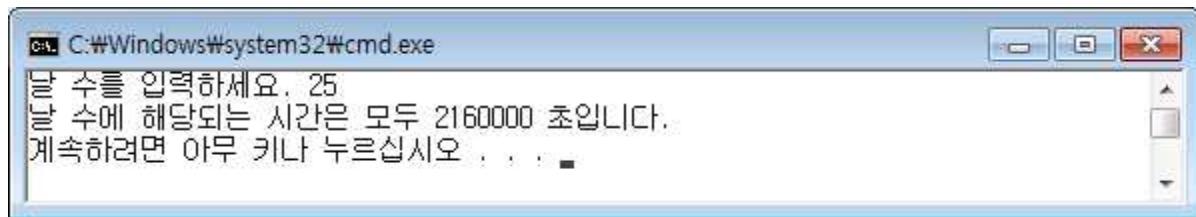
### [A05] 날짜 계산

날 수를 입력받아 이 날 수에 해당되는 기간은 모두 몇 초인지 계산하여 출력하라.

단, 초 = 날 수 \* 24 \* 60 \* 60 으로 계산한다.

변수는 다음과 같이 사용하라.

```
days          # 날 수  
seconds      # 초 단위 시간
```



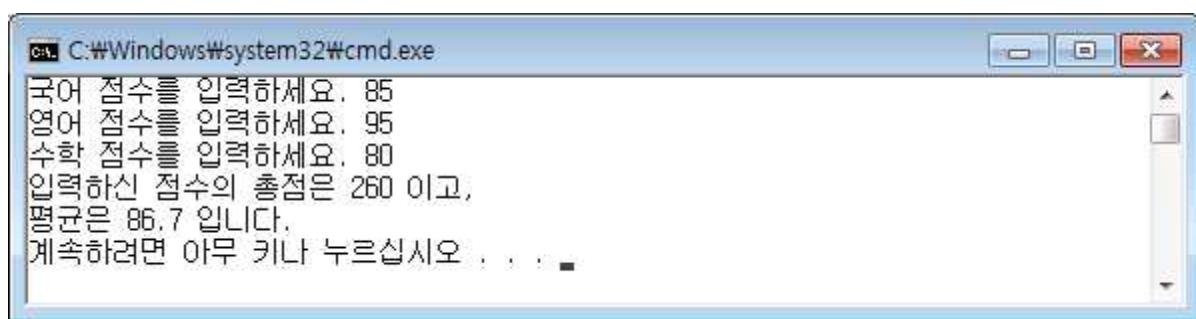
### [A06] 점수 계산

국어, 영어, 수학 점수를 입력받아 이 점수의 총점과 평균을 계산하여 출력하라.

단, 총점 = 국어점수 + 영어점수 + 수학점수, 평균 = 총점 / 3.0 으로 계산하라.

변수는 다음과 같이 사용하라.

```
kor          # 국어점수  
eng          # 영어점수  
math         # 수학점수  
total        # 총점  
average      # 평균점수
```



### [A07] 파일 용량 계산

파일 용량을 기가바이트 단위로 입력받아 이 값을 메가바이트, 킬로바이트, 바이트 단위로 계산하여 각각 출력하라.

단, 계산방법은 다음과 같다.

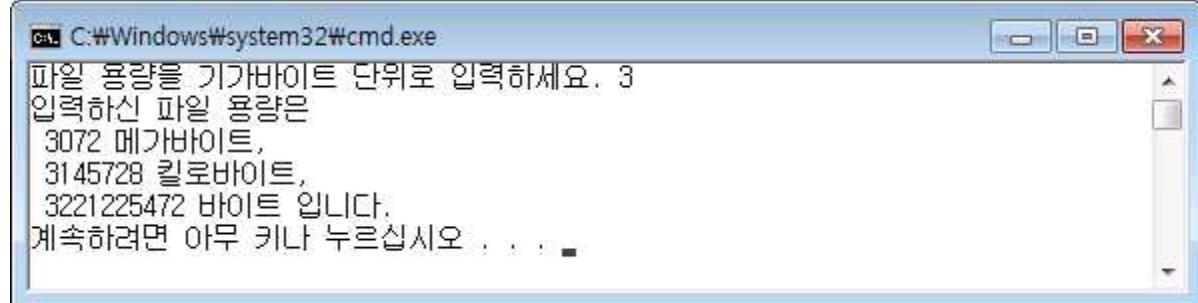
메가바이트 수 = 기가바이트 수 \* 1024

킬로바이트 수 = 메가바이트 수 \* 1024

바이트 수 = 킬로바이트 수 \* 1024

변수는 다음과 같이 사용하라.

gigabytes	# 용량(기가바이트 단위)
megabytes	# 용량(메가바이트 단위)
kilobytes	# 용량(킬로바이트 단위)
bytes	# 용량(바이트 단위)



## [Step B] 단순 조건문 사용하기

프로그램에서는 입력된 값이나 계산한 값에 대해서 특정 조건에 부합하는지의 여부를 판정하여 그 결과에 따라 다른 작업을 수행하도록 해야 할 필요가 있다.这时候에 사용하는 구문이 조건문인데, 조건문의 가장 기본적인 모습은 다음과 같다. 여기서 주의해야 할 것은 반드시 조건문이 참인 경우에 수행할 문장들은 문장 앞에서 들여쓰기를 해 주어야 한다.

```
if 조건문 :  
    조건문이 참인 경우에 수행해야 하는 구문들
```

예를 들어 입력받은 사과의 개수가 30개가 넘는 경우에 “한 박스에 담을 수 없습니다.”라고 출력하려고 하면 다음과 같이 하면 된다.

```
unit_price = int(input("사과 1개의 가격은 얼마입니까? "))  
apple_count = int(input("사과의 개수는 모두 몇 개 입니까? "))  
if apple_count > 30 :  
    print("한 박스에 담을 수 없습니다.")
```

조건문이 참인 경우에 수행해야 하는 내용과 그 외의 경우 즉, 조건문이 거짓인 경우에 수행해야 하는 내용이 따로 있는 경우에는 다음과 같은 형식의 구문을 사용한다.

```
if 조건문 :  
    조건문이 참인 경우에 수행해야 하는 구문들  
else :  
    조건문이 거짓인 경우에 수행해야 하는 구문들
```

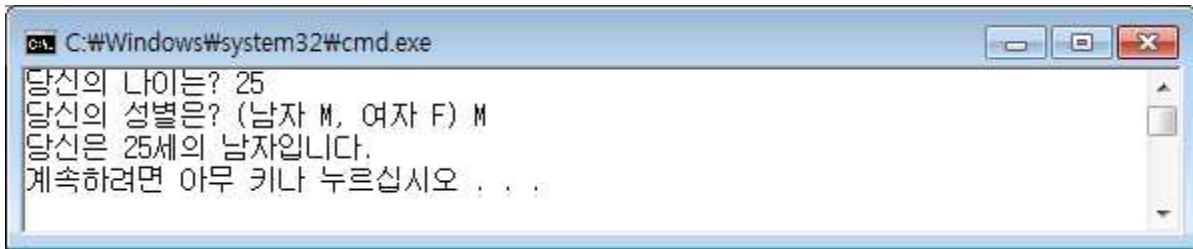
위의 예제에서 사과의 개수가 30개가 넘지 않는 경우에는 “한 박스에 담을 수 있습니다.”라고 출력하도록 변경하면 다음과 같이 하면 된다.

```
unit_price = int(input("사과 1개의 가격은 얼마입니까? "))  
apple_count = int(input("사과의 개수는 모두 몇 개 입니까? "))  
if apple_count > 30 :  
    print("한 박스에 담을 수 없습니다.")  
else :  
    print("한 박스에 담을 수 있습니다.")
```

이번에는 나이와 성별을 입력받아 이를 출력하는 구문을 만들어 보자. 나이는 정수형 변수로 처리하고, 성별은 'M' 또는 'F'의 값을 갖는 문자열 변수로 처리하도록 한다.

```
age = int(input("당신의 나이는? "))
gender = input("당신의 성별은? (남자 M, 여자 F) ")
if gender == 'M':
    print("당신은 ", age, "세의 남자입니다.")
else:
    print("당신은 ", age, "세의 여자입니다")
```

이 때 성별을 입력받을 경우에 `input()`앞에 `int`를 넣지 않는 이유는 성별은 숫자가 아닌 문자열로 입력받기 때문이라는 것에 주의하라. 문자열은 따옴표로 표시되는데 작은따옴표나 큰따옴표 모두 사용할 수 있다.



그러면 조건문을 이용하는 간단한 프로그램을 만들어보자. 문제는 2개의 숫자를 입력받아 이 숫자들이 같은 수인지를 판정하는 것이다. 이 문제를 해결하기 위해 다음과 같은 순서를 생각하자.

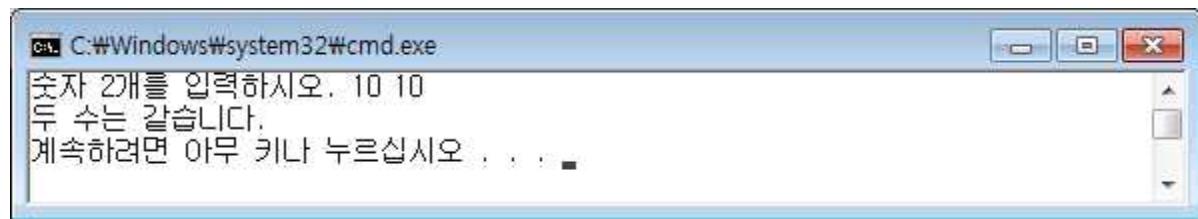
1. 숫자 2개의 값을 입력받는다.
2. 만일 첫 번째 숫자와 두 번째 숫자가 같으면 3번을, 그렇지 않으면 4번을 수행한다.
3. “두 수는 같습니다.”라고 출력한다.
4. “두 수는 같지 않습니다.”라고 출력한다.

이 문제를 해결하는 프로그램은 다음과 같다.

```
# -*- coding: utf-8 -*-
# 예제 ex_B2.py
num1, num2 = input("숫자 2개를 입력하시오. ").split() # 입력받은 문자열을 둘로 나눈다
num1 = int(num1) # 첫 문자열을 정수로 변환
num2 = int(num2) # 두번째 문자열을 정수로 변환
if num1 == num2 :
    print("두 수는 같습니다.")
else :
    print("두 수는 같지 않습니다.")
```

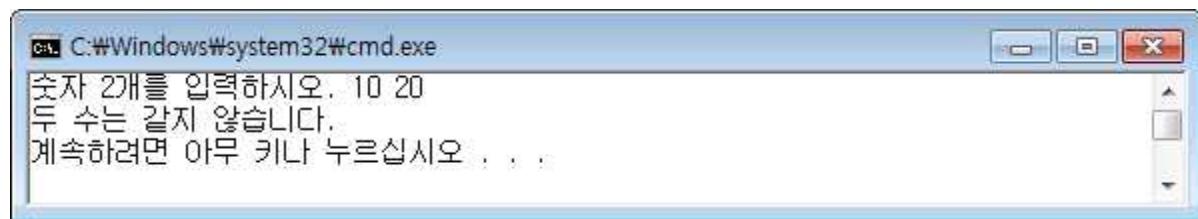
위 프로그램에서 `input()`을 통해 입력된 문자열은 2개의 숫자로 구성되어 있기 때문에 `.split()`를 붙여서 문자열을 빈칸을 기준으로 2개로 나눈 후에 각각 `num1`과 `num2`에 할당하게 된다. 그리고 이 2개의 변수에는 숫자가 아닌 문자열의 형태로 할당되었기 때문에 `int(num1)`을 해 주면 숫자로 바꾸어 다시 `num1`에 할당하게 되는 것이다.

이 프로그램을 수행한 결과는 다음과 같다. 같은 수를 넣은 경우와 다른 수를 넣은 경우에 따라 출력 내용이 달라지는 것을 확인할 수 있다.



C:\Windows\system32\cmd.exe

```
숫자 2개를 입력하시오. 10 10
두 수는 같습니다.
계속하려면 아무 키나 누르십시오 . . .
```



C:\Windows\system32\cmd.exe

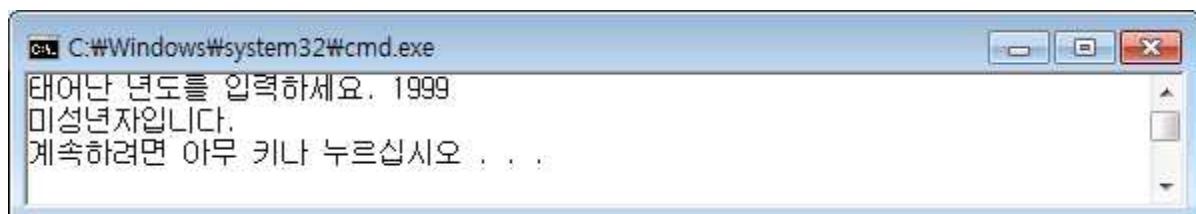
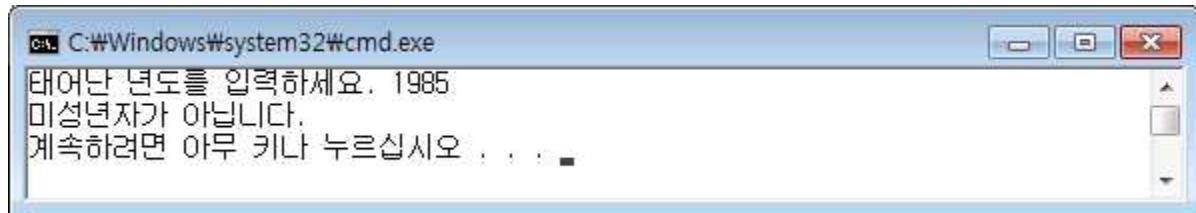
```
숫자 2개를 입력하시오. 10 20
두 수는 같지 않습니다.
계속하려면 아무 키나 누르십시오 . . .
```

○ 실습 문제

[B01] 나이 계산 및 미성년자 판정

태어난 년도를 입력받아 나이를 계산한 후, 미성년자인지 여부를 판정하여 그 결과를 출력하라.  
단, 나이 =  $2012 - \text{태어난 년도} + 1$ 로 계산하고 나이가 20세 미만인 경우, 미성년자로 판정한다.  
변수는 다음과 같이 사용하라.

```
birth_year # 태어난 년도  
age       # 나이
```



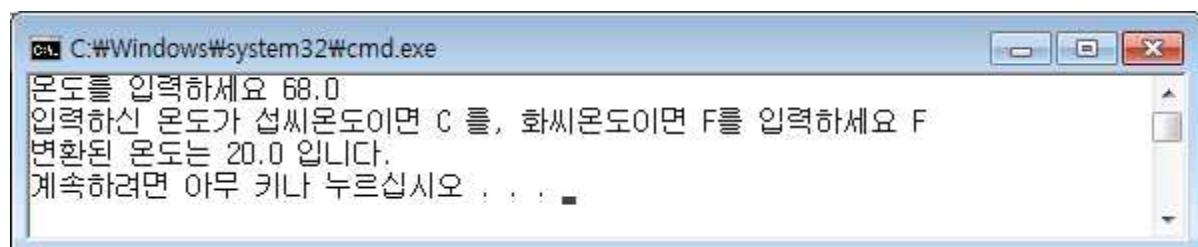
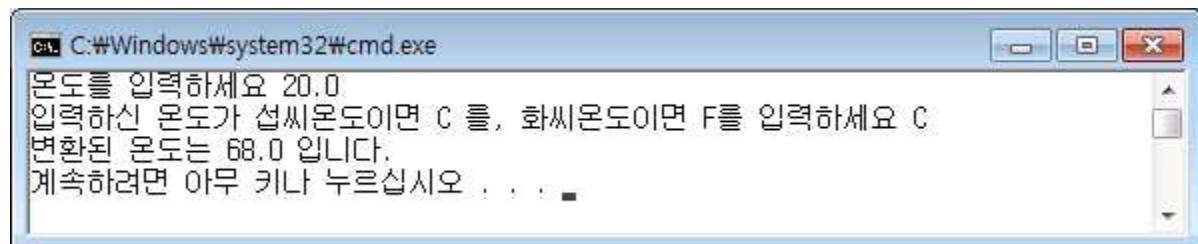
## [B02] 온도 상호 변환

온도를 입력받은 후, 이 값이 섭씨온도인지 화씨온도인지 종류를 C또는 F로 입력받아 섭씨온도이면 화씨온도로 변환하고, 화씨온도이면 섭씨온도로 변환하여 그 값을 출력하라.

단, 화씨온도 = 섭씨온도 \* 1.8 + 32, 섭씨온도 = (화씨온도 - 32)/1.8 로 계산한다.

변수는 다음과 같이 사용하라.

```
input_degree      # 입력받은 온도  
kind              # 온도의 종류, 섭씨온도이면 'C', 화씨온도이면 'F'  
output_degree    # 변환한 온도
```



**[B03] 직사각형 넓이 계산 및 정사각형 판정**

직사각형의 가로크기와 세로크기를 입력받아 이 값을 이용하여 직사각형의 넓이를 계산하고 정사각형인지의 여부를 판정하여 함께 출력하라.

단, 직사각형의 넓이 = 가로크기 \* 세로크기 로 계산한다.

변수는 다음과 같이 사용하라.

width, height # 가로크기, 세로크기

area # 사각형의 넓이

```
C:\Windows\system32\cmd.exe
직사각형의 가로 크기를 입력하시오. 10
직사각형의 세로 크기를 입력하시오. 20
직사각형의 넓이는 200 이고
정사각형이 아닙니다.
계속하려면 아무 키나 누르십시오 . . .
```

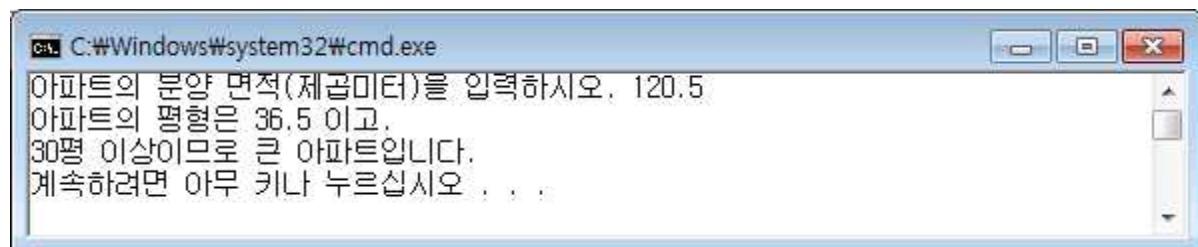
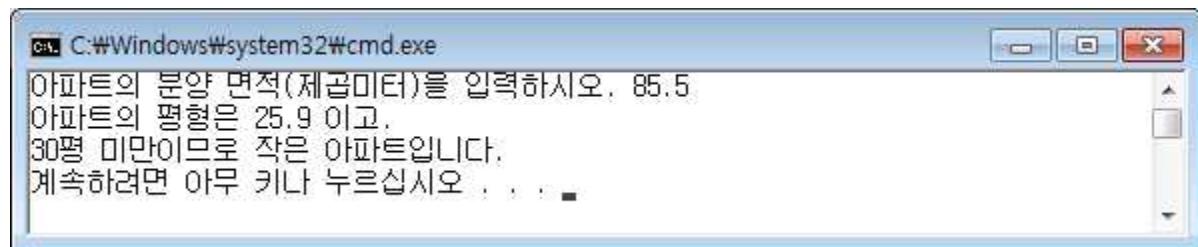
```
C:\Windows\system32\cmd.exe
직사각형의 가로 크기를 입력하시오. 20
직사각형의 세로 크기를 입력하시오. 20
직사각형의 넓이는 400 이고
정사각형입니다.
계속하려면 아무 키나 누르십시오 . . .
```

### [B04] 아파트 평형 계산 및 종류 판정

아파트의 분양 면적을 제곱미터( $m^2$ ) 단위로 입력받아 이 값을 평형 단위의 값으로 변환하라. 그리고 평형 수에 따라 아파트의 종류가 작은 아파트인지 큰 아파트인지 판정하여 판정 결과를 출력하라.  
단, 평형 수 = 제곱미터 / 3.305 로 계산하고, 30평 미만이면 작은 아파트 30평 이상이면 큰 아파트로 판정한다.

변수는 다음과 같이 사용하라.

```
m2_area      # 면적 (제곱미터)  
pyung_area   # 면적 (평수)
```



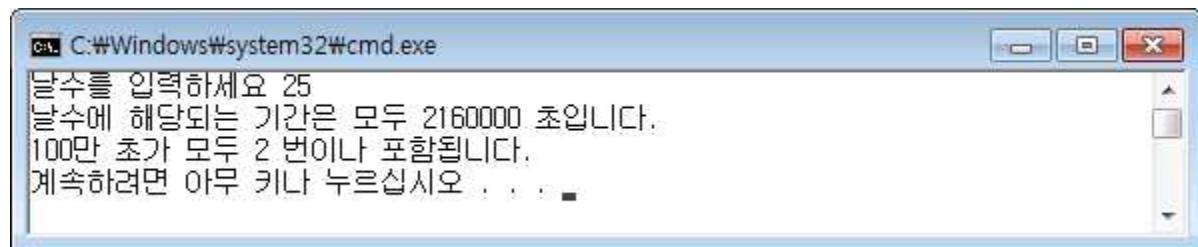
### [B05] 날짜 계산

날 수를 입력받아 이 날 수에 해당되는 기간은 모두 몇 초인지 계산하고, 100만 초가 넘는 경우에는 100만 초가 모두 몇 번이나 포함되는 지 계산하여 출력하라.

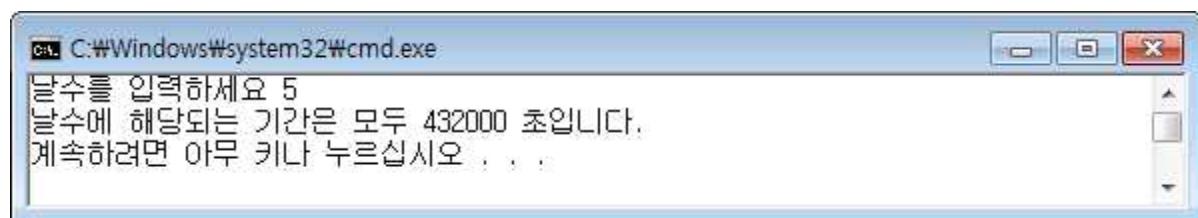
단, 초 = 날 수 \* 24 \* 60 \* 60 으로 계산한다.

변수는 다음과 같이 사용하라.

```
days          # 날 수  
seconds      # 초 단위 시간  
m_count      # 100만 초 포함 횟수
```



C:\Windows\system32\cmd.exe  
날수를 입력하세요 25  
날수에 해당되는 기간은 모두 2160000 초입니다.  
100만 초가 모두 2 번이나 포함됩니다.  
계속하려면 아무 키나 누르십시오 . . .



C:\Windows\system32\cmd.exe  
날수를 입력하세요 5  
날수에 해당되는 기간은 모두 432000 초입니다.  
계속하려면 아무 키나 누르십시오 . . .

### [B06] 점수 계산

국어, 영어, 수학 점수를 입력받아 이 점수의 총점과 평균을 계산하고, 각 과목별로 90점 이상이면 성적우수로 표시하여 출력하라.

단, 총점 = 국어점수 + 영어점수 + 수학점수, 평균 = 총점 / 3.0 으로 계산하라.

변수는 다음과 같이 사용하라.

```
kor, eng, math      # 국어점수, 영어점수, 수학점수  
total             # 총점  
average           # 평균점수
```

C:\Windows\system32\cmd.exe

```
국어 점수를 입력하세요 95  
영어 점수를 입력하세요 85  
수학 점수를 입력하세요 88  
입력하신 점수의 총점은 268이고,  
평균은 89.3입니다.  
국어점수가 우수합니다.  
계속하려면 아무 키나 누르십시오 . . .
```

C:\Windows\system32\cmd.exe

```
국어 점수를 입력하세요 95  
영어 점수를 입력하세요 99  
수학 점수를 입력하세요 94  
입력하신 점수의 총점은 288이고,  
평균은 96.0입니다.  
국어점수가 우수합니다.  
영어점수가 우수합니다.  
수학점수가 우수합니다.  
계속하려면 아무 키나 누르십시오 . . .
```

### [B07] 파일 전송 시간 계산

파일 용량을 메가바이트 단위로 입력받고, USB 포트가 2.0인지 아닌지를 'Y' 또는 'N'으로 입력받아 이에 따라 파일 전송 시간을 초 단위로 계산하여 출력하라.

단, 계산방법은 다음과 같다.

바이트 수 = 메가바이트 수 \* 1024 \* 1024

USB 1.1 전송 속도 = 1,500,000바이트 / 초

USB 2.0 전송 속도 = 60,000,000바이트 / 초

변수는 다음과 같이 사용하라.

`megabytes # 용량(메가바이트 단위)`

`bytes # 용량(바이트 단위)`

`usb2 # USB 2.0 사용여부 (Y: 예, N: 아니요)`

`time # 전송시간(초 단위)`

```
C:\Windows\system32\cmd.exe
파일 용량을 메가바이트 단위로 입력하세요 800
USB 포트가 2.0 이면 Y, 아니면 N을 입력하세요 Y
파일 전송 시간은 13 초입니다.
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\Windows\system32\cmd.exe
파일 용량을 메가바이트 단위로 입력하세요 800
USB 포트가 2.0 이면 Y, 아니면 N을 입력하세요 N
파일 전송 시간은 559 초입니다.
계속하려면 아무 키나 누르십시오 . . .
```

### [B08] 다양한 조건 판정

3개의 정수를 입력받아 이 숫자들에 대해서 다음 조건 중에 만족시키는 번호들을 모두 출력하라.

1번. 3개의 숫자 중 적어도 두 수의 값이 같다.

2번. 3개의 숫자 중 적어도 두 수의 크기가 모두 50 보다 크다.

3번. 3개의 숫자 중 어떤 두 수의 합이 나머지 하나의 숫자와 같다.

4번. 3개의 숫자 중 어떤 하나의 수로 다른 두 수를 나누면 나누어떨어지는 경우가 있다.

변수는 다음과 같이 사용하라.

```
num1, num2, num3      # 첫 번째 숫자, 두 번째 숫자, 세 번째 숫자
```

```
C:\Windows\system32\cmd.exe
첫번째 숫자를 입력하세요 10
두번째 숫자를 입력하세요 20
세번째 숫자를 입력하세요 30
3번 조건 만족 : 3개의 숫자 중 어떤 두 수의 합이 나머지 하나의 숫자와 같다.
4번 조건 만족 : 3개의 숫자 중 어떤 하나의 수로 다른 두 수를 나누면 나누어떨어지는 경우가 있다.
계속하려면 아무 키나 누르십시오 . . .
```

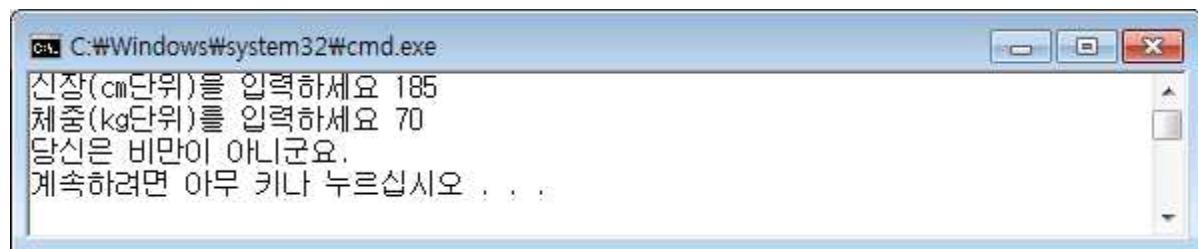
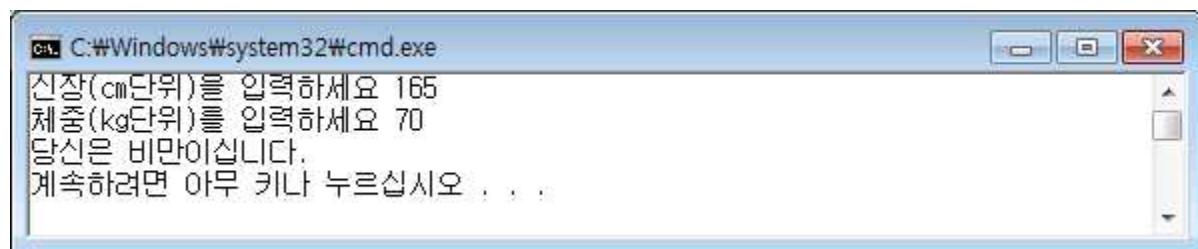
```
C:\Windows\system32\cmd.exe
첫번째 숫자를 입력하세요 10
두번째 숫자를 입력하세요 10
세번째 숫자를 입력하세요 20
1번 조건 만족 : 3개의 숫자 중 적어도 두 수의 값이 같다.
3번 조건 만족 : 3개의 숫자 중 어떤 두 수의 합이 나머지 하나의 숫자와 같다.
4번 조건 만족 : 3개의 숫자 중 어떤 하나의 수로 다른 두 수를 나누면 나누어떨어지는 경우가 있다.
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\Windows\system32\cmd.exe
첫번째 숫자를 입력하세요 50
두번째 숫자를 입력하세요 100
세번째 숫자를 입력하세요 200
2번 조건 만족 : 3개의 숫자 중 적어도 두 수의 크기가 모두 50 보다 크다.
4번 조건 만족 : 3개의 숫자 중 어떤 하나의 수로 다른 두 수를 나누면 나누어떨어지는 경우가 있다.
계속하려면 아무 키나 누르십시오 . . .
```

### [B09] 비만 판정

신장(cm단위)과 체중(kg단위)를 입력받은 후, 비만 여부를 판정하여 출력하라.  
 단, 비만여부는 다음 비만도 수치가 25이상인 경우에 "비만"으로 판단한다.  
 $\text{비만도 수치} = \text{체중(kg)} / (\text{신장(m)} \text{의 제곱})$  으로 계산한다. 이 때, 신장은 미터 단위로 환산해야 함을 유의하라.  
 변수는 다음과 같이 사용하라.

```
height, weight      # 신장(cm), 체중(kg)
bmi                 # 비만도 수치
```



## [Step C] 복합 조건문 사용하기

조건문을 사용하여 프로그램을 만드는 경우에 단순하게 조건문이 참이냐 거짓이냐만으로 구분할 수 없는 복잡한 경우가 있다. 즉 조건문이 연속으로 필요한 경우에는 다음과 같은 형식의 조건문을 사용해야 한다.

```
if 첫 번째 조건문 :  
    첫 번째 조건문이 참인 경우에 수행해야 하는 구문들  
elif 두 번째 조건문 :  
    첫 번째 조건문이 거짓이고 두 번째 조건문이 참인 경우에 수행해야 하는 구문들  
elif 세 번째 조건문 :  
    첫 번째 조건문과 두 번째 조건문 모두 거짓이고 세 번째 조건문이 참인 경우에 수행해야 하는 구문들  
...  
else :  
    위에서의 모든 조건문이 전부 거짓인 경우에 수행해야 하는 구문들
```

예를 들어 숫자를 하나 입력받은 후에 이 숫자가 양수인지, 0인지, 음수인지를 판별하는 경우를 생각해보자. 이 때 생각해 볼 수 있는 조건문은 (숫자가 양수이다) 와 (숫자가 음수이다) 와 (숫자가 0이다)의 세 종류가 필요하다.

```
if num1 > 0 :                      # 첫 번째 조건 : num1이 양수인가?  
    print("숫자는 양수입니다.")  
elif num1 < 0 :                      # 두 번째 조건 : num1이 양수는 아니고 음수인가?  
    print("숫자는 음수입니다.")  
else :                                # 마지막 조건 : num1이 양수도 음수도 아닌가?  
    print("숫자는 0입니다.")
```

그러면 이런 복합 조건문을 이용하는 간단한 프로그램을 만들어보자. 문제는 2개의 숫자를 입력받아 이 숫자 중에 어떤 숫자가 더 큰 수인지를 판정하는 것이다. 이 문제를 해결하기 위해 다음과 같은 순서를 생각하자.

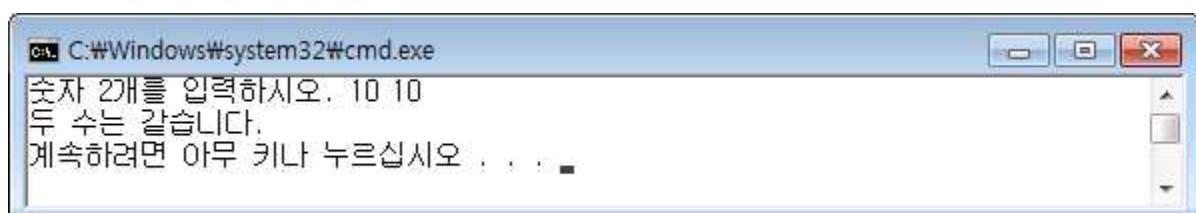
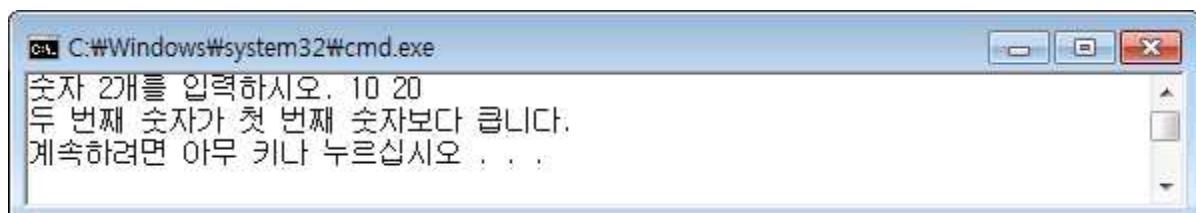
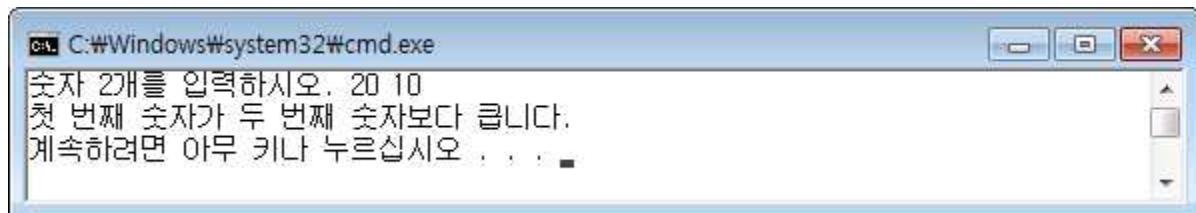
1. 숫자 2개의 값을 입력받는다.
2. 만일 첫 번째 숫자가 두 번째 숫자보다 크면 “첫 번째 숫자가 두 번째 숫자보다 큽니다.”라고 출력한다.
3. 그렇지 않고 만일 두 번째 숫자가 첫 번째 숫자보다 크면 “두 번째 숫자가 첫 번째 숫자보다 큽니다.”라고 출력한다.

4. 위 2번과 3번 조건 모두에 해당되지 않는 경우에는 “두 수는 같습니다.”라고 출력한다.

이 문제를 해결하는 프로그램은 다음과 같다.

```
# -*- coding: utf-8 -*-
# 예제 ex_C.py
num1, num2 = input("숫자 2개를 입력하시오. ").split()
num1 = int(num1)
num2 = int(num2)
if num1 > num2 :
    print("첫 번째 숫자가 두 번째 숫자보다 큽니다.")
elif num1 < num2 :
    print("두 번째 숫자가 첫 번째 숫자보다 큽니다.")
else :
    print("두 수는 같습니다.")
```

이 프로그램을 수행한 결과는 다음과 같다. 첫 번째 숫자가 더 큰 경우와 두 번째 숫자가 더 큰 경우, 그리고 두 숫자가 같은 경우에 따라 출력 내용이 달라지는 것을 확인할 수 있다.



## ○ 실습 문제

### [C01] 나이 계산 및 연령대 판정

태어난 년도를 입력받아 나이를 계산한 후, 나이에 따라 유아, 어린이, 청소년, 청년, 중년, 노년 여부를 판정하여 그 결과를 출력하라.

단, 나이 =  $2012 - \text{태어난 년도} + 1$ 로 계산하고 연령대 구분은 다음과 같이 판정한다.

7세 미만 : 유아,

7세 이상 ~ 13세미만 : 어린이,

13세 이상 ~ 20세 미만 : 청소년,

20세 이상 ~ 30세 미만 : 청년,

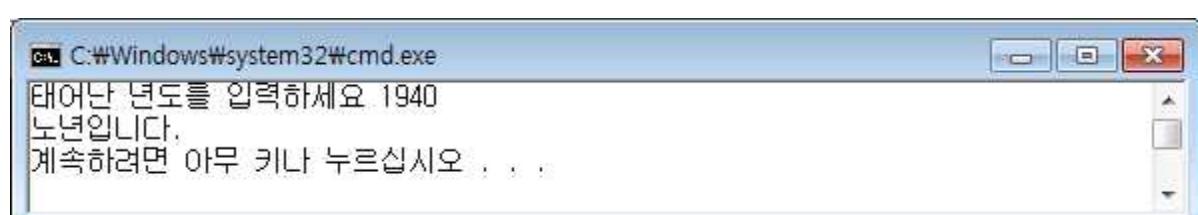
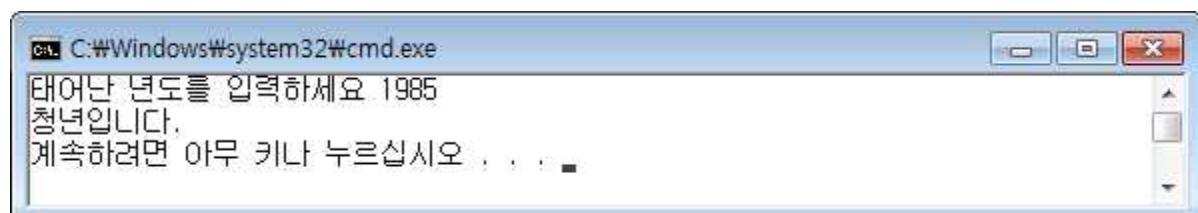
30세 이상 ~ 60세 미만 : 중년,

60세 이상 : 노년

변수는 다음과 같이 사용하라.

`birth_year # 태어난 년도`

`age # 나이`



### [C02] 물의 온도 구간 판정

물의 온도를 입력받은 후, 이 물이 어느 정도의 온수인지 판정하여 그 결과를 출력하라.

단, 온수의 판정 구간은 다음과 같이 판정한다.

음수값 (0미만) : 잘못입력

0도 이상 ~ 25도 미만 : 냉수

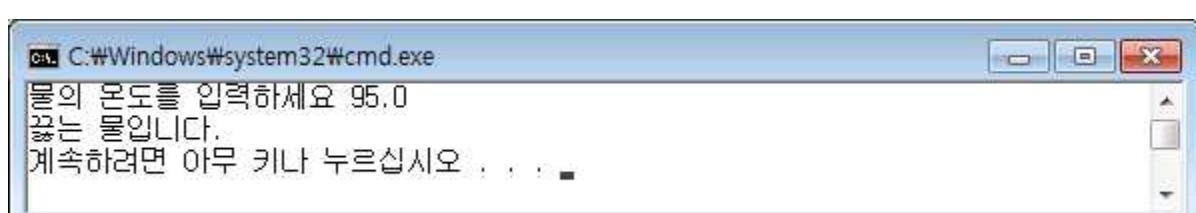
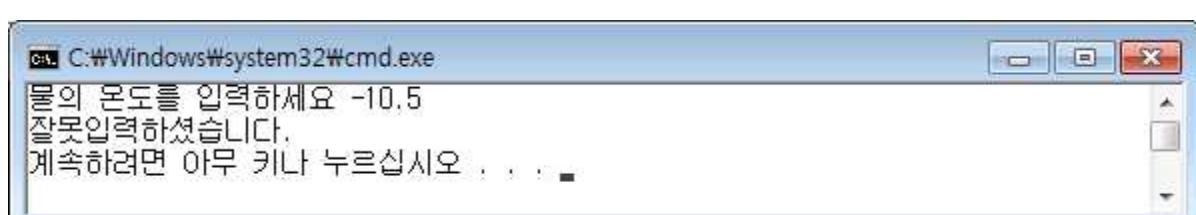
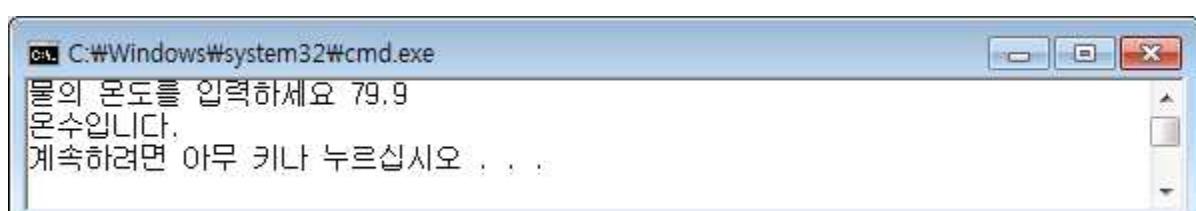
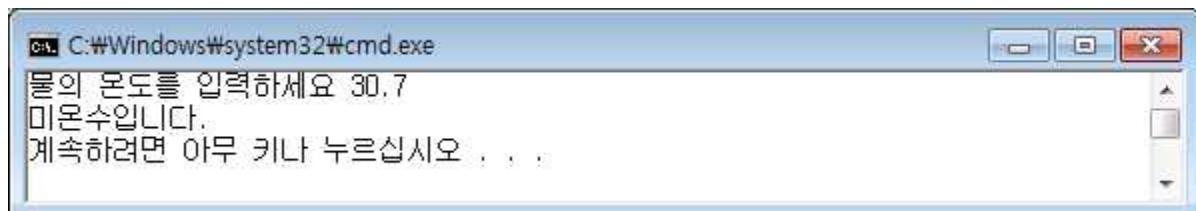
25도 이상 ~ 40도 미만 : 미온수

40도 이상 ~ 80도 미만 : 온수

80도 이상 : 끓는 물

변수는 다음과 같이 사용하라.

```
input_degree          # 입력받은 온도
```



### [C03] 직사각형 형태 판정

직사각형의 가로크기와 세로크기를 입력받아 이 값을 이용하여 직사각형의 모양에 대해 평가하는 내용을 출력하라.

단, 평가 내용은 다음 중 1가지 경우로 결정한다.

가로 크기와 세로크기가 동일 : "정사각형입니다."

가로 크기가 세로크기의 2배 이상 : "좌우로 길쭉한 직사각형입니다."

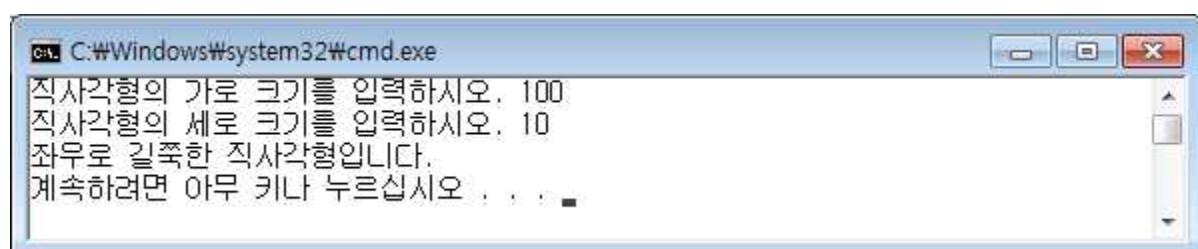
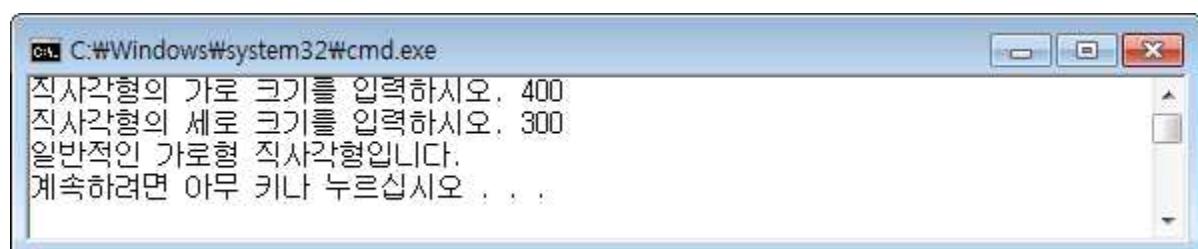
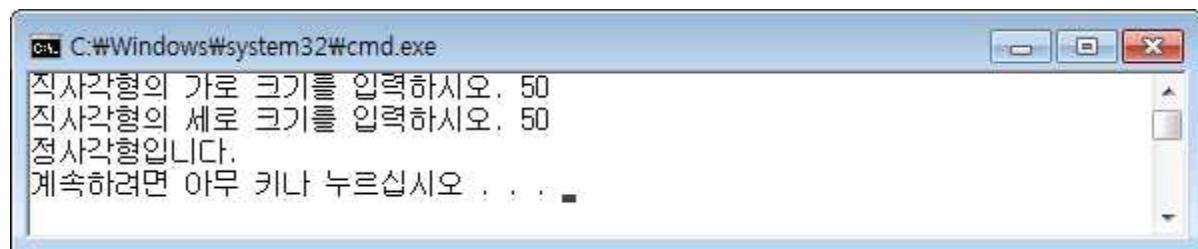
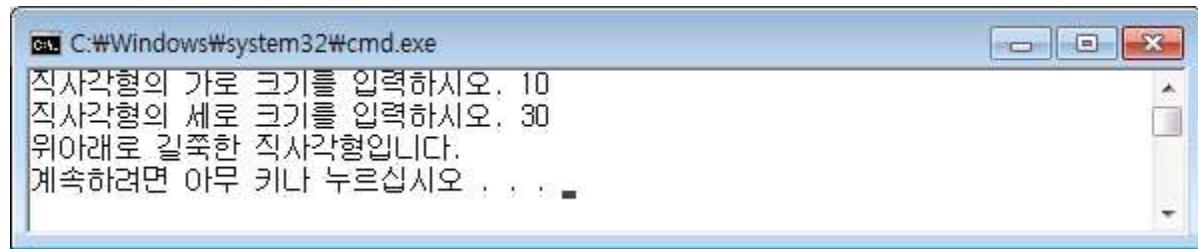
세로 크기가 가로크기의 2배 이상 : "위아래로 길쭉한 직사각형입니다."

가로 크기가 세로크기보다 크면 : "일반적인 가로형 직사각형입니다"

세로 크기가 가로크기보다 크면 : "일반적인 세로형 직사각형입니다"

변수는 다음과 같이 사용하라.

`width, height # 가로크기, 세로크기`



### [C04] 아파트 평형 계산 및 종류 판정

아파트의 분양 면적을 제곱미터( $m^2$ ) 단위로 입력받아 이 값을 평형 단위의 값으로 변환하라. 그리고 평형 수에 따라 아파트의 종류를 구분하여 그 결과를 출력하라.

단, 평형 수 = 제곱미터 / 3.305 로 계산하고, 크기에 따른 아파트 종류는 다음과 같이 판정한다.

15평 미만 : 소형

15평 이상 ~ 30평 미만 : 중소형

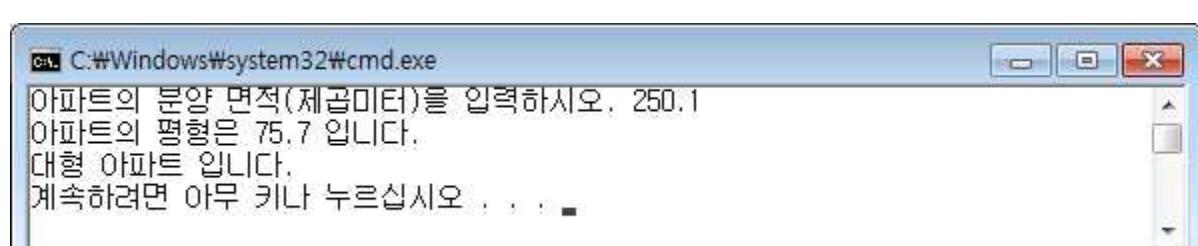
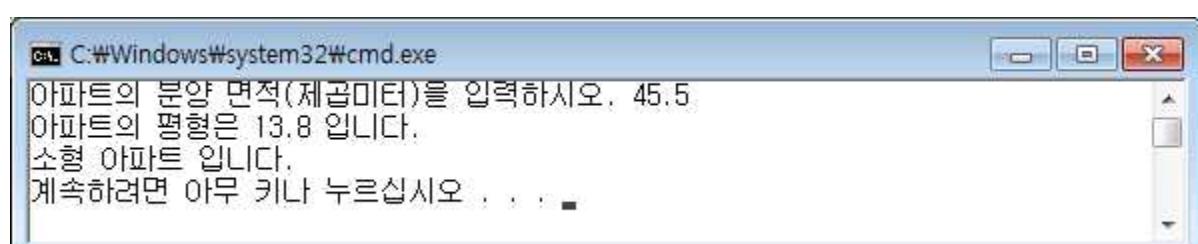
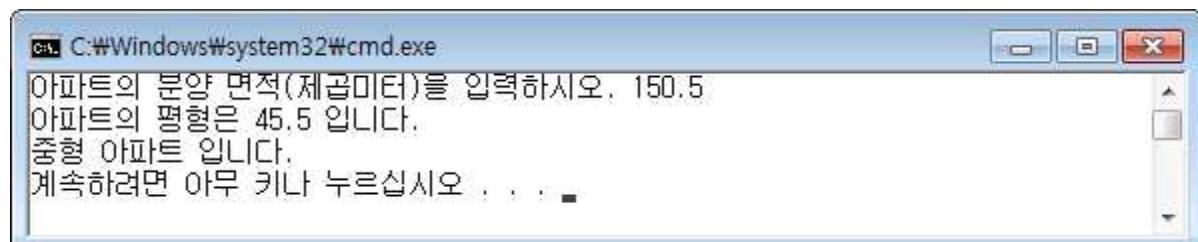
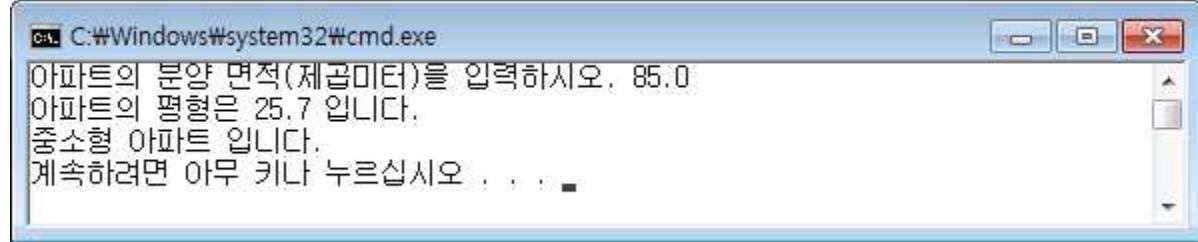
30평 이상 ~ 50평 미만 : 중형

50평 이상 : 대형

변수는 다음과 같이 사용하라.

`m2_area # 면적 (제곱미터)`

`pyung_area # 면적 (평수)`



### [C05] 연중 날짜 계산

날짜를 월과 일로 입력받아 이 날짜는 1년 중 몇 번째 날에 해당되는지 계산하여 출력하라.

단, 매 월의 날 수는 다음과 같이 정한다.

2월 : 28일

1, 3, 5, 7, 8, 10, 12월 : 31일

4, 6, 9, 11월 : 30일

변수는 다음과 같이 사용하라.

month, day # 월, 일

day\_count # 1년 중 날 수

```
C:\Windows\system32\cmd.exe
월을 입력하시오. 12
일을 입력하시오. 25
이 날짜는 1년 중 359 번째 날에 해당됩니다.
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\Windows\system32\cmd.exe
월을 입력하시오. 7
일을 입력하시오. 17
이 날짜는 1년 중 198 번째 날에 해당됩니다.
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\Windows\system32\cmd.exe
월을 입력하시오. 13
일을 입력하시오. 50
잘못 입력하셨습니다.
계속하려면 아무 키나 누르십시오 . . .
```

### [C06] 점수 계산

국어, 영어, 수학 점수를 입력받아 이 점수의 총점과 평균을 계산하고, 평균에 따라 등급을 정하여 출력하라.  
단, 총점 = 국어점수 + 영어점수 + 수학점수, 평균 = 총점 / 3.0 으로 계산하고 등급은 다음과 같은 기준으로 결정하라.

평균 90이상 : 수

평균 80이상 ~ 90미만 : 우

평균 70이상 ~ 80미만 : 미

평균 60이상 ~ 70미만 : 양

평균 60미만 : 가

변수는 다음과 같이 사용하라.

```
kor, eng, math          # 국어점수, 영어점수, 수학점수
```

```
total                  # 총점
```

```
average                # 평균점수
```

## [C07] 파일 전송 시간 계산

파일 용량을 메가바이트 단위로 입력받고, 전송 방식을 숫자로 입력받아 이에 따라 파일 전송 시간을 초 단위(소수점 1자리)로 계산하여 출력하라.

단, 계산방법은 다음과 같다.

바이트 수 = 메가바이트 수 \* 1024 \* 1024

Wi-Fi 전송 속도 = 1,500,000바이트 / 초

Bluetooth 전송 속도 = 300,000바이트 / 초

LTE 전송 속도 = 1,000,000바이트 / 초

USB 전송 속도 = 60,000,000바이트 / 초

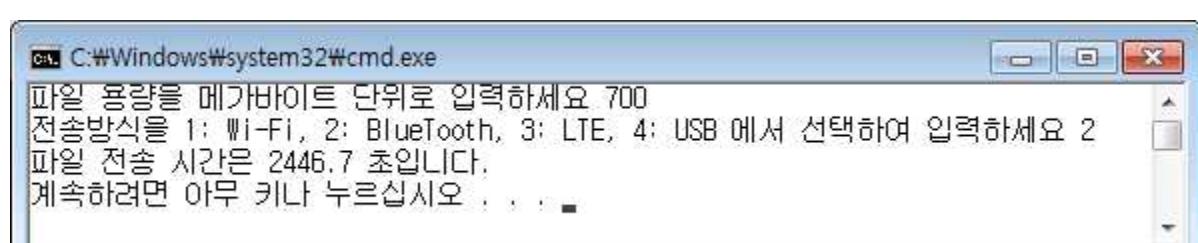
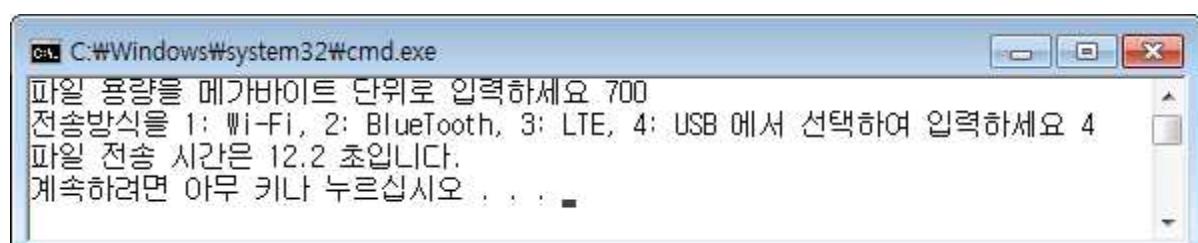
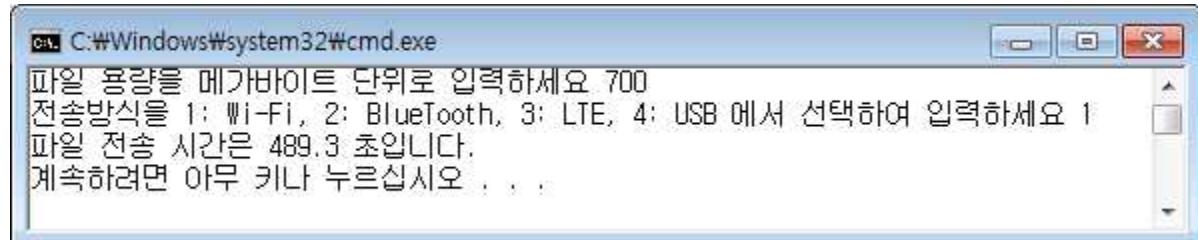
변수는 다음과 같이 사용하라.

megabytes # 용량(메가바이트 단위)

bytes # 용량(바이트 단위)

kind # 전송방식 (1: Wi-Fi, 2: BlueTooth, 3: LTE, 4: USB)

time # 전송시간(초 단위)

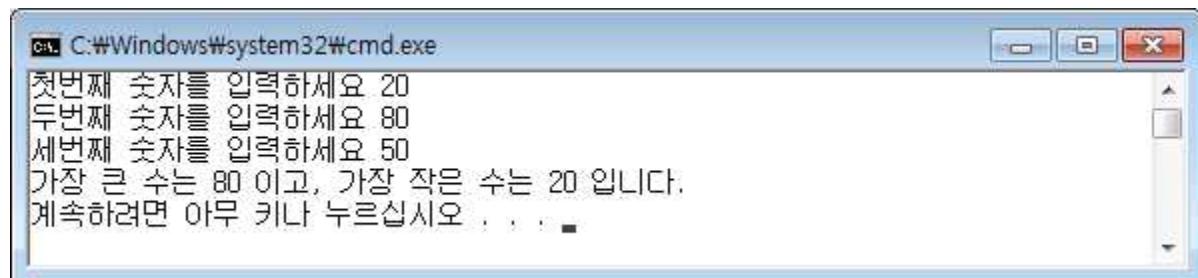


**[C08] 3개의 수 중 최댓값과 최솟값 구하기**

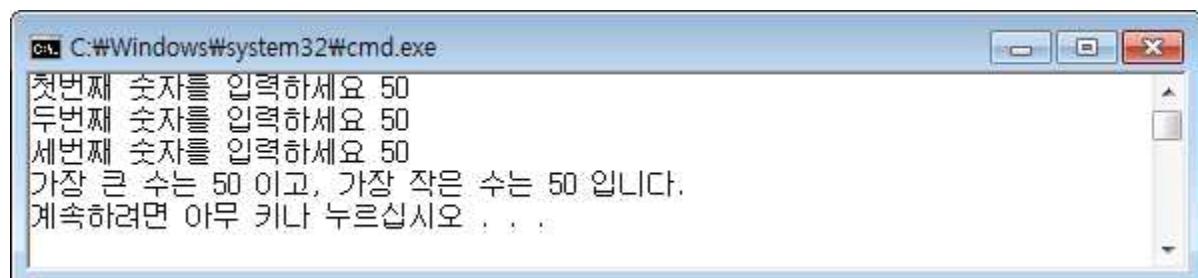
숫자를 3개 입력받은 후, 이 숫자 중에서 가장 큰 수, 가장 작은 수를 출력하라.

변수는 다음과 같이 사용하라.

```
num1, num2, num3      # 첫 번째 숫자, 두 번째 숫자, 세 번째 숫자  
max_num, min_num      # 가장 큰 숫자, 가장 작은 숫자
```



```
C:\Windows\system32\cmd.exe
첫번째 숫자를 입력하세요 20
두번째 숫자를 입력하세요 80
세번째 숫자를 입력하세요 50
가장 큰 수는 80이고, 가장 작은 수는 20입니다.
계속하려면 아무 키나 누르십시오 . . .
```



```
C:\Windows\system32\cmd.exe
첫번째 숫자를 입력하세요 50
두번째 숫자를 입력하세요 50
세번째 숫자를 입력하세요 50
가장 큰 수는 50이고, 가장 작은 수는 50입니다.
계속하려면 아무 키나 누르십시오 . . .
```

### [C09] 소득세 계산

연봉(원 단위)을 숫자로 입력받은 후, 연봉 금액에 대한 소득세를 계산하여 출력하라.

단, 소득세의 금액은 다음과 같이 계산한다.

연봉 1천만 원 미만 : 연봉의 9.5%

연봉 1천만 원 이상 ~ 4천만원미만 : 연봉의 19%

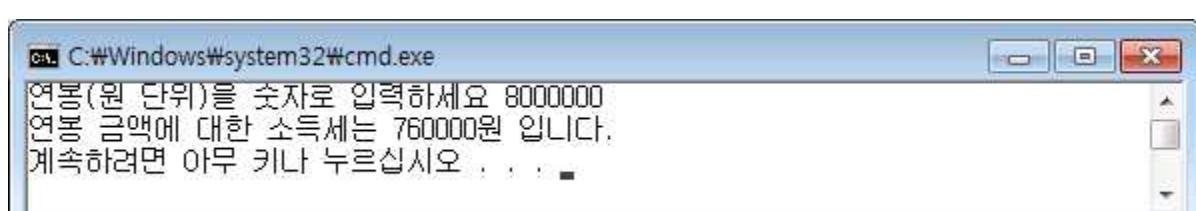
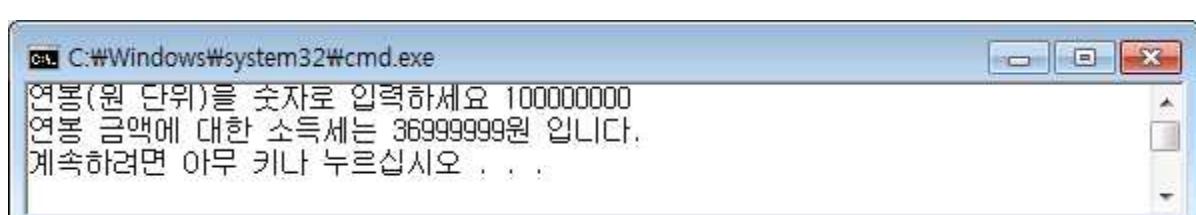
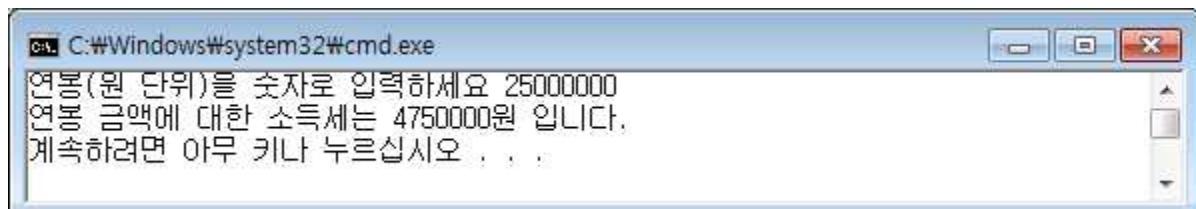
연봉 4천만 원 이상 ~ 8천만원미만 : 연봉의 28%

연봉 8천만 원 이상 : 연봉의 37%

변수는 다음과 같이 사용하라.

```
income      # 연봉 (원 단위)
```

```
tax         # 소득세 (원 단위)
```

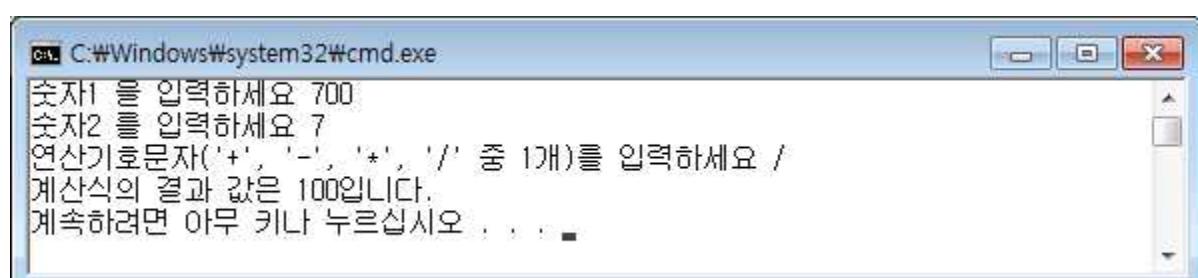
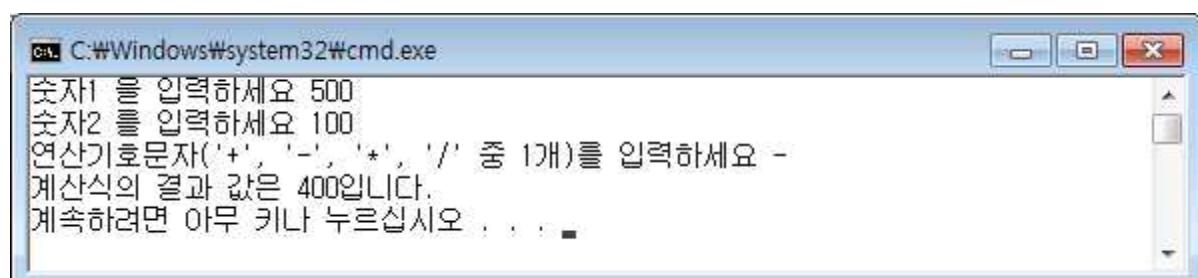
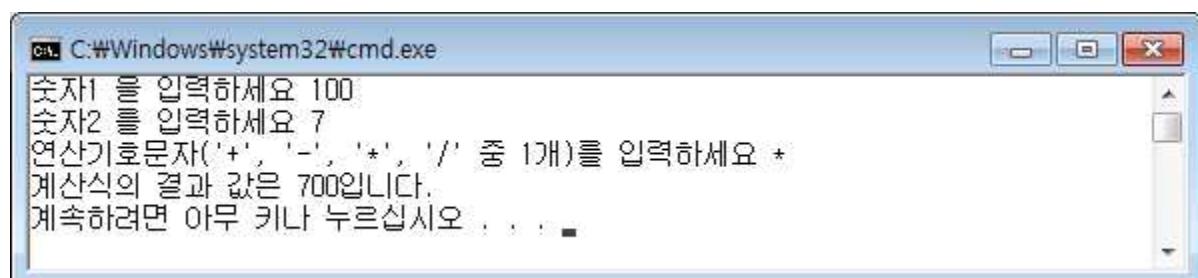
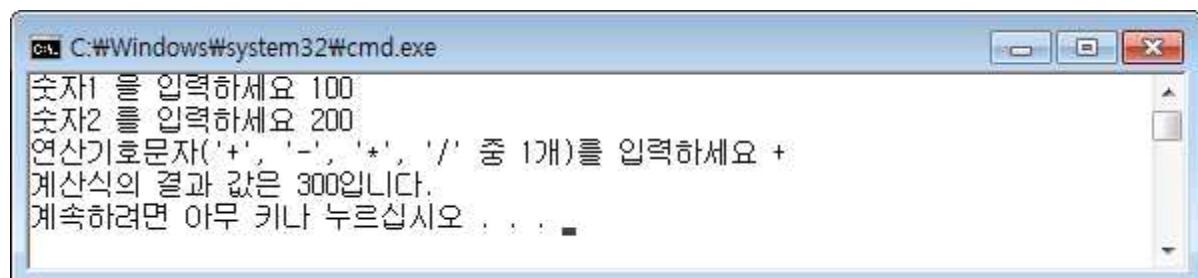


### [C10] 간단한 사칙연산 계산기

숫자 2개와 하나의 연산기호문자('+', '-', '\*', '/' 중 1개)를 입력받은 후, 첫 번째 숫자와 두 번째 숫자 사이에 연산기호를 넣은 계산식의 결과 값을 계산하여 출력하라.

변수는 다음과 같이 사용하라.

```
num1, num2      # 첫 번째 숫자, 두 번째 숫자
operator        # 연산기호문자('+', '-', '*', '/' 중 1개)
result          # 연산 결과
```



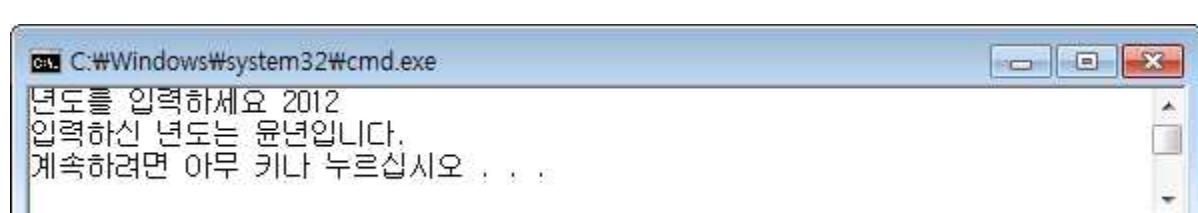
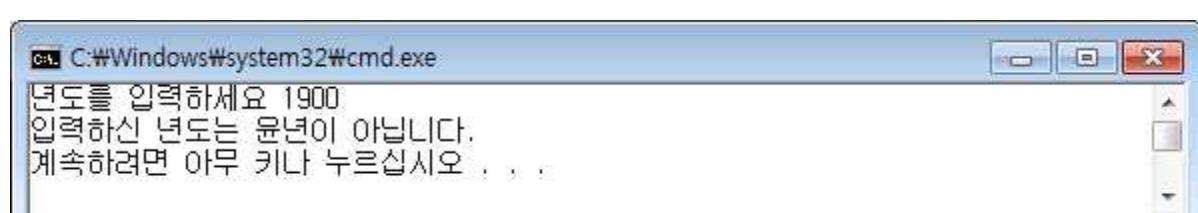
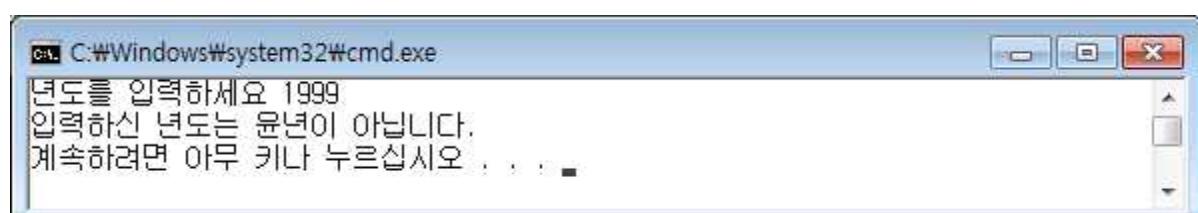
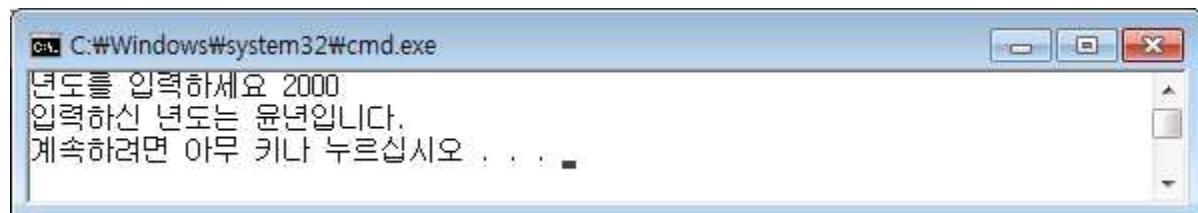
### [C11] 윤년 판정하기

년도를 입력받은 후, 이 년도가 윤년이지 아닌지를 판정하여 그 결과를 출력하라. 윤년의 판정 기준은 다음과 같다.

- 1) 년도가 4로 나누어떨어지는 경우에 윤년이다.
- 2) 위 1)의 기준 중에 100으로 나누어떨어지는 년도는 윤년에서 제외한다.
- 3) 위 2)의 기준 중에 400으로 나누어떨어지는 년도는 윤년이다.

변수는 다음과 같이 사용하라.

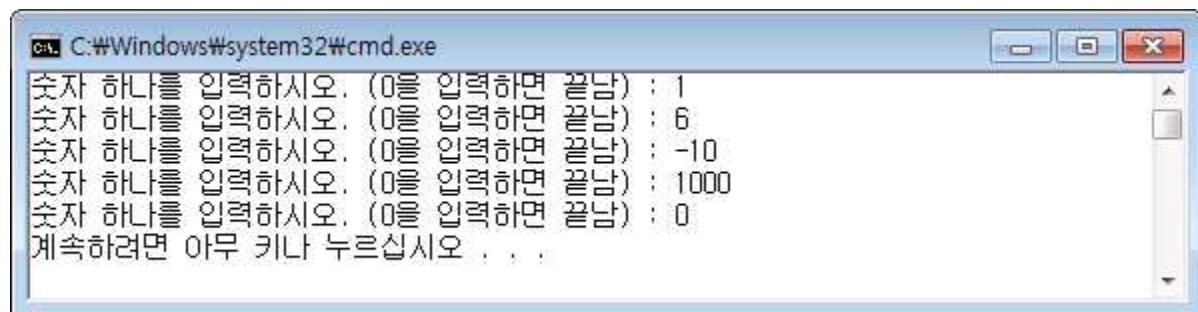
```
year          # 입력받은 년도
```



## [Step D] 반복문 사용하기

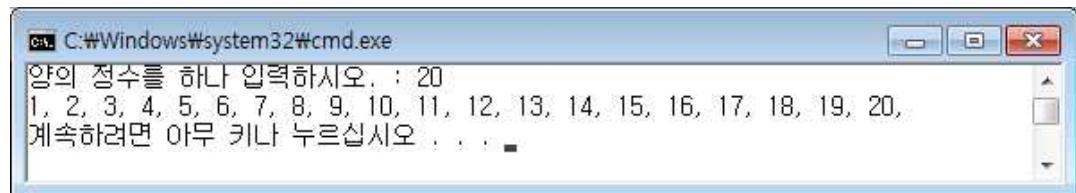
컴퓨터 프로그램에서는 동일하거나 비슷한 구문을 처리할 때에 반복문을 사용하면 효과적으로 프로그램을 수행할 수 있다. 가장 간단한 반복문은 while 1 : 구문으로 무한히 일련의 문장들을 반복시키는 경우에 사용한다. 다음의 예는 0이 입력되기 전까지 계속해서 숫자를 입력받는 프로그램 구문이다. 먼저 숫자를 입력받은 후에 입력된 숫자가 0인지에 따라 반복문의 중단 여부를 결정하게 된다. 반복문을 중단시켜야 하는 경우에는 조건문을 사용해서 break라는 명령을 사용하면 된다.

```
while 1 : # 1이라는 조건은 무조건 참인 조건이다.
    number = int(input("숫자 하나를 입력하시오. (0을 입력하면 끝남) : "))
    if number == 0 : break # number의 값이 0이면 반복을 끝낸다.
```



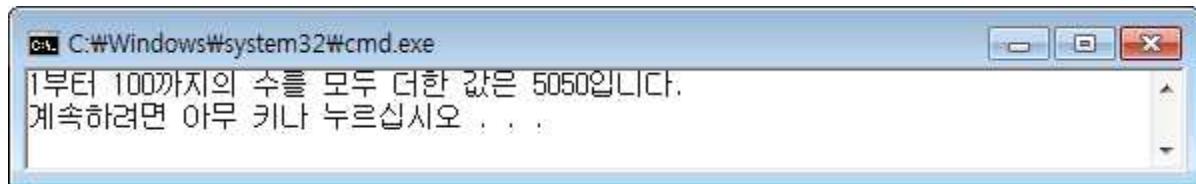
다음으로 while 에 특정 조건문을 사용하는 경우를 살펴보자. 예를 들어 양의 정수를 하나 입력받은 후에 1부터 입력받은 숫자까지의 모든 수를 화면에 출력시키는 문제를 해결하기 위해 while 구문을 사용해 보자.

```
number = int(input("양의 정수를 하나 입력하시오. : "))
count = 1
while (number >= count): # 1부터 시작하는 count값이 number를 넘지 않는 동안 반복
    print(count, end=", ") # count 출력 후 다음 줄로 내리지 않고 ", "를 출력함
    count = count + 1
print() # 빈 줄을 출력한다.
```



또 다른 반복문의 종류는 **for** 구문으로서 일반적으로 반복 횟수를 미리 알고 있는 경우 또는 특정 범위나 리스트, 튜플에 포함되는 요소만큼 반복해야 하는 경우에 사용한다. 예를 들어 1부터 100까지의 모든 수를 더한 값을 알아내려고 할 때 다음과 같이 프로그램 구문을 만들면 된다. 먼저 1부터 100까지의 숫자들을 모아놓은 리스트를 **range(1,101)**이라고 표현할 수 있다. 이 구문은 1부터 시작해서 101이 되기 전까지의 숫자들을 리스트로 만들라는 의미이다. 그래서 **for** 구문에서 이를 이용해서 반복문을 수행시킬 수 있다.

```
sum = 0
for i in range(1,101) :    # 변수 i의 값이 1부터 100까지 증가하는 동안 반복
    sum = sum + i
print("1부터 100까지의 수를 모두 더한 값은", sum, "입니다.")
```

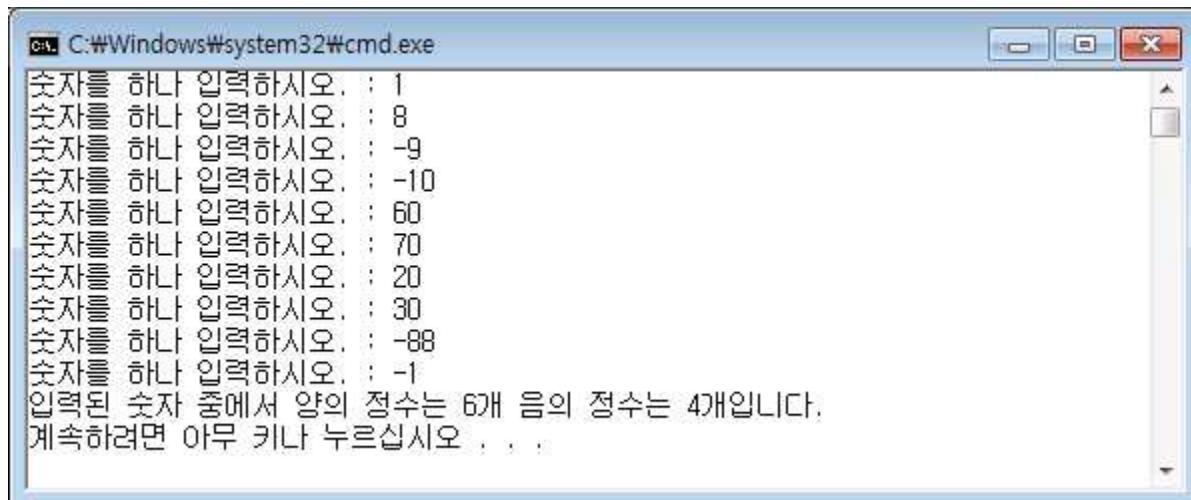


반복문과 조건문을 함께 사용해서 해결하는 문제를 생각해보자. 숫자 10개를 입력받은 후에 이 중에서 양의 정수는 몇 개이고, 음의 정수는 몇 개 인지 세는 문제를 풀어보자. 이 문제를 해결하기 위해서 다음과 같은 순서를 생각해 볼 수 있다.

1. 양의 정수 개수와 음의 정수 개수의 초기 값은 모두 0으로 정한다.
2. 숫자를 10번 입력 받으면서 3번 ~ 4번의 작업을 수행한다.
3. 만일 입력된 숫자가 양의 정수이면 양의 정수 개수를 하나 증가시킨다.
4. 그렇지 않고 만일 입력된 숫자가 음의 정수이면 음의 정수 개수를 하나 증가시킨다.
5. 반복문이 끝난 후에 양의 정수 개수와 음의 정수 개수를 출력한다.

위의 순서대로 프로그램을 제작하면 다음과 같다.

```
# -*- coding: utf-8 -*-
# 예제 ex_D.py
count_plus = 0                      # 양의 정수 개수 (초기값 0)
count_minus = 0                      # 음의 정수 개수 (초기값 0)
for i in range(10) :                 # 0부터 9까지 반복, 즉 10회 반복
    number = int(input("숫자를 하나 입력하시오. : "))
    if number > 0 :
        count_plus = count_plus + 1
    elif number < 0 :
        count_minus = count_minus + 1
print("입력된 숫자 중에서 양의 정수는", count_plus, "개 음의 정수는", count_minus, "개입니다.")
```



for 문에서는 인덱스로 사용하는 변수를 반복 구문 안에서 잘 활용하는 것이 중요하다. 예를 들어 구구단의 5단을 출력하는 프로그램을 만들어보자. 먼저 출력될 모습을 미리 확인해보면서 반복문에 적용할 인덱스의 규칙을 찾아내야 한다.

```
5 × 1 = 5
5 × 2 = 10
5 × 3 = 15
5 × 4 = 20
5 × 5 = 25
5 × 6 = 30
5 × 7 = 35
5 × 8 = 40
5 × 9 = 45
```

위의 출력 내용에서 매 줄에서 보이는 숫자는 3부분인데, 첫 번째 5는 모두 동일하고, 두 번째 수는 1부터 9까지 변하는 값이다. 세 번째 수는 첫 번째 수와 두 번째 수를 곱한 값이다. 그렇다면 다음 구문처럼 `for` 구문을 만들 수 있다.

```
for i in range(1,10) :  
    print(5, "x", i, "=", 5*i)
```

## ○ 실습 문제

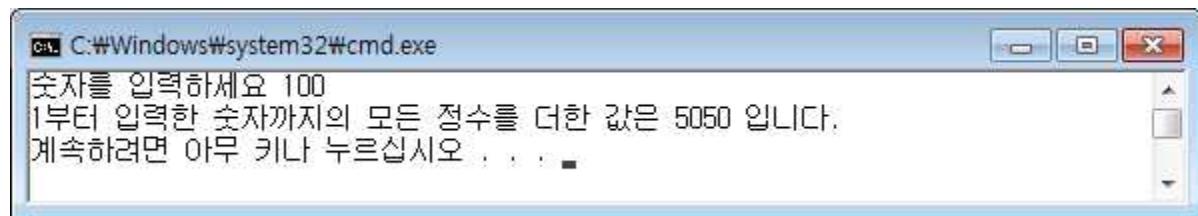
### [D01] 1부터 숫자 더하기

숫자를 입력받은 후, 이 숫자가 1보다 큰 경우 1부터 이 숫자까지의 모든 정수를 더한 값을 출력하라.

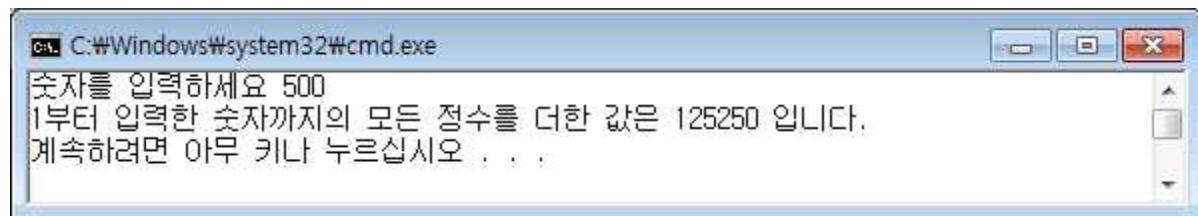
단, 입력한 숫자가 1 이하이면 "잘못 입력하였습니다."라고 출력한다.

반복은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
number      # 입력받은 수  
totalsum   # 1부터 더한 계산 결과 값  
i          # 반복문 사용을 위한 변수
```



```
C:\Windows\system32\cmd.exe
숫자를 입력하세요 100
1부터 입력한 숫자까지의 모든 정수를 더한 값은 5050 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

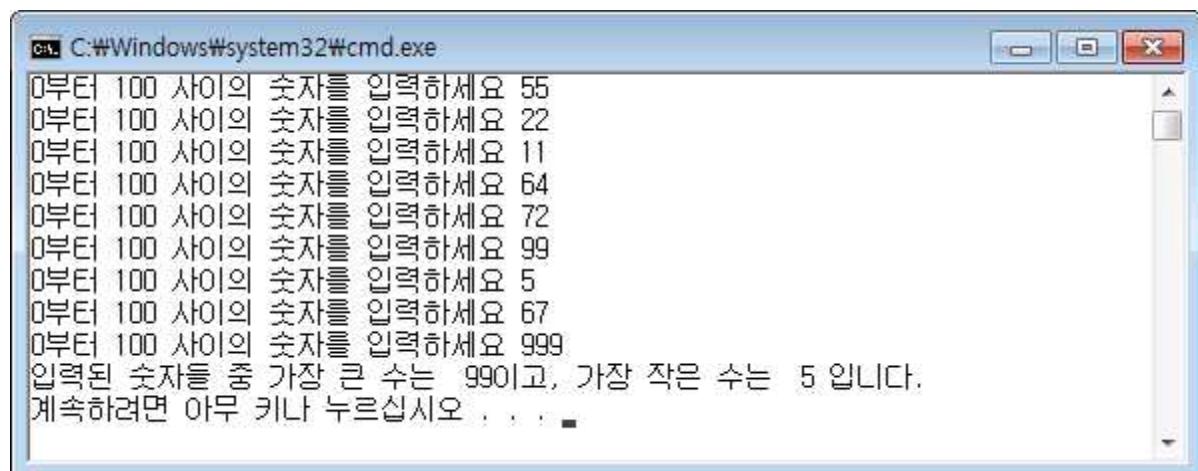


```
C:\Windows\system32\cmd.exe
숫자를 입력하세요 500
1부터 입력한 숫자까지의 모든 정수를 더한 값은 125250 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

### [D02] 입력 받은 숫자들 중에서 가장 큰 수와 가장 작은 수 구하기

반복해서 0부터 100 사이의 숫자를 입력받아서 지금까지 입력된 숫자들 중 가장 큰 수와 가장 작은 수가 무엇인지 출력하라. 0부터 100 사이의 숫자가 아닌 수가 입력되면 반복문이 중단되도록 하라.  
반복은 while 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
number          # 입력받은 수  
max_num, min_num # 가장 큰 숫자, 가장 작은 숫자
```



**[D03] 입력 받은 숫자들의 총합계와 평균 값 구하기**

반복해서 0부터 100 사이의 숫자를 입력받아서 지금까지 입력된 숫자들이 모두 몇 개이며, 이 숫자들의 총 합계와 평균 값을 계산하여 출력하라. 0부터 100 사이의 숫자가 아닌 수가 입력되면 반복문이 중단되도록 하라.

반복은 while 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
number      # 입력받은 수  
count=0    # 입력받은 숫자의 개수  
totalsum   # 총합계  
average    # 평균 값
```

The screenshot shows a command-line interface window titled 'cmd C:\Windows\system32\cmd.exe'. The window contains the following text output:

```
0부터 100 사이의 숫자를 입력하세요 10  
0부터 100 사이의 숫자를 입력하세요 50  
0부터 100 사이의 숫자를 입력하세요 64  
0부터 100 사이의 숫자를 입력하세요 2  
0부터 100 사이의 숫자를 입력하세요 13  
0부터 100 사이의 숫자를 입력하세요 97  
0부터 100 사이의 숫자를 입력하세요 88  
0부터 100 사이의 숫자를 입력하세요 12  
0부터 100 사이의 숫자를 입력하세요 35  
0부터 100 사이의 숫자를 입력하세요 50  
0부터 100 사이의 숫자를 입력하세요 999  
입력한 10 개의 숫자들의 총합계는 421이고, 평균 값은 42.1입니다.  
계속하려면 아무 키나 누르십시오 . . .
```

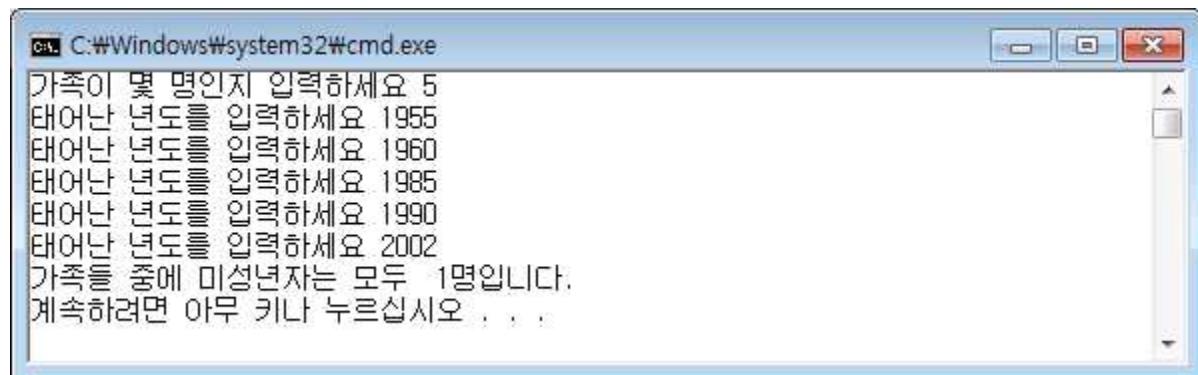
## [D04] 미성년자 숫자 세기

가족이 몇 명인지 입력받은 후, 그 인원 수 만큼 태어난 연도를 입력받으면서, 나이를 계산하여 미성년자가 모두 몇 명인지 출력하라.

단, 나이 =  $2012 - \text{태어난 연도} + 1$ 로 계산하고 나이가 20세 미만인 경우, 미성년자로 판정한다.

반복은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
count_all      # 가족 인원수  
count_youth    # 미성년자의 수  
birth_year     # 태어난 연도  
age            # 나이  
i              # 반복문을 위한 변수
```



### [D05] 직사각형 형태 개수 세기

반복해서 직사각형의 가로크기와 세로크기를 입력받으면서 이 값을 이용하여 직사각형의 모양을 판정하여 각각의 종류별로 몇 개가 입력되었는지 결과를 출력하라. 가로 크기와 세로 크기 중 하나라도 0이하의 값이 입력되면 반복을 중단한다.

단, 평가 기준은 다음과 같다.

가로 크기와 세로크기가 동일 : "정사각형"

가로 크기가 세로크기의 2배 이상 : "좌우로 길쭉한 직사각형"

세로 크기가 가로크기의 2배 이상 : "위아래로 길쭉한 직사각형"

가로 크기가 세로크기보다 크면 : "일반적인 가로형 직사각형"

세로 크기가 가로크기보다 크면 : "일반적인 세로형 직사각형"

반복은 while 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
width, height      # 가로크기, 세로크기
count1            # "정사각형"의 개수
count2            # "좌우로 길쭉한 직사각형"의 개수
count3            # "위아래로 길쭉한 직사각형"의 개수
count4            # "일반적인 가로형 직사각형"의 개수
count5            # "일반적인 세로형 직사각형"의 개수
```

The screenshot shows a Windows command prompt window titled 'cmd.exe' with the path 'C:\Windows\system32\cmd.exe'. The window displays the following text output:

```

C:\Windows\system32\cmd.exe
직사각형의 가로 크기와 세로 크기를 입력하시오. : 10 10
직사각형의 가로 크기와 세로 크기를 입력하시오. : 10 50
직사각형의 가로 크기와 세로 크기를 입력하시오. : 20 100
직사각형의 가로 크기와 세로 크기를 입력하시오. : 50 50
직사각형의 가로 크기와 세로 크기를 입력하시오. : 40 30
직사각형의 가로 크기와 세로 크기를 입력하시오. : 60 70
직사각형의 가로 크기와 세로 크기를 입력하시오. : 70 80
직사각형의 가로 크기와 세로 크기를 입력하시오. : 100 20
직사각형의 가로 크기와 세로 크기를 입력하시오. : 200 50
직사각형의 가로 크기와 세로 크기를 입력하시오. : 30 30
직사각형의 가로 크기와 세로 크기를 입력하시오. : -1 -1
"정사각형"의 개수는 3 입니다.
"좌우로 길쭉한 직사각형"의 개수는 2 입니다.
"위아래로 길쭉한 직사각형"의 개수는 2 입니다.
"일반적인 가로형 직사각형"의 개수는 1 입니다.
"일반적인 세로형 직사각형"의 개수는 2 입니다.
계속하려면 아무 키나 누르십시오 . . .

```

## [D06] 아파트 평형 계산 및 종류 판정

아파트 10채의 분양 면적을 제곱미터( $m^2$ ) 단위로 입력받아 이 값을 평형 단위의 값으로 변환하여 평형 수에 따라 아파트의 종류를 구분하여 종류별로 개수를 센 후, 그 결과를 출력하라.  
단, 평형 수 = 제곱미터 / 3.305 로 계산하고, 크기에 따른 아파트 종류는 다음과 같이 판정한다.

15평 미만 : 소형

15평 이상 ~ 30평 미만 : 중소형

30평 이상 ~ 50평 미만 : 중형

50평 이상 : 대형

반복문은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
m2_area          # 면적 (제곱미터)  
pyung_area       # 면적 (평수)  
count1           # 소형 아파트 개수  
count2           # 중소형 아파트 개수  
count3           # 중형 아파트 개수  
count4           # 대형 아파트 개수  
i                # 반복문을 위한 변수
```

The screenshot shows a Windows Command Prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the output of a Python script. The script prompts the user for the area of 10 apartments in square meters and calculates their equivalent size in pyungs. It then classifies each apartment into one of four categories: Small (less than 15 pyungs), Medium-Small (between 15 and 30 pyungs), Medium (between 30 and 50 pyungs), or Large (50 pyungs or more). The script also counts the number of apartments in each category. The output is as follows:

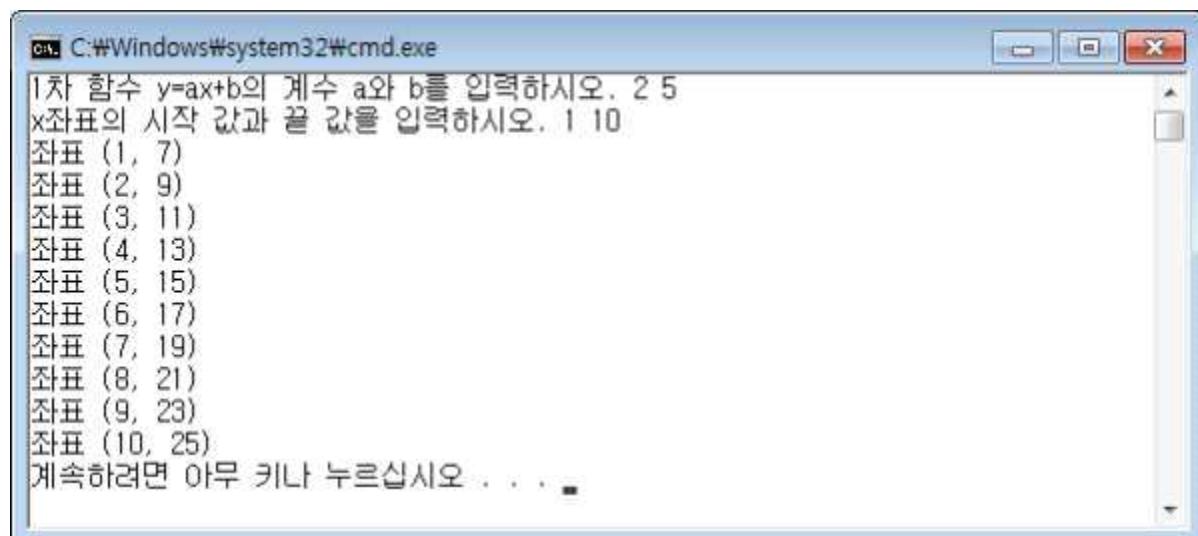
```
아파트의 분양 면적(제곱미터)을 입력하시오. 85.5  
→ 이 아파트의 평형은 25.9 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 120.4  
→ 이 아파트의 평형은 36.4 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 200.0  
→ 이 아파트의 평형은 60.5 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 40.5  
→ 이 아파트의 평형은 12.3 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 55  
→ 이 아파트의 평형은 16.6 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 155  
→ 이 아파트의 평형은 46.9 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 77.7  
→ 이 아파트의 평형은 23.5 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 99.9  
→ 이 아파트의 평형은 30.2 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 65.0  
→ 이 아파트의 평형은 19.7 입니다.  
아파트의 분양 면적(제곱미터)을 입력하시오. 250.1  
→ 이 아파트의 평형은 75.7 입니다.  
"소형 아파트"의 개수는 1 입니다.  
"중소형 아파트"의 개수는 4 입니다.  
"중형 아파트"의 개수는 3 입니다.  
"대형 아파트"의 개수는 2 입니다.  
계속하려면 아무 키나 누르십시오 . . .
```

**[D07] 1차 함수의 좌표 구하기**

1차 함수  $y=ax+b$ 에 대해 계수  $a$ 와  $b$ 를 입력받은 후,  $x$ 값의 시작 값과 마지막 값을 입력받아 이 두 수 사이의  $x$  값에 대한 1차 함수의  $(x, y)$  좌표들을 출력하라.

반복문은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
a, b          # 1차 함수의 계수 a, b  
x_begin, x_end    # x좌표의 시작 값과 끝 값  
x, y          # x좌표, y좌표
```

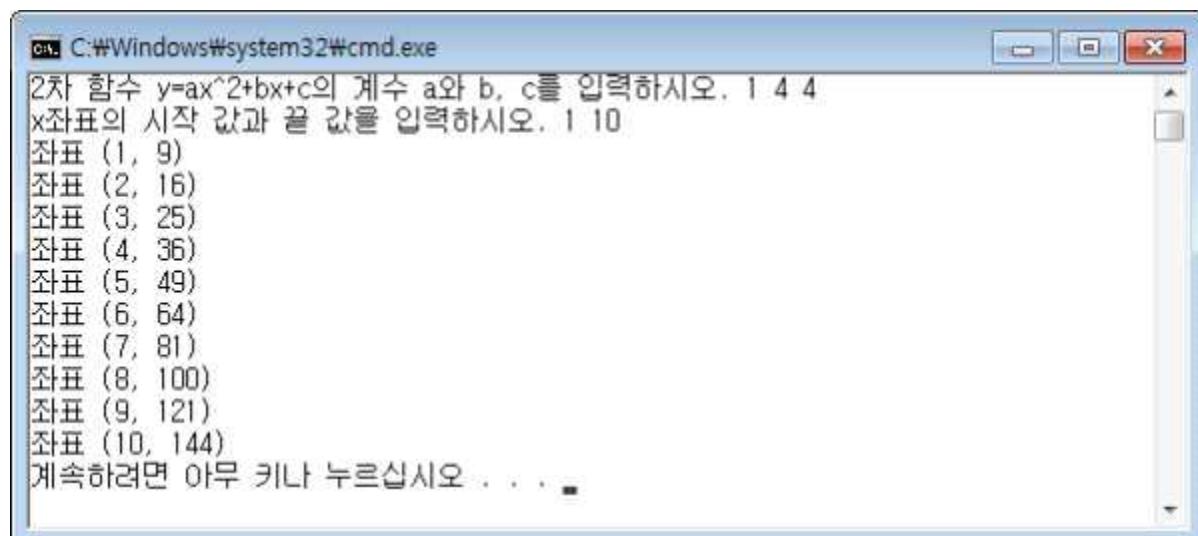


### [D08] 2차 함수의 좌표 구하기

2차 함수  $y=ax^2 + bx + c$ 에 대해 계수  $a$ 와  $b$ 와  $c$ 를 입력받은 후,  $x$ 값의 시작 값과 마지막 값을 입력받아 이 두 수 사이의  $x$ 값에 대한 2차 함수의  $(x, y)$  좌표들을 출력하라.

반복문은 `for` 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
a, b, c          # 2차 함수의 계수 a, b, c  
x_begin, x_end    # x좌표의 시작 값과 끝 값  
x, y            # x좌표, y좌표
```



**[D09] 원하는 구구단의 단 출력하기**

2부터 9 사이의 숫자를 입력받아 이 숫자에 해당하는 구구단을 출력하라. 단, 2부터 9 사이의 숫자가 아닌 수를 입력하면 "잘못 입력하였습니다."라고 출력하고 바르게 입력할 때까지 다시 입력을 받도록 하라.  
반복문은 적당하게 선택하고, 변수는 다음과 같이 사용하라.

```
dan          # 출력하려는 구구단의 단 수  
i           # 반복문을 위한 변수
```

The screenshot shows a Windows Command Prompt window titled 'cmd.exe' with the path 'C:\Windows\system32'. The window contains the following text:

```
출력하고 싶은 구구단의 단 수를 입력하시오(2~9). 1
잘못 입력하였습니다.
출력하고 싶은 구구단의 단 수를 입력하시오(2~9). 10
잘못 입력하였습니다.
출력하고 싶은 구구단의 단 수를 입력하시오(2~9). 7
7 × 1 = 7
7 × 2 = 14
7 × 3 = 21
7 × 4 = 28
7 × 5 = 35
7 × 6 = 42
7 × 7 = 49
7 × 8 = 56
7 × 9 = 63
계속하려면 아무 키나 누르십시오 . . .
```

### [D10] 두 수의 배타적 배수 출력하기

숫자 2개를 입력받은 후, 1부터 100까지의 숫자 중에 이 두 숫자 중 하나의 숫자에 대해서만 배수인 수를 모두 출력하라. 즉, 두 숫자의 공통 배수인 숫자는 출력하지 않아야 한다.

예를 들어 15와 20을 입력하게 되면 "15, 20, 30, 40, 45, 75, 80, 90, 100"이 출력된다.

반복문은 for 구문을 사용하고, 변수는 다음과 같이 사용하라.

```
num1, num2      # 입력받은 두 수  
i                # 반복문을 위한 변수
```

C:\Windows\system32\cmd.exe  
2개의 숫자를 입력하시오. 15 20  
15, 20, 30, 40, 45, 75, 80, 90, 100  
계속하려면 아무 키나 누르십시오 . . .

C:\Windows\system32\cmd.exe  
2개의 숫자를 입력하시오. 4 5  
4, 5, 8, 10, 12, 15, 16, 24, 25, 28, 30, 32, 35, 36, 44, 45, 48, 50, 52, 55, 56,  
64, 65, 68, 70, 72, 75, 76, 84, 85, 88, 90, 92, 95, 96  
계속하려면 아무 키나 누르십시오 . . .

## 영역 2 : 프로그래밍 제어구조 응용

영역 2에서는 영역 1에서 습득한 프로그래밍의 기본 제어구조를 응용하는 방법을 연습한다. 먼저 2단계 이상이 중복된 복합 반복문을 사용하는 방법과 뒷자리 형식인 리스트를 다루는 방법을 연습한다. 또한 그동안 연습한 조건문과 반복문이 다양하게 적용되는 응용 예제들을 해결하는 연습을 하게 된다. 마지막으로 영역 3으로 진행하기 위한 준비단계로서 파이썬에서 제공하는 주요 함수들을 사용하는 연습을 하게 된다. 영역 2는 다음과 같이 5개의 단계로 구성된다.

Step E : 복합 반복문 사용하기

Step F : 리스트 사용하기

Step G : 조건과 반복을 활용하는 응용 예제 해결하기

Step H : 파이썬의 주요 함수 사용하기

Step H' : 딕셔너리 사용하기

## [Step E] 복합 반복문 사용하기

이번 단계에서는 반복문을 2개 이상 겹쳐서 사용해야 하는 복합 반복문을 연습하려고 한다. 2개의 `for` 구문을 사용하는 이중 반복문의 사용법을 [Step D]에서 다룬 구구단 문제를 통해 알아보도록 하자. 구구단 중에서 5단을 출력하는 구문은 다음과 같았다.

```
for i in range(1,10) :  
    print(5, "x", i, "=", 5*i)
```

2단부터 9단까지의 구구단을 출력하는 첫번째 방법은 다음과 같이 위의 구문에서 5단을 출력하는 `for` 문장을 2단부터 9단까지 반복하는 것이다.

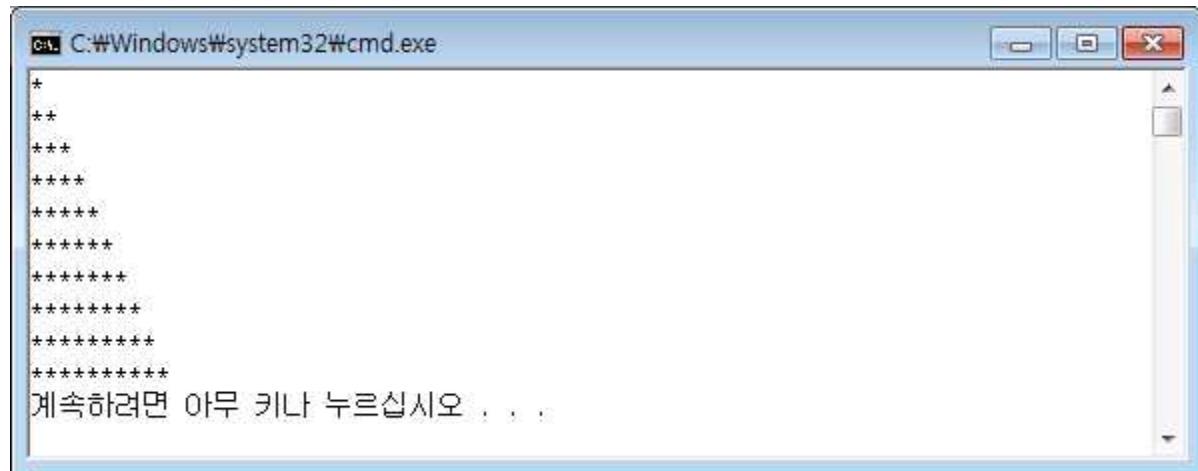
```
for i in range(1,10) :  
    print(2, "x", i, "=", 2*i)      #2단 출력  
for i in range(1,10) :  
    print(3, "x", i, "=", 3*i)      #3단 출력  
for i in range(1,10) :  
    print(4, "x", i, "=", 4*i)      #4단 출력  
... 중략  
for i in range(1,10) :  
    print(9, "x", i, "=", 9*i)      #9단 출력
```

위의 구문들을 잘 살펴보면 각각의 `for` 구문에서 달라지는 부분은 2부터 9까지 변하는 두 군데이다. 이제는 위 9개의 다시 하나의 반복문으로 묶는 일만 남았다. 다음 구문을 잘 살펴보면서 이중 반복문을 이해하도록 하자.

```
for dan in range(2,10) :  
    for i in range(1,10) :  
        print(dan, "x", i, "=", dan*i)  #dan에 해당되는 단 출력
```

2개의 `for` 구문 중에서 바깥의 반복문은 2단부터 9단까지 증가하는 반복을 처리하고 있으며, 안쪽의 반복문은 각각의 단을 출력하는데 있어서 곱해지는 수 1부터 9까지를 반복하고 있다. 이중 반복문에서는 바깥쪽의 반복문 인덱스와 안쪽의 반복문 인덱스를 바르게 결정하는 것이 가장 중요하다고 할 수 있다.

한 문제를 더 풀어보자. 다음과 같이 삼각형 모양의 별을 화면에 출력하려면 어떻게 해야 할까?



이런 경우에 무엇을 바깥쪽 반복문으로 처리하고, 무엇을 안쪽 반복문으로 처리해야 할지를 찾아야 한다. 출력된 모양을 자세히 살펴보면 다음과 같은 규칙을 찾아낼 수 있다.

첫 줄에는 별 1개, 두 번째 줄에는 별 2개, 세 번째 줄에는 별 3개, ... 이렇게 해서 10번째 줄에는 별 10개가 출력된다. 이를 `for` 구문으로 표현하면 다음과 같다.

```
for count in range(1,11) :
    #이 구문에서 별을 count 수만큼 출력한다.
```

그렇다면 별을 `count` 수만큼 출력하려면 어떻게 해야 할까? 파이썬에서는 다음과 같이 하면 쉽게 문자를 여러번 반복하여 출력할 수 있다.

```
print("*" * count)
```

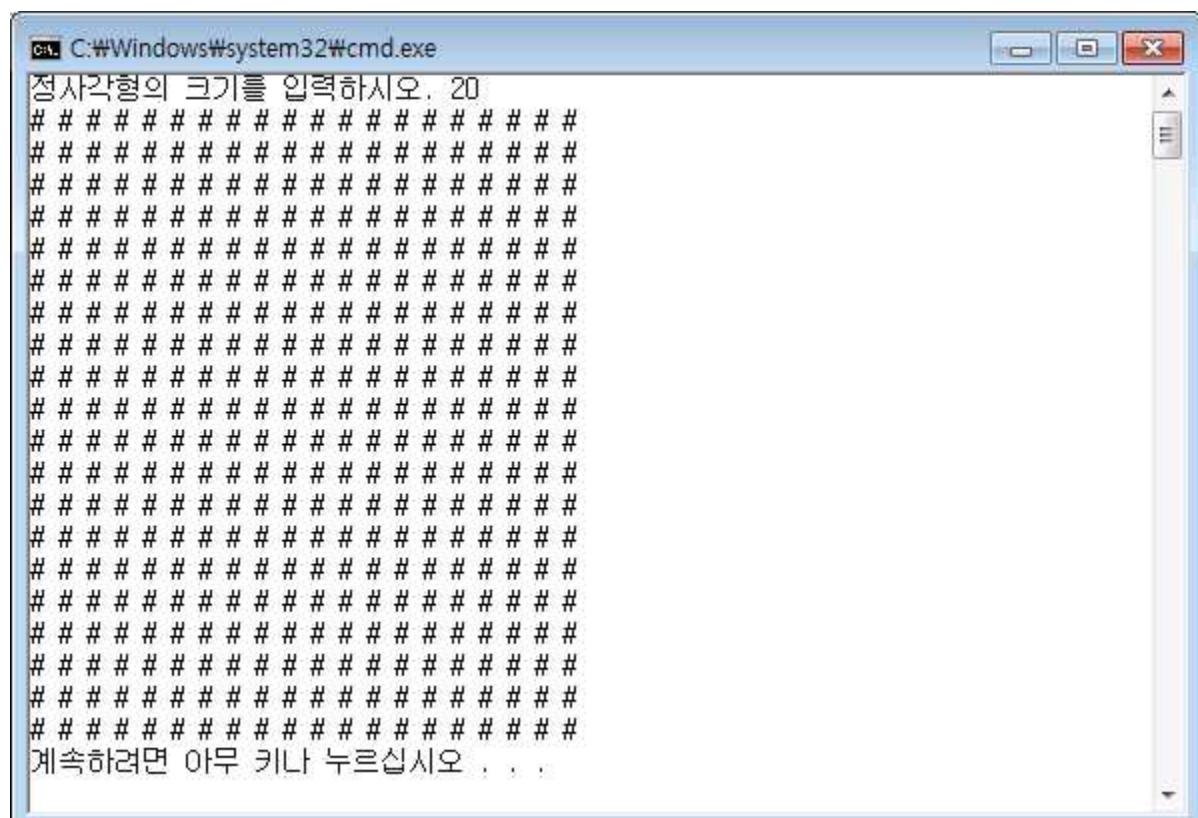
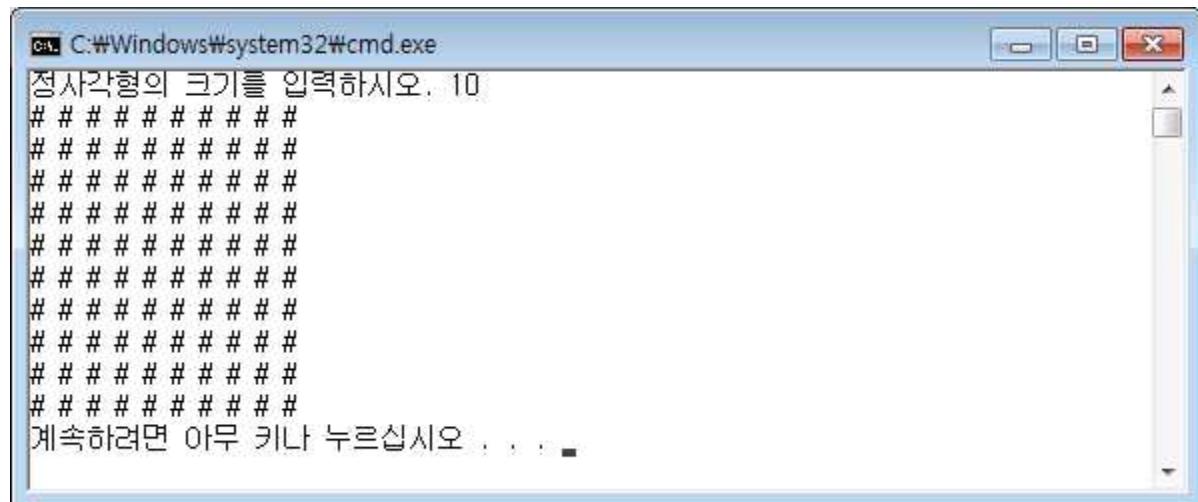
이제는 위 2개의 구문을 하나의 프로그램으로 묶어주면 된다.

```
# -*- coding: utf-8 -*-
# 예제 ex_E.py
for count in range(1,11) :
    print("*" * count)
```

## [E01] 입력한 숫자 크기의 정사각형 출력하기

숫자를 하나 입력받은 후에 이 숫자만큼의 크기를 갖는 정사각형을 '#' 문자로 화면에 출력하라. 예를 들어 10을 입력하면 10개의 '#' 문자가 들어있는 라인 10개를 출력하는 것이다.  
변수는 다음과 같이 사용하라.

```
length          # 입력받은 정사각형 한 변의 길이  
i               # 반복문 사용을 위한 변수
```

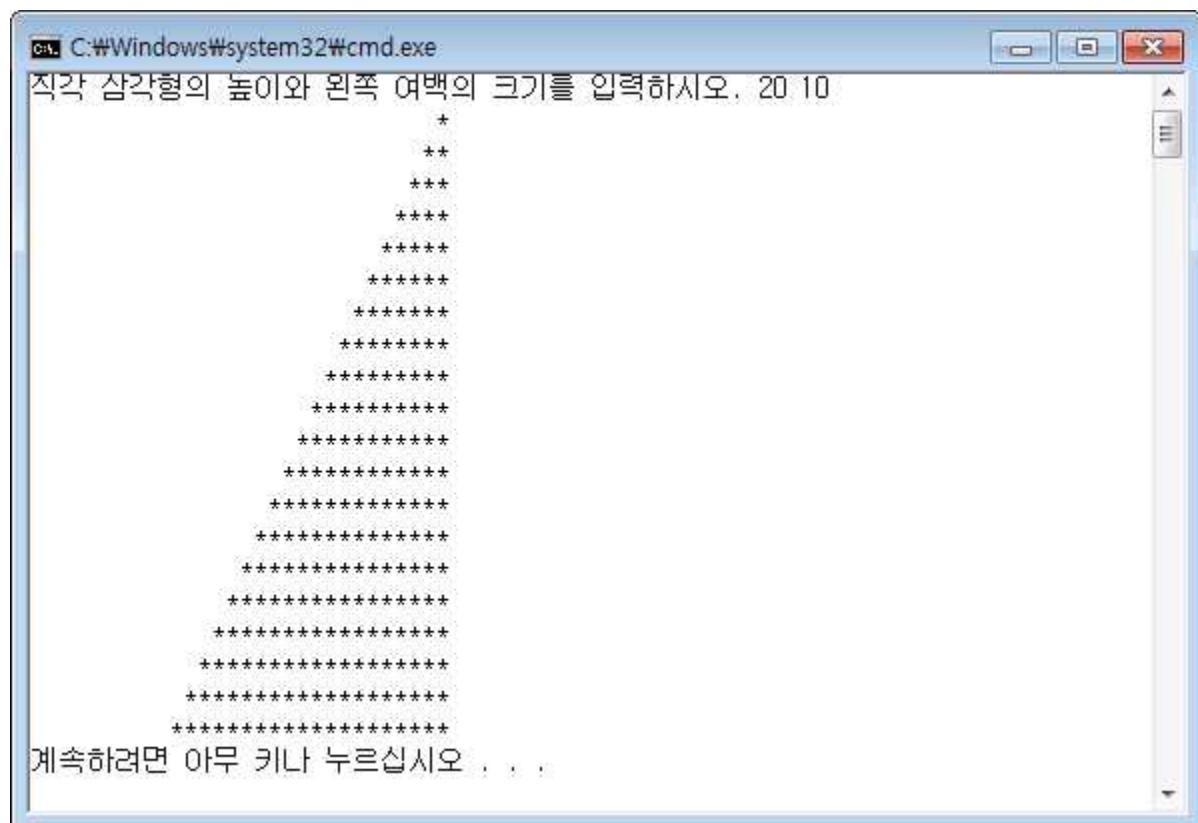


**[E02] 입력한 숫자 크기의 높이를 갖는 우직각 삼각형 출력하기**

높이와 여백을 정하는 숫자 두 개를 입력받은 후에 이 숫자만큼의 높이와 왼쪽 여백을 갖는 우직각 삼각형을 '\*' 문자로 화면에 출력하라. 예를 들어 10을 입력하면 첫 줄에는 1개, 2번째 줄에는 2개, 3번째 줄에는 3개, .. 10번째 줄에는 10개의 '\*' 을 왼쪽 여백을 가진 우측 정렬된 모습으로 출력하는 것이다.

변수는 다음과 같이 사용하라.

```
height          # 입력받은 높이  
blank          # 입력받은 여백 크기  
i, j           # 반복문 사용을 위한 변수
```

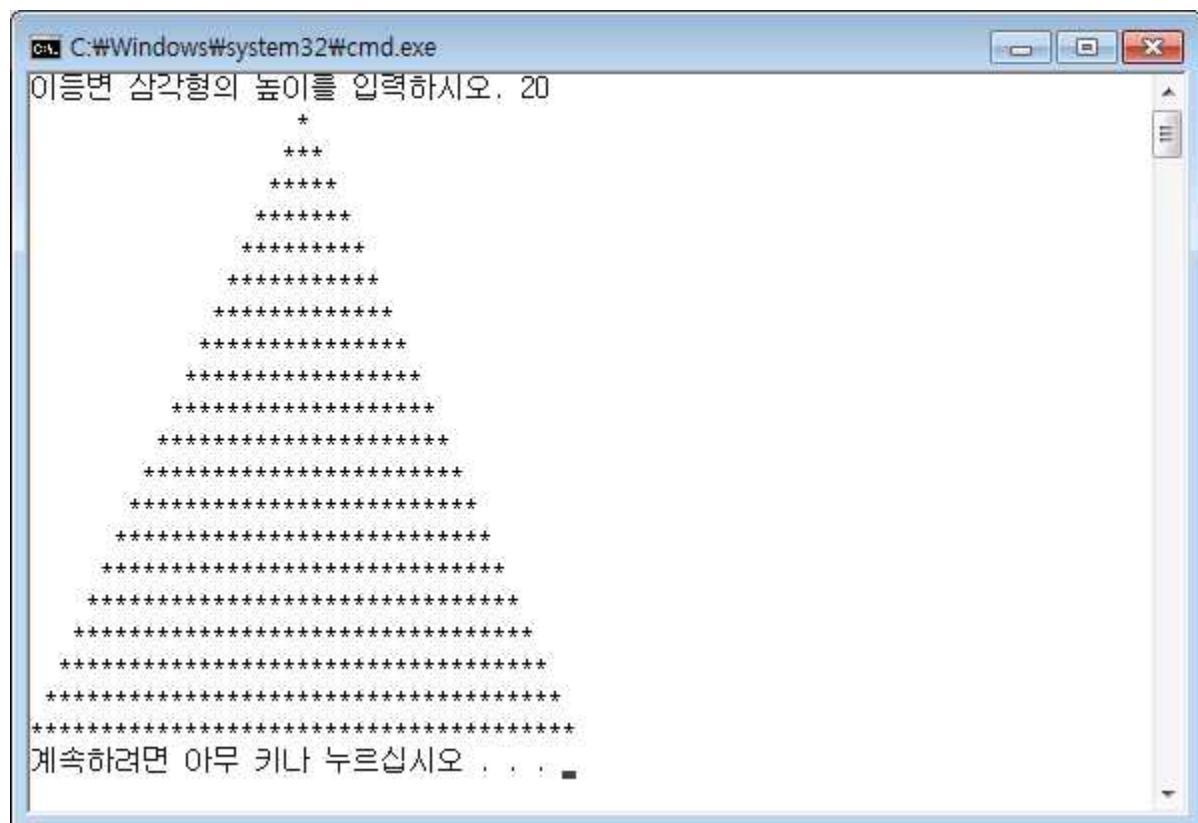


### [E03] 입력한 숫자 크기의 높이를 갖는 이등변 삼각형 출력하기

숫자를 하나 입력받은 후에 이 숫자만큼의 높이를 갖는 이등변삼각형을 '\*' 문자로 화면에 출력하라. 예를 들어 10을 입력하면 첫 줄에는 1개, 2번째 줄에는 3개, 3번째 줄에는 5개, ..., 10번째 줄에는 19개의 '\*' 을 가운데 정렬한 모습으로 출력하는 것이다.

변수는 다음과 같이 사용하라.

```
height          # 입력받은 높이  
i, j           # 반복문 사용을 위한 변수
```



### [E04] 홀수단 또는 짝수단의 구구단 출력하기

출력모드(홀수 또는 짝수)를 입력받아 이에 따라 홀수 단 또는 짝수 단의 구구단 만을 1줄에 3개씩 출력하라. 단, 출력모드 입력 내용이 1이면 홀수단, 2이면 짝수단으로 결정하도록 한다.

변수는 다음과 같이 사용하라.

```
mode          # 출력모드(1: 홀수단, 2: 짝수단)
i, j          # 반복문 사용을 위한 변수
```

C:\Windows\system32\cmd.exe

구구단의 출력모드(1: 홀수단, 2: 짝수단)를 입력하시오. 1

3 x 1 = 3	3 x 2 = 6	3 x 3 = 9
3 x 4 = 12	3 x 5 = 15	3 x 6 = 18
3 x 7 = 21	3 x 8 = 24	3 x 9 = 27
5 x 1 = 5	5 x 2 = 10	5 x 3 = 15
5 x 4 = 20	5 x 5 = 25	5 x 6 = 30
5 x 7 = 35	5 x 8 = 40	5 x 9 = 45
7 x 1 = 7	7 x 2 = 14	7 x 3 = 21
7 x 4 = 28	7 x 5 = 35	7 x 6 = 42
7 x 7 = 49	7 x 8 = 56	7 x 9 = 63
9 x 1 = 9	9 x 2 = 18	9 x 3 = 27
9 x 4 = 36	9 x 5 = 45	9 x 6 = 54
9 x 7 = 63	9 x 8 = 72	9 x 9 = 81

계속하려면 아무 키나 누르십시오 . . .

C:\Windows\system32\cmd.exe

구구단의 출력모드(1: 홀수단, 2: 짝수단)를 입력하시오. 2

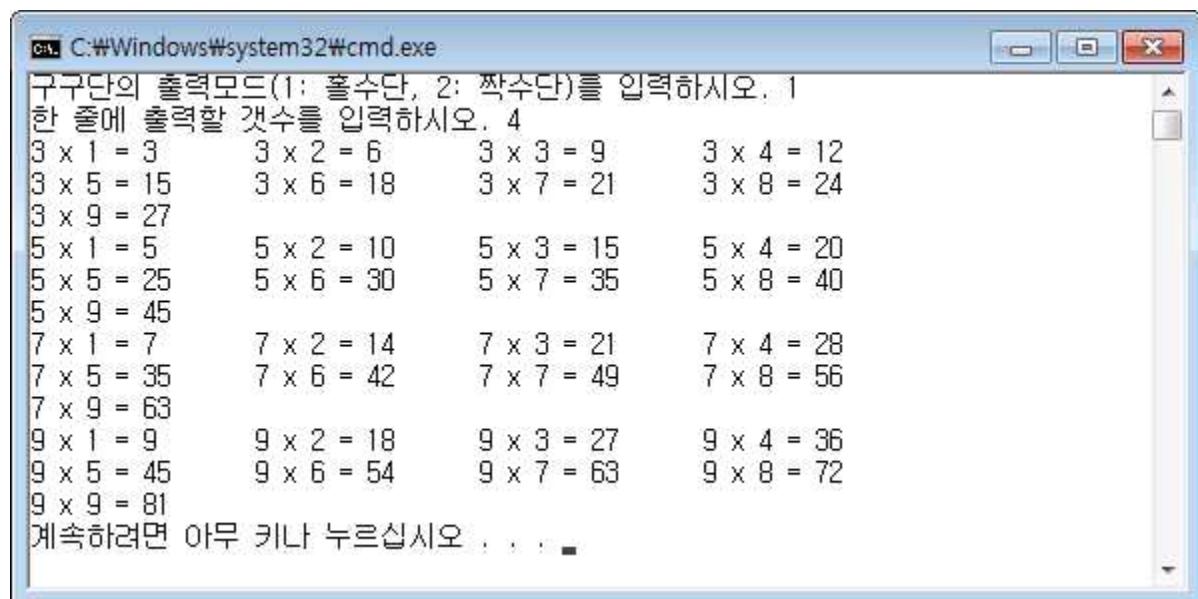
2 x 1 = 2	2 x 2 = 4	2 x 3 = 6
2 x 4 = 8	2 x 5 = 10	2 x 6 = 12
2 x 7 = 14	2 x 8 = 16	2 x 9 = 18
4 x 1 = 4	4 x 2 = 8	4 x 3 = 12
4 x 4 = 16	4 x 5 = 20	4 x 6 = 24
4 x 7 = 28	4 x 8 = 32	4 x 9 = 36
6 x 1 = 6	6 x 2 = 12	6 x 3 = 18
6 x 4 = 24	6 x 5 = 30	6 x 6 = 36
6 x 7 = 42	6 x 8 = 48	6 x 9 = 54
8 x 1 = 8	8 x 2 = 16	8 x 3 = 24
8 x 4 = 32	8 x 5 = 40	8 x 6 = 48
8 x 7 = 56	8 x 8 = 64	8 x 9 = 72

계속하려면 아무 키나 누르십시오 . . .

### [E05] 훌수단 또는 짹수단의 구구단을 열의 개수를 맞추어 출력하기

출력모드(훌수 또는 짹수)와 열 갯수를 입력받아 이에 따라 훌수 단 또는 짹수 단의 구구단만을 1줄에 열 개수만큼씩 출력하라. 단, 출력모드 입력 내용이 1이면 훌수단, 2이면 짹수단으로 결정하도록 한다.  
변수는 다음과 같이 사용하라.

```
mode          # 출력모드(1: 훌수단, 2: 짹수단)
column        # 열 개수
i, j          # 반복문 사용을 위한 변수
```



### [E06] 2차원 숫자 출력하기

행의 크기(rows)와 열의 크기(columns)를 입력받은 후에, 이 크기만큼의 바둑판 모양의 2차원 공간의 각 칸마다 행 번호(1, 2, ..., width)와 열 번호(1, 2, ..., height)를 곱한 값을 출력하라. (아래 그림 참고)  
변수는 다음과 같이 사용하라.

rows, columns	# 행의 개수, 열의 개수
number	# 각 칸에 출력하는 값
i, j	# 반복문 사용을 위한 변수

```
C:\>Windows\system32\cmd.exe
총력하려는 행의 크기와 열의 크기를 입력하시오. 4 5
 1  2  3  4  5
 2  4  6  8  10
 3  6  9  12 15
 4  8  12 16 20
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\>Windows\system32\cmd.exe
총력하려는 행의 크기와 열의 크기를 입력하시오. 10 15
 1  2  3  4  5  6  7  8  9  10 11 12 13 14 15
 2  4  6  8  10 12 14 16 18 20 22 24 26 28 30
 3  6  9  12 15 18 21 24 27 30 33 36 39 42 45
 4  8  12 16 20 24 28 32 36 40 44 48 52 56 60
 5  10 15 20 25 30 35 40 45 50 55 60 65 70 75
 6  12 18 24 30 36 42 48 54 60 66 72 78 84 90
 7  14 21 28 35 42 49 56 63 70 77 84 91 98 105
 8  16 24 32 40 48 56 64 72 80 88 96 104 112 120
 9  18 27 36 45 54 63 72 81 90 99 108 117 126 135
10 20 30 40 50 60 70 80 90 100 110 120 130 140 150
계속하려면 아무 키나 누르십시오 . . .
```

## [Step F] 리스트 사용하기

대부분의 프로그램에서는 비슷한 용도로 사용되는 여러 개의 변수가 필요한 경우에 배열이라는 루음형 변수로 선언하면 아주 편리하다. 파이썬에서는 리스트를 사용하면 이러한 루음형 데이터들을 쉽게 다룰 수 있다. 예를 들어 5개의 정수형 변수를 사용해서 숫자 5개를 입력받아 그 합계를 계산 하려면 다음과 같다.

```
sum = 0
num1 = int(input("1번 숫자를 입력하시오. "))
sum = sum + num1
num2 = int(input("2번 숫자를 입력하시오. "))
sum = sum + num2
num3 = int(input("3번 숫자를 입력하시오. "))
sum = sum + num3
num4 = int(input("4번 숫자를 입력하시오. "))
sum = sum + num4
num5 = int(input("5번 숫자를 입력하시오. "))
sum = sum + num5

print("숫자의 합계는", sum, "입니다.")
```

위의 프로그램을 자세히 들여다보면 다섯 개의 숫자에 대해 비슷한 모양의 구문이 반복되는 것을 볼 수 있다. 여기에서 다섯 개의 숫자를 위한 변수를 리스트로 선언해서 프로그램을 변경하면 다음과 같다.

```
num = [] # 리스트를 초기화
sum = 0
for i in range(5) : # i 가 0부터 4까지 5회 반복하며 실행
    newnum = int(input("%d번 숫자를 입력하시오. %(i+1)"))
    num.append(newnum) # 리스트에 입력받은 숫자를 추가함
for i in range(5) : # i 가 0부터 4까지 5회 반복하며 실행
    sum = sum + num[i] # 리스트에서 인덱스 i에 해당되는 수를 누적
print("숫자의 합계는", sum, "입니다.")
```

위 구문에서 `int(input("%d번 숫자를 입력하시오. %(i+1)"))`를 보면 문자열 내에 `%d`라고 표시한 부분은 형식 지시자라고 부르는데 이곳에 문자열 뒤에 따라오는 `%(i+1)`의 값으로 대체하라는 의미이다.

`num[i]`와 같이 리스트 안에 들어있는 요소를 접근할 때에는 인덱스 값이 언제나 0부터 시작한다

는 점을 유의해야 한다. 위의 2종류의 프로그램 구문은 동일하게 다음 화면과 같은 실행 결과를 보인다. 하지만 두 번째 프로그램에서는 입력받은 5개의 숫자가 리스트 num에 한꺼번에 보관된다.

```
C:\Windows\system32\cmd.exe
1번 숫자를 입력하시오. 11
2번 숫자를 입력하시오. 22
3번 숫자를 입력하시오. 33
4번 숫자를 입력하시오. 44
5번 숫자를 입력하시오. 55
숫자의 합계는 165입니다.
계속하려면 아무 키나 누르십시오 . . .
```

이번에는 리스트 내에 다시 리스트가 들어 있는 중복 구조를 살펴보자. 예를 들어 다음과 같이 리스트를 만들어 사용한다고 가정해보자.

```
number = [[1,2,3],[4,5,6],[7,8,9],[10,11,12]]
```

위 배열의 구조를 그림으로 표시하면 다음 그림과 같다.

number[0]	number[0][0]	number[0][1]	number[0][2]
[1,2,3]	1	2	3
number[1]	number[1][0]	number[1][1]	number[1][2]
[4,5,6]	4	5	6
number[2]	number[2][0]	number[2][1]	number[2][2]
[7,8,9]	7	8	9
number[3]	number[3][0]	number[3][1]	number[3][2]
[10,11,12]	10	11	12

그렇다면 반복문을 사용해서 위 리스트에 값을 넣으려면 어떻게 하면 될까? 우선 다음과 같은 방법을 생각해볼 수 있다.

```
number = []
number.append([1, 2, 3])
number.append([4, 5, 6])
number.append([7, 8, 9])
number.append([10, 11, 12])
```

위 구문을 자세히 살펴보면 매 줄마다 3개의 요소를 갖는 작은 리스트를 number 리스트에 추가하고 있음을 볼 수 있다. 그리고 작은 리스트는 각각 1, 4, 7, 10으로 시작하는 3개의 숫자들로

이뤄지고 있다. 그러므로 이를 처리하는 구문은 다음과 같다.

```
number = []
for i in range(1,12,3) :          # 1부터 11까지 3씩 증가하는 숫자들(1,4,7,10)에 대해 반복
    small_list = []               # 작은 리스트 초기화
    for j in range(3):           # 0부터 2까지 3개의 숫자들에 대해 반복
        small_list.append(i+j)    # 작은 리스트에 요소를 추가
    number.append(small_list)     # 만들어진 작은 리스트를 number 리스트에 추가
print(number)
```

이렇듯 반복문과 리스트를 사용하여 프로그램을 만들 때에는 인덱스를 사용하는 부분을 정확하게 파악하여 프로그램에 적용하는 것이 중요하다.

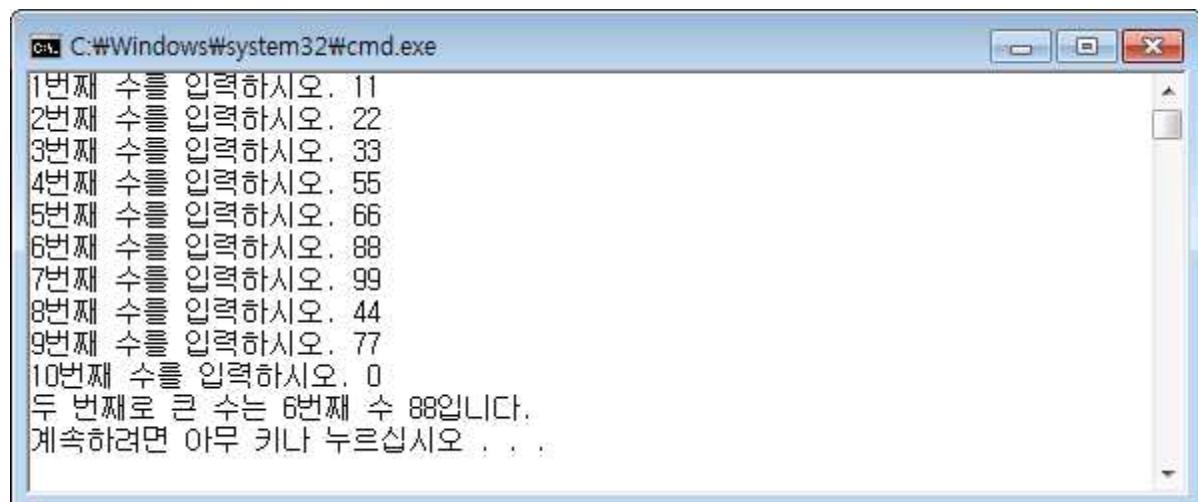
## ○ 실습 문제

### [F01] 두 번째로 큰 수의 순서 찾기

10개의 숫자를 입력받아 리스트에 저장한 후에 이 중에서 두 번째로 큰 수가 몇 번째 숫자인지 찾아내어 출력하라.

변수는 다음과 같이 사용하라.

```
num = []          # 10개의 숫자를 담을 리스트
first           # 첫 번째로 큰 수
second          # 두 번째로 큰 수
second_max_index # 두 번째로 큰 수의 인덱스
newnum          # 숫자 입력을 위한 변수
i               # 반복문을 위한 변수
```

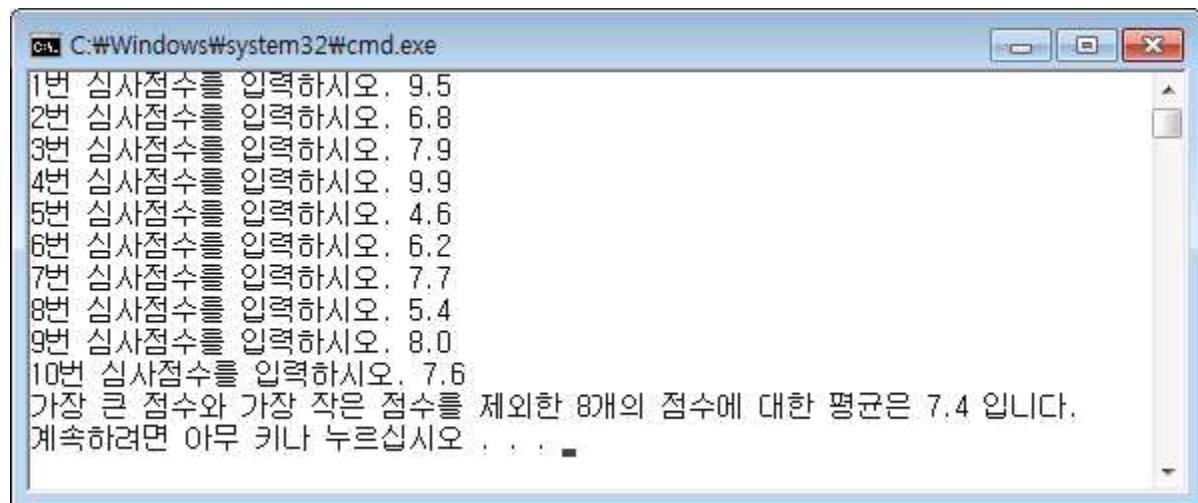


## [F02] 심사점수 계산

심사점수를 10개 입력받아 리스트에 저장한 후, 이 중에서 가장 큰 점수와 가장 작은 점수를 제외한 8개의 점수에 대한 평균을 계산하여 출력하라.

변수는 다음과 같이 사용하라.

```
score = []          # 심사점수 리스트  
maxscore, minscore # 가장 큰 점수, 가장 작은 점수  
total             # 점수 총 합계  
average           # 평균점수  
newnum            # 숫자 입력을 위한 변수  
i                 # 반복문을 위한 변수
```



### [F03] 5명의 국, 영, 수 3과목 점수의 과목별 총점, 평균값 구하기

학생 5명의 국어, 영어, 수학 점수를 각각 입력받아 저장한 후에, 각 과목별 총점과 평균 점수를 계산하여 출력하라.

변수는 다음과 같이 사용하라.

```
jumsu=[]          # 5명의 3과목 점수를 저장하고 있는 중복 리스트
sum=[]            # 3과목 총점 리스트
average=[]        # 3과목 평균 리스트
kor, eng, mat    # 3과목 점수 입력을 위한 변수
i, j              # 반복문을 위한 변수
```

The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the following output:

```
1번 학생 국어, 영어, 수학 점수를 입력하시오. 95 85 75
2번 학생 국어, 영어, 수학 점수를 입력하시오. 87 79 88
3번 학생 국어, 영어, 수학 점수를 입력하시오. 95 85 75
4번 학생 국어, 영어, 수학 점수를 입력하시오. 65 77 79
5번 학생 국어, 영어, 수학 점수를 입력하시오. 77 88 99
국어의 총점은 419이고, 평균은 83.8입니다.
영어의 총점은 414이고, 평균은 82.8입니다.
수학의 총점은 416이고, 평균은 83.2입니다.
계속하려면 아무 키나 누르십시오 . . .
```

### [F04] 5명의 국, 영, 수 3과목 점수의 학생별 총점, 평균값 구하기

학생 5명의 국어, 영어, 수학 점수를 각각 입력받아 저장한 후에, 각 과목별 총점과 평균 점수를 계산하여 출력하라.

변수는 다음과 같이 사용하라.

```
jumsu=[]          # 5명의 3과목 점수를 저장하고 있는 중복 리스트  
sum=[]            # 3과목 총점 리스트  
average=[]         # 3과목 평균 리스트  
kor, eng, mat     # 3과목 점수 입력을 위한 변수  
i, j              # 반복문을 위한 변수
```

The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the output of a Python script. The script asks for five students' scores and calculates their total and average. The output is as follows:

```
1번 학생 국어, 영어, 수학 점수를 입력하시오. 95 85 75  
2번 학생 국어, 영어, 수학 점수를 입력하시오. 87 79 88  
3번 학생 국어, 영어, 수학 점수를 입력하시오. 95 85 75  
4번 학생 국어, 영어, 수학 점수를 입력하시오. 65 77 79  
5번 학생 국어, 영어, 수학 점수를 입력하시오. 77 88 99  
1번 학생의 총점은 255이고, 평균은 85.0입니다.  
2번 학생의 총점은 254이고, 평균은 84.7입니다.  
3번 학생의 총점은 255이고, 평균은 85.0입니다.  
4번 학생의 총점은 221이고, 평균은 73.7입니다.  
5번 학생의 총점은 264이고, 평균은 88.0입니다.  
계속하려면 아무 키나 누르십시오 . . .
```

## [F05] 비만 판정

10명의 신장(cm단위)과 체중(kg단위)를 입력받은 후, 이들 중 몇 번째 사람들이 비만인지를 판정하여 출력하라. 그리고 도합 몇 명이 비만인지 출력하라.

단, 비만여부는 다음 비만도 수치가 25이상인 경우에 "비만"으로 판단한다.

비만도 수치 = 체중(kg) / (신장(m)의 제곱) 으로 계산한다. 이 때, 신장은 미터 단위로 환산해야 함을 유의하라.

변수는 다음과 같이 사용하라.

```
height, weight          # 입력받은 신장(cm), 체중(kg)
bmi                   # 계산된 비만도 수치
bmilist = []           # 10명에 대한 신장, 체중, 비만도수치를 담고 있는 리스트
count                # 비만인 사람의 숫자
i                     # 반복문을 위한 변수
```

The screenshot shows a command-line interface window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the following text:

```
1번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 166 56
2번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 176 90
3번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 187 60
4번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 155 47
5번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 170 75
6번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 173 80
7번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 165 50
8번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 182 73
9번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 158 48
10번째 사람의 신장(cm)과 체중(kg)을 입력하시오. 170 66
2번째 사람은 비만입니다.
5번째 사람은 비만입니다.
6번째 사람은 비만입니다.

총 3명의 사람이 비만입니다.
계속하려면 아무 키나 누르십시오 . . .
```

### [F06] 5층 아파트의 거주자 숫자 구하기

한 층에 3집(1호, 2호, 3호)으로 되어 있는 5층짜리 아파트가 있다. 2차원 배열을 사용하여 101호부터 503호까지 각 집에 살고 있는 사람의 숫자를 입력받아 보관하라. 그리고 이 아파트에 사는 거주자는 모두 몇 명인지 출력하라.

변수는 다음과 같이 사용하라.

```
number = []          # 각 집의 거주자 수, 층별, 호수별 중복 리스트
newnum             # 입력받은 숫자
total              # 아파트의 총 거주자 수
ho                 # 아파트 호를 나타내는 변수
i, j               # 반복문 사용을 위한 변수
```

The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the output of a Python script. The script asks for the number of residents in each apartment from 101 to 503 and calculates the total number of residents. The output is as follows:

```
101호에 살고 있는 사람의 숫자를 입력하시오. 3
102호에 살고 있는 사람의 숫자를 입력하시오. 4
103호에 살고 있는 사람의 숫자를 입력하시오. 6
201호에 살고 있는 사람의 숫자를 입력하시오. 2
202호에 살고 있는 사람의 숫자를 입력하시오. 7
203호에 살고 있는 사람의 숫자를 입력하시오. 5
301호에 살고 있는 사람의 숫자를 입력하시오. 7
302호에 살고 있는 사람의 숫자를 입력하시오. 4
303호에 살고 있는 사람의 숫자를 입력하시오. 3
401호에 살고 있는 사람의 숫자를 입력하시오. 4
402호에 살고 있는 사람의 숫자를 입력하시오. 3
403호에 살고 있는 사람의 숫자를 입력하시오. 2
501호에 살고 있는 사람의 숫자를 입력하시오. 1
502호에 살고 있는 사람의 숫자를 입력하시오. 3
503호에 살고 있는 사람의 숫자를 입력하시오. 4
이 아파트에 사는 거주자는 모두 58명 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

### [F07] 5층 아파트의 층별, 호수별 거주자 숫자 구하기

한 층에 3집(1호, 2호, 3호)으로 되어 있는 5층짜리 아파트가 있다. 2차원 배열을 사용하여 101호부터 503호까지 각 집에 살고 있는 사람의 숫자를 입력받아 보관하라. 그리고 이 아파트에 사는 거주자의 숫자를 층별(1층~5층)로 합산하여 출력하고, 호수별(1호~3호)로 합산하여 출력하라. 예를 들어 1층 거주자의 수는 101호, 102호, 103호 거주자의 수를 합한 것이고, 2호 라인 거주자의 수는 102호, 202호, 302호, 402호, 502호 거주자의 수를 합한 것이다.

변수는 다음과 같이 사용하라.

```
number = []          # 각 집의 거주자 수, 층별, 호수별 중복 리스트
newnum             # 입력받은 숫자
floor_total = []    # 층별 거주자 합계 리스트 (1층, 2층, 3층, 4층, 5층)
line_total = []     # 호수별 거주자 합계 리스트 (1호라인, 2호라인, 3호라인)
total              # 아파트의 총 거주자 수
ho                 # 아파트 호를 나타내는 변수
i, j               # 반복문 사용을 위한 변수
```

The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the output of a Python script. The script prompts for input for each unit from 101 to 503, then calculates and prints the total number of residents per floor and per line (set of three units). The output is as follows:

```

103호에 살고 있는 사람의 숫자를 입력하시오. 5
201호에 살고 있는 사람의 숫자를 입력하시오. 4
202호에 살고 있는 사람의 숫자를 입력하시오. 3
203호에 살고 있는 사람의 숫자를 입력하시오. 6
301호에 살고 있는 사람의 숫자를 입력하시오. 5
302호에 살고 있는 사람의 숫자를 입력하시오. 7
303호에 살고 있는 사람의 숫자를 입력하시오. 4
401호에 살고 있는 사람의 숫자를 입력하시오. 3
402호에 살고 있는 사람의 숫자를 입력하시오. 2
403호에 살고 있는 사람의 숫자를 입력하시오. 5
501호에 살고 있는 사람의 숫자를 입력하시오. 6
502호에 살고 있는 사람의 숫자를 입력하시오. 4
503호에 살고 있는 사람의 숫자를 입력하시오. 3
1층에 사는 거주자는 모두 10명 입니다.
2층에 사는 거주자는 모두 13명 입니다.
3층에 사는 거주자는 모두 16명 입니다.
4층에 사는 거주자는 모두 10명 입니다.
5층에 사는 거주자는 모두 13명 입니다.

1호 라인에 사는 거주자는 모두 20명 입니다.
2호 라인에 사는 거주자는 모두 19명 입니다.
3호 라인에 사는 거주자는 모두 23명 입니다.

이 아파트에 사는 거주자는 모두 62명 입니다.
계속하려면 아무 키나 누르십시오 . .

```

### [F08] 겹치지 않는 숫자 10개 입력 받기

사용자에게 1부터 100사이의 숫자를 10개 입력받아 이를 순서대로 출력하라. 단, 사용자가 입력하는 동안 이미 입력한 숫자와 같은 숫자를 입력하면 "잘못 입력하였습니다. 다시 입력하세요."라는 문구와 함께 다시 입력받도록 하라. 입력이 완료되면 10개의 수를 모두 출력하라.

변수는 다음과 같이 사용하라.

```
number = []          # 사용자가 입력한 숫자 10개
newnum             # 입력받은 숫자
count              # 현재까지 입력된 숫자의 개수(0~10)
dup                # 중복검사 통과 여부 (True : 중복, False : 중복없음)
i                  # 반복문을 위한 변수
```

```
C:\Windows\system32\cmd.exe
1부터 100사이의 숫자를 입력하시오.
1번째 숫자를 입력하시오. 1
2번째 숫자를 입력하시오. 1
잘못 입력하였습니다. 다시 입력하세요.
2번째 숫자를 입력하시오. 2
3번째 숫자를 입력하시오. 4
4번째 숫자를 입력하시오. 6
5번째 숫자를 입력하시오. 4
잘못 입력하였습니다. 다시 입력하세요.
5번째 숫자를 입력하시오. 2
잘못 입력하였습니다. 다시 입력하세요.
5번째 숫자를 입력하시오. 7
6번째 숫자를 입력하시오. 10
7번째 숫자를 입력하시오. 35
8번째 숫자를 입력하시오. 10
잘못 입력하였습니다. 다시 입력하세요.
8번째 숫자를 입력하시오. 23
9번째 숫자를 입력하시오. 43
10번째 숫자를 입력하시오. 35
잘못 입력하였습니다. 다시 입력하세요.
10번째 숫자를 입력하시오. 33
1번째 숫자는 1입니다
2번째 숫자는 2입니다
3번째 숫자는 4입니다
4번째 숫자는 6입니다
5번째 숫자는 7입니다
6번째 숫자는 10입니다
7번째 숫자는 35입니다
8번째 숫자는 23입니다
9번째 숫자는 43입니다
10번째 숫자는 33입니다
계속하려면 아무 키나 누르십시오 . . .
```

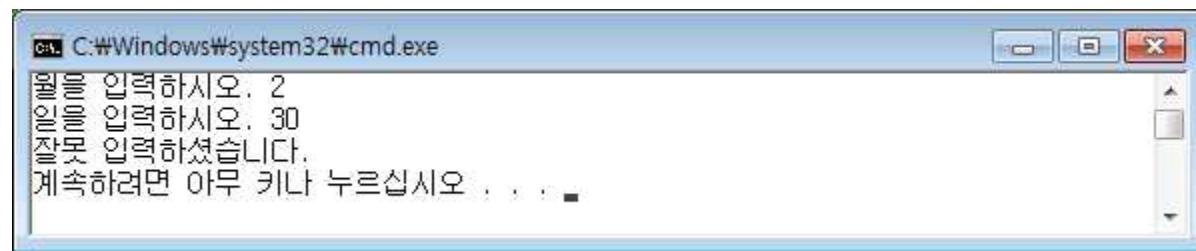
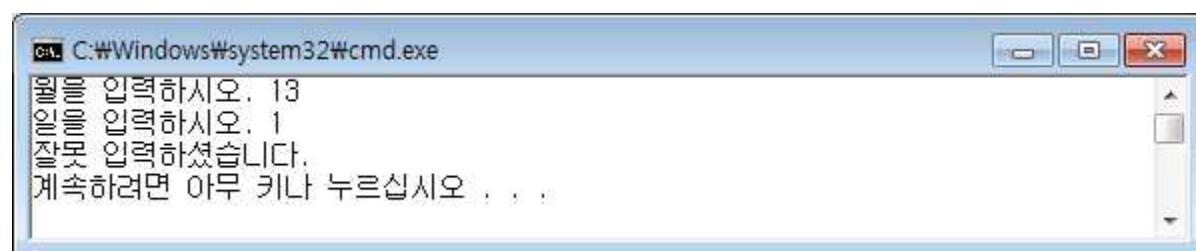
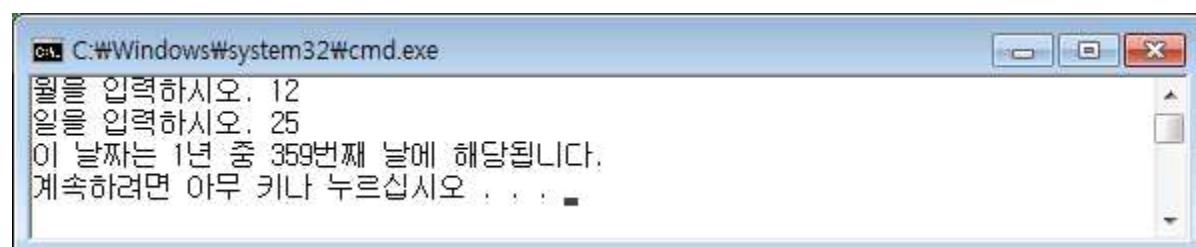
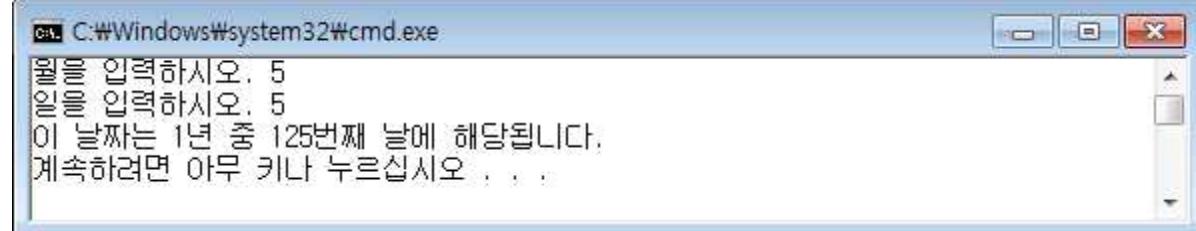
### [F09] 리스트를 이용한 연중 날짜 계산

날짜를 월과 일로 입력받아 이 날짜는 1년 중 몇 번째 날에 해당되는지 계산하여 출력하라.

단, 매 월의 날 수를 다음과 같이 리스트로 만들어 이를 이용하여 계산하라.

변수는 다음과 같이 사용하라.

```
monthdays = [31,28,31,30,31,30,31,31,30,31,30,31]      # 1~12월의 날 수
month, day      # 입력받은 월, 일
day_count       # 1년 중 날 수
i               # 반복문을 위한 변수
```



## [Step G] 조건과 반복을 활용하는 응용 예제 해결하기

이번 단계에서는 그동안 연습해 온 조건문과 반복문을 함께 응용하여 해결해야 하는 문제들을 풀어보려고 한다. 예를 들어서 어떤 레스토랑에서 하루 동안의 식사 금액을 계산하는 프로그램을 만들여보자. 이 식당에는 4가지 메뉴가 있으며 각 메뉴별 가격은 다음과 같다.

- 1. 피자(15,000원), 2. 스파게티(10,000원), 3. 샐러드(7,000원), 4. 음료수(2,000원)

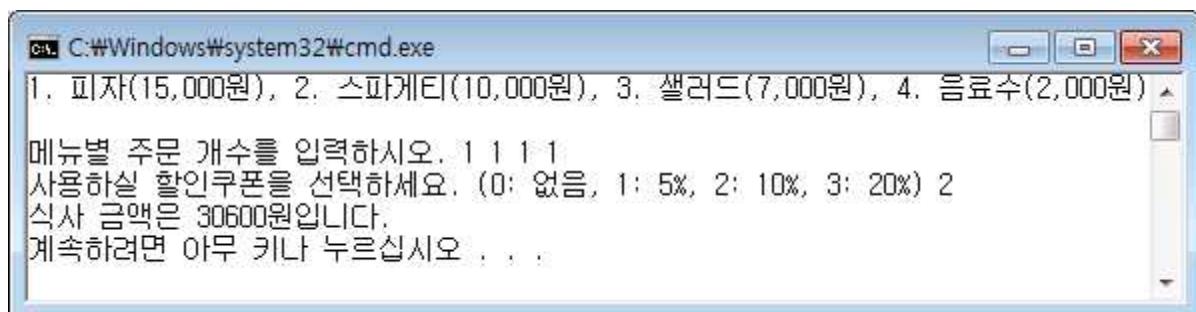
어떤 한 팀의 주문서 내용 즉, 메뉴별 주문 개수를 입력받아 식대를 계산하는 구문은 다음과 같다.

```
price = [15000, 10000, 7000, 2000]           # 메뉴별 가격
print("피자(15,000원), 스파게티(10,000원), 샐러드(7,000원), 음료수(2,000원)")
order = input("메뉴별 주문 개수를 입력하시오.").split()
sum = 0
for i in range(4) :
    order[i] = int(order[i])
    sum = sum + (order[i] * price[i])
print("식사 금액은 %d원입니다."%(sum))
```

그런데, 이 레스토랑에서는 할인쿠폰제도가 있어서 5%, 10%, 20% 세 종류의 할인쿠폰 중 하나를 제시하는 팀에게는 할인해 주는 제도를 시행하고 있다면, 프로그램의 하단 부분은 다음과 같이 변경 되어야 할 것이다.

```
# 위 코드와 1~7줄 동일
coupon = int(input("사용하실 할인쿠폰을 선택하세요. (0: 없음, 1: 5%, 2: 10%, 3: 20%) "))
if coupon == 1 : sum = sum * 0.95
elif coupon == 2 : sum = sum * 0.90
elif coupon == 3 : sum = sum * 0.80
print("식사 금액은 %d원입니다."%(sum))
```

현재까지 만든 프로그램의 실행 화면은 다음과 같다.



이제는 하루 동안의 매출을 계산해보자. 하루 동안 식사하러 온 팀이 모두 몇 팀인지를 입력받아 이 숫자만큼 위 프로그램을 반복해가면서 총 매출액을 계산하는 것이다. 그렇기 위해서는 다음과 같이 앞의 프로그램을 전체적으로 반복문으로 둘러싸야 한다. 그리고 매 반복 때마다 계산한 sum의 값을 누적해서 합산해야 한다.

```
# coding: euc-kr
# 예제 ex_G.py
price = [15000, 10000, 7000, 2000] # 메뉴별 가격
total_sum = 0
team = int(input("오늘 방문한 팀은 모두 몇 팀입니까? "))
print("피자(15,000원), 스파게티(10,000원), 샐러드(7,000원), 음료수(2,000원)")
print("방문한 팀 별로 메뉴별 주문 개수를 입력하세요. ")
for t in range(team) :
    order = input("%d번 팀의 메뉴별 주문 개수를 입력하시오. %(t+1)).split()
    sum = 0
    for i in range(4) :
        order[i] = int(order[i])
        sum = sum + (order[i] * price[i])
    coupon = input("사용하실 할인쿠폰을 선택하세요. (0: 없음, 1: 5%, 2: 10%, 3: 20%) ")
    if coupon == 1 : sum = sum * 0.95
    elif coupon == 2 : sum = sum * 0.90
    elif coupon == 3 : sum = sum * 0.80
    print("식사 금액은 %d원입니다."%(sum))
    total_sum = total_sum + sum

print("오늘 총 매출액은 %d원입니다."%(total_sum))
```



## ○ 실습 문제

### [G01] 나이 계산 및 연령대 판정

최대 100명까지 사람들의 태어난 년도를 입력받아 나이를 계산한 후, 그 값을 저장하되, 2012보다 큰 년도가 입력되기 전까지 태어난 년도를 반복하여 입력받도록 하라. 반복이 끝나면 지금까지 입력된 사람들의 나이를 모두 출력하고, 각자의 나이에 따라 유아, 어린이, 청소년, 청년, 중년, 노년 여부를 판정하여 이 중에 유아, 어린이, 청소년, 청년, 중년, 노년이 각각 몇 명인지 출력하라.

단, 나이 =  $2012 - \text{태어난 년도} + 1$ 로 계산하고 연령대 구분은 다음과 같이 판정한다.

7세 미만 : 유아, 7세 이상 ~ 13세미만 : 어린이, 13세 이상 ~ 20세 미만 : 청소년,

20세 이상 ~ 30세 미만 : 청년, 30세 이상 ~ 60세 미만 : 중년, 60세 이상 : 노년

변수는 다음과 같이 사용하라.

```
birth_year      # 입력받은 태어난 년도
age=[]          # 각 사람들의 나이 (최대 100명)
count_person    # 입력된 인원 수
count_baby      # 유아 수
count_child     # 어린이 수
count_youth     # 청소년 수
count_young     # 청년 수
count_adult     # 중년 수
count_old       # 노년 수
i               # 반복문을 위한 변수
```

The screenshot shows a Windows command prompt window titled 'cmd.exe' with the path 'C:\Windows\system32\cmd.exe'. The window displays the following text output:

```
1번째 사람의 태어난 년도를 입력하시오. 1989
2번째 사람의 태어난 년도를 입력하시오. 1999
3번째 사람의 태어난 년도를 입력하시오. 2002
4번째 사람의 태어난 년도를 입력하시오. 1976
5번째 사람의 태어난 년도를 입력하시오. 1950
6번째 사람의 태어난 년도를 입력하시오. 1930
7번째 사람의 태어난 년도를 입력하시오. 1945
8번째 사람의 태어난 년도를 입력하시오. 1962
9번째 사람의 태어난 년도를 입력하시오. 1974
10번째 사람의 태어난 년도를 입력하시오. 2222
1번째 사람의 나이는 24 입니다.
2번째 사람의 나이는 14 입니다.
3번째 사람의 나이는 11 입니다.
4번째 사람의 나이는 37 입니다.
5번째 사람의 나이는 63 입니다.
6번째 사람의 나이는 83 입니다.
7번째 사람의 나이는 68 입니다.
8번째 사람의 나이는 51 입니다.
9번째 사람의 나이는 39 입니다.
유아는 0명 입니다.
어린이는 1명 입니다.
청소년은 1명 입니다.
청년은 1명 입니다.
중년은 3명 입니다.
노년은 3명 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

## [G02] 물의 온도 구간 판정

물의 온도를 10회 입력받은 후, 이 물이 각각 어느 정도의 온수인지 판정하여 그 결과를 출력하라. 출력할 내용은 입력된 10개의 온도 값, 냉수 입력 횟수, 미온수 입력 횟수, 온수 입력 횟수, 끓는 물 입력 횟수를 각각 출력하라.

단, 온수의 판정 구간은 다음과 같이 판정한다.

음수 값 (0미만) : 잘못입력

0도 ~ 25도 미만 : 냉수

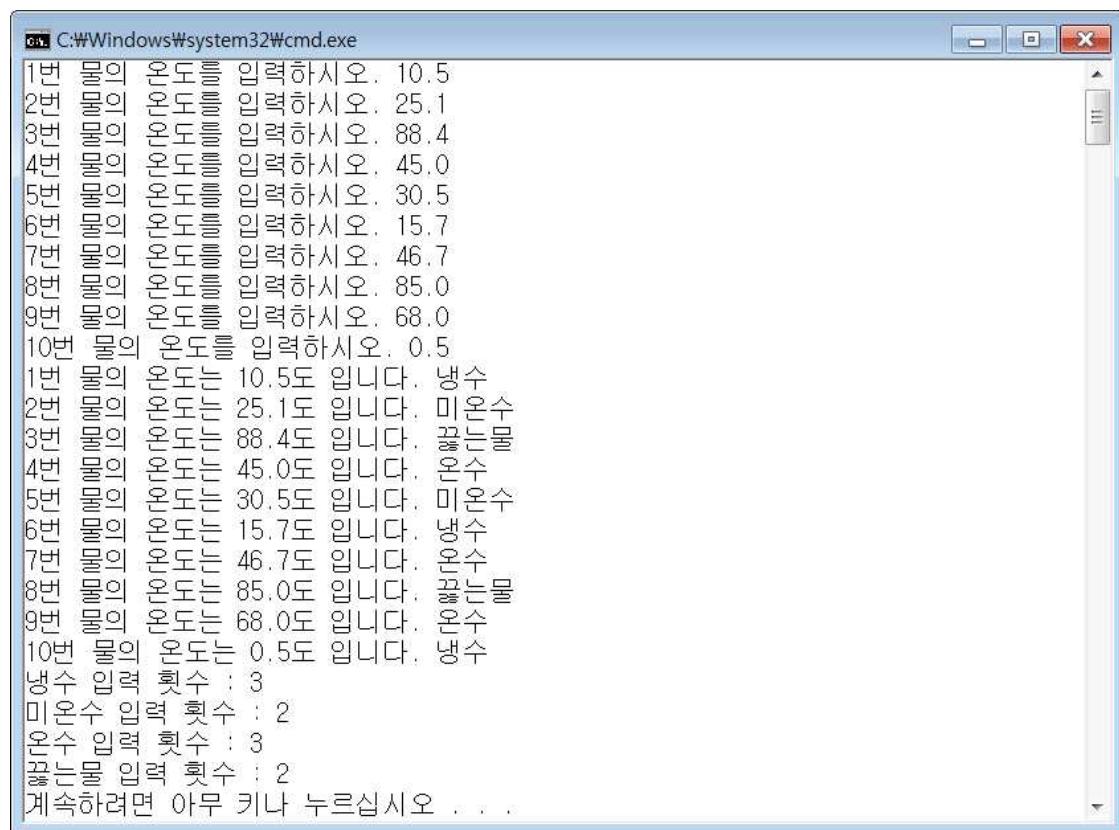
25도 ~ 40도 미만 : 미온수

40도 ~ 80도 미만 : 온수

80도 이상 : 끓는 물

변수는 다음과 같이 사용하라.

```
input_degree      # 입력받은 온도
degrees = []      # 온도 리스트
degree_name = ["냉수", "미온수", "온수", "끓는물"]
count = [0,0,0,0]  # 입력 횟수 목록 (순서대로 냉수 입력 횟수, 미온수 입력 횟수, 온수 입력 횟수,
                  # 끓는 물 입력 횟수)
i                # 반복문을 위한 변수
sel              # 온도 판정 번호 (0,1,2,3)
```



```

1번 물의 온도를 입력하시오. 10.5
2번 물의 온도를 입력하시오. 25.1
3번 물의 온도를 입력하시오. 88.4
4번 물의 온도를 입력하시오. 45.0
5번 물의 온도를 입력하시오. 30.5
6번 물의 온도를 입력하시오. 15.7
7번 물의 온도를 입력하시오. 46.7
8번 물의 온도를 입력하시오. 85.0
9번 물의 온도를 입력하시오. 68.0
10번 물의 온도를 입력하시오. 0.5
1번 물의 온도는 10.5도 입니다. 냉수
2번 물의 온도는 25.1도 입니다. 미온수
3번 물의 온도는 88.4도 입니다. 끓는물
4번 물의 온도는 45.0도 입니다. 온수
5번 물의 온도는 30.5도 입니다. 미온수
6번 물의 온도는 15.7도 입니다. 냉수
7번 물의 온도는 46.7도 입니다. 온수
8번 물의 온도는 85.0도 입니다. 끓는물
9번 물의 온도는 68.0도 입니다. 온수
10번 물의 온도는 0.5도 입니다. 냉수
냉수 입력 횟수 : 3
미온수 입력 횟수 : 2
온수 입력 횟수 : 3
끓는물 입력 횟수 : 2
계속하려면 아무 키나 누르십시오 . . .

```

### [G03] 점수 계산

학생 5명의 국어, 영어, 수학 점수를 각각 입력받아 저장한 후에, 다음 항목들을 계산하여 출력하라.

- 1) 각 과목별 총점과 평균 점수
- 2) 각 학생별 총점과 평균점수, 평균에 따른 등급

등급은 다음과 같은 기준으로 결정하라.

평균 90이상 : A

평균 80이상 ~ 90미만 : B

평균 70이상 ~ 80미만 : C

평균 60이상 ~ 70미만 : D

평균 60미만 : F

변수는 다음과 같이 사용하라.

```
jumsu=[]          # 5명의 3과목 점수, 총점, 평균, 등급을 저장하고 있는 중복 리스트
class_score=[]      # 3과목 총점과 평균을 저장하고 있는 중복 리스트
class_name=["국어","영어","수학"]    # 과목명 리스트
kor, eng, mat      # 3과목 점수 입력을 위한 변수
sum, average, grade # 총점, 평균, 등급 계산을 위한 변수
i, j               # 반복문을 위한 변수
```

The screenshot shows a command-line interface (cmd.exe) window with the following text output:

```
C:\Windows\system32\cmd.exe
1번 학생의 국어, 영어, 수학 점수는? 85 95 75
2번 학생의 국어, 영어, 수학 점수는? 90 80 70
3번 학생의 국어, 영어, 수학 점수는? 65 85 75
4번 학생의 국어, 영어, 수학 점수는? 60 70 80
5번 학생의 국어, 영어, 수학 점수는? 60 50 60

1) 각 과목별 총점과 평균 점수
국어 과목 총점은 360 평균은 72.0입니다.
영어 과목 총점은 380 평균은 76.0입니다.
수학 과목 총점은 360 평균은 72.0입니다.

2) 각 학생별 총점과 평균 점수
1번 학생 점수 : 총점 255, 평균 85.0, 등급 B
2번 학생 점수 : 총점 240, 평균 80.0, 등급 B
3번 학생 점수 : 총점 225, 평균 75.0, 등급 C
4번 학생 점수 : 총점 210, 평균 70.0, 등급 C
5번 학생 점수 : 총점 170, 평균 56.7, 등급 F
계속하려면 아무 키나 누르십시오 . . .
```

### [G04] 부동산 중개 수수료 계산기

부동산 거래종류(1:매매, 2:임대, 0:계산종료)와 거래금액을 입력받은 후에 이에 대한 중개 수수료를 계산하여 출력하라. 입력과 출력을 계속 반복하되 계산종료(0)를 입력하면 반복을 중단하고 지금까지의 수수료 총액을 출력하라.

중개 수수료 계산 방법은 아래 표를 참고하라.

거래종류	거래금액	수수료비율	상한금액
매매	5천만원 미만	0.6%	250,000
	5천만원 이상 ~ 2억원 미만	0.5%	800,000
	2억원 이상	0.4%	없음
임대	2천만원 미만	0.5%	70,000
	2천만원 이상 ~ 5천만원 미만	0.5%	200,000
	5천만원 이상 ~ 1억원 미만	0.4%	300,000
	1억원 이상	0.3%	없음

변수는 다음과 같이 사용하라.

```
kind          # 거래종류(1:매매, 2:임대, 0:계산종료)
money        # 거래금액
charge       # 중개 수수료
total_charge # 수수료 총액
```

```
C:\Windows\system32\cmd.exe
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 1
부동산 거래금액(원)를 입력하세요 : 48000000
이에 대한 중개 수수료는 250000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 1
부동산 거래금액(원)를 입력하세요 : 70000000
이에 대한 중개 수수료는 350000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 1
부동산 거래금액(원)를 입력하세요 : 250000000
이에 대한 중개 수수료는 1000000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 2
부동산 거래금액(원)를 입력하세요 : 190000000
이에 대한 중개 수수료는 70000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 2
부동산 거래금액(원)를 입력하세요 : 48000000
이에 대한 중개 수수료는 200000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 2
부동산 거래금액(원)를 입력하세요 : 70000000
이에 대한 중개 수수료는 280000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 2
부동산 거래금액(원)를 입력하세요 : 190000000
이에 대한 중개 수수료는 70000원입니다.
부동산 거래종류(1:매매, 2:임대, 0:계산종료)를 입력하세요 : 0
지금까지의 수수료 총액은 2220000원입니다.
계속하려면 아무 키나 누르십시오 . . .
```

## [G05] PC방 이용료 계산기

PC방 이용료를 계산하는 프로그램을 작성하라. 사용한 시간(시간, 분)을 입력받은 후 이에 따른 이용료를 화면에 출력하는 것을 반복한다. 시간과 분이 모두 0으로 입력되면 계산을 마치고 지금까지의 이용료 총금액을 출력하라.

단, 이용료는 매 30분 당 1,000원씩으로 계산하며, 다음과 같이 시간에 따라 할인혜택을 적용한다.

- 1) 2시간 이상 3시간 미만 이용자는 비용의 5%를 할인받는다.
- 2) 3시간 이상 5시간 미만 이용자는 비용의 10%를 할인받는다.
- 3) 5시간 이상 이용자는 비용의 20%를 할인받는다.

예) 이용시간이 4시간 20분이면 정상금액 9,000원에서 10% 할인을 받아 이용료는 8,100원이 된다.

변수는 다음과 같이 사용한다.

```
hour, minute      # 이용한 시간, 분  
charge           # 이용료  
total_charge     # 이용료 총액
```

The screenshot shows a Windows command prompt window titled 'cmd.exe' with the path 'C:\Windows\system32\cmd.exe'. The window displays a series of calculations for PC room usage fees. It asks for hours and minutes, calculates the charge based on a 30-minute interval rate of 1,000 won, applies discounts for longer durations, and finally prints the total accumulated charge. The last line of output is a prompt asking if the user wants to continue.

```
C:\Windows\system32\cmd.exe  
사용한 시간을 시간과 분으로 입력하세요 : 1 20  
고객님의 이용료는 3000원입니다.  
사용한 시간을 시간과 분으로 입력하세요 : 2 40  
고객님의 이용료는 5700원입니다.  
사용한 시간을 시간과 분으로 입력하세요 : 4 20  
고객님의 이용료는 8100원입니다.  
사용한 시간을 시간과 분으로 입력하세요 : 6 0  
고객님의 이용료는 9600원입니다.  
사용한 시간을 시간과 분으로 입력하세요 : 3 30  
고객님의 이용료는 6300원입니다.  
사용한 시간을 시간과 분으로 입력하세요 : 1 50  
고객님의 이용료는 4000원입니다.  
사용한 시간을 시간과 분으로 입력하세요 : 0 50  
고객님의 이용료는 2000원입니다.  
사용한 시간을 시간과 분으로 입력하세요 : 3 40  
고객님의 이용료는 7200원입니다.  
사용한 시간을 시간과 분으로 입력하세요 : 2 10  
고객님의 이용료는 4750원입니다.  
사용한 시간을 시간과 분으로 입력하세요 : 2 59  
고객님의 이용료는 5700원입니다.  
사용한 시간을 시간과 분으로 입력하세요 : 0 0  
  
지금까지의 이용료 총금액은 56350원입니다.  
계속하려면 아무 키나 누르십시오 . . . ■
```

## [G06] 쇼핑몰 매출 계산기

어떤 가게에서 세 종류의 제품(가죽장갑 1만원, 털장갑 6천원, 비닐장갑 3천원)을 판매하고 있다. 손님들이 들어오면 이 제품들에 대해 각각 몇 개를 구입할 것인지를 입력받아서 판매금액을 계산하여 출력하라. 구입 개수를 모두 0으로 입력하게 되면 판매가 종료되도록 하며, 지금까지 판매한 제품의 종류별 개수와 총 매출 금액을 화면에 출력하라.

변수는 다음과 같이 사용한다.

```
order = [0,0,0]          # 입력받는 구매 제품 개수 리스트 [가죽장갑, 털장갑, 비닐장갑]
sale                   # 계산한 판매금액
total_sale             # 총 매출액 총액
total_order = [0,0,0]    # 총 판매 개수 리스트 [가죽장갑, 털장갑, 비닐장갑]
price = [1000, 6000, 3000]# 제품별 가격
i                      # 반복문을 위한 변수
```

```
C:\Windows\system32\cmd.exe
세 종류의 제품이 있습니다.
(1. 가죽장갑 1만원, 2. 털장갑 6천원, 3. 비닐장갑 3천원)

1번 제품은 몇개를 구입하실래요? 1
2번 제품은 몇개를 구입하실래요? 1
3번 제품은 몇개를 구입하실래요? 1
판매금액은 19000원입니다.

1번 제품은 몇개를 구입하실래요? 2
2번 제품은 몇개를 구입하실래요? 3
3번 제품은 몇개를 구입하실래요? 4
판매금액은 50000원입니다.

1번 제품은 몇개를 구입하실래요? 0
2번 제품은 몇개를 구입하실래요? 20
3번 제품은 몇개를 구입하실래요? 1
판매금액은 123000원입니다.

1번 제품은 몇개를 구입하실래요? 3
2번 제품은 몇개를 구입하실래요? 5
3번 제품은 몇개를 구입하실래요? 5
판매금액은 75000원입니다.

1번 제품은 몇개를 구입하실래요? 0
2번 제품은 몇개를 구입하실래요? 0
3번 제품은 몇개를 구입하실래요? 0
지금까지의 총 매출금액은 267000원입니다.
계속하려면 아무 키나 누르십시오 . . .
```

## [G07] 놀이공원 매표소

놀이공원 매표소 프로그램을 제작하라. 프로그램 시작하면 몇 팀이 방문하였는지 입력받아 팀의 수만큼 다음과 같은 순서로 진행하도록 한다.

1) 팀별 인원 구성을 입력받는다. 인원구성은 (초등학생, 청소년, 일반인, 경로대상) 4종류별 인원수를 숫자로 입력받는다.

2) 팀별 정상요금을 계산한다. 1인당 요금은 다음과 같다.

- 초등학생 : 5000원, - 청소년 : 10000원, - 일반인 : 15000원, - 경로대상 : 3000원

3) 팀 별로 할인카드 소지 여부를 확인한다. 할인카드 종류별 할인율은 다음과 같다.

- 카드없음 : 할인 없음, - 일반등급 카드 : 10% 할인, - VIP 등급 카드 : 20% 할인

4) 최종 계산된 팀별 입장료를 출력한다.

모든 팀에 대한 처리가 완료되면 다음 내용을 화면에 출력하라.

1) 총 방문자 수

2) 인원구성별 방문자 합계 (4종류)

3) 총 입장료

변수는 다음과 같이 사용하라.

```
team_count          # 방문한 팀수
charge = [5000, 10000, 15000, 3000]      # 연령별 수
count = []           # 입력받은 연령별 인원 수
v_count = []         # 연령별 방문자 합계
total_count = 0      # 총 방문자 수
sum               # 팀별 계산한 요금
total_sum = 0        # 총 요금
membership = 0       # 할인카드 종류(카드없음:0, 일반등급 카드 : 1, VIP 등급 카드 : 2)
i, j              # 반복문을 위한 변수
```

```
C:\Windows\system32\cmd.exe
오늘 방문한 팀 수를 입력하세요. 4
1번팀 인원수(초등학생, 청소년, 일반, 경로대상)를 입력하세요. 2 1 2 0
1번팀 할인카드 종류(카드없음:0, 일반등급 : 1, VIP 등급 : 2)를 입력하세요. 2
1번팀 입장료는 40000원입니다.
2번팀 인원수(초등학생, 청소년, 일반, 경로대상)를 입력하세요. 5 0 1 0
2번팀 할인카드 종류(카드없음:0, 일반등급 : 1, VIP 등급 : 2)를 입력하세요. 0
2번팀 입장료는 40000원입니다.
3번팀 인원수(초등학생, 청소년, 일반, 경로대상)를 입력하세요. 4 0 2 2
3번팀 할인카드 종류(카드없음:0, 일반등급 : 1, VIP 등급 : 2)를 입력하세요. 1
3번팀 입장료는 50400원입니다.
4번팀 인원수(초등학생, 청소년, 일반, 경로대상)를 입력하세요. 0 5 1 0
4번팀 할인카드 종류(카드없음:0, 일반등급 : 1, VIP 등급 : 2)를 입력하세요. 0
4번팀 입장료는 65000원입니다.

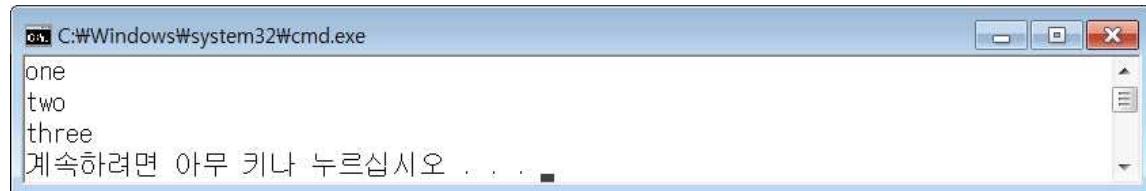
오늘 총 방문자 수는 25명입니다.
초등학생 수는 11명입니다.
청소년 수는 6명입니다.
일반인 수는 6명입니다.
경로대상 수는 2명입니다.

총 입장료는 195400원입니다.
계속하려면 아무 키나 누르십시오 . . .
```

## [Step H] 파이썬에서 함수 사용하기

대부분의 프로그램 언어와 같이 파이썬은 다양한 기능의 함수를 제공하고 있다. 프로그램 어느 곳에서나 사용할 수 있는 함수들도 있고, 특정 형식의 변수에 대해서 사용되는 함수들도 있다. 이번 단계에서는 특정 형식의 변수에서 자주 사용되는 파이썬 함수들을 연습해보자. 함수를 사용하는 방법은 변수 뒤에 점(.)을 쓰고 함수의 이름을 넣어주면 된다. 예를 들어 문자열에서 공백을 기준으로 단어들을 나누어 리스트로 만드는 `split()`함수는 다음과 같이 사용한다.

```
str1 = "one two three"
list1 = str1.split()
for item in list1 : print(item)
```



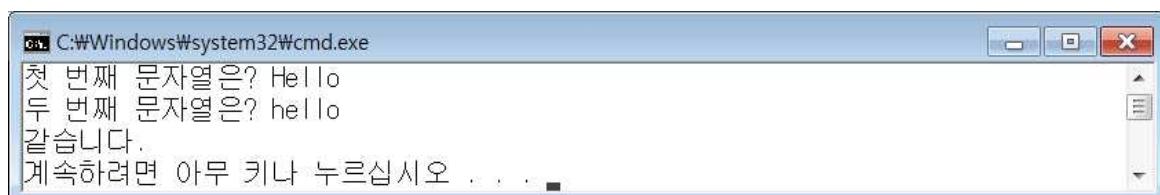
먼저 문자열 형식의 변수에서 사용되는 주요 함수들을 살펴보도록 하자.

함수	설명
<code>capitalize()</code>	문자열의 첫 문자만 대문자로 변경된 문자열을 리턴한다.
<code>count(sub)</code>	문자열 내에 <code>sub</code> 문자열의 포함 개수를 리턴한다.
<code>find(sub)</code>	문자열 내에 <code>sub</code> 문자열이 나타나는 위치를 리턴한다. 없으면 -1 리턴
<code>index(sub)</code>	문자열 내에 <code>sub</code> 문자열이 나타나는 위치를 리턴한다. 없으면 에러 발생
<code>isalnum()</code>	문자열이 알파벳과 숫자로만 되어있으면 <code>True</code> 를 리턴한다.
<code>isalpha()</code>	문자열이 알파벳으로만 되어있으면 <code>True</code> 를 리턴한다.
<code>isdigit()</code>	문자열이 숫자로만 되어있으면 <code>True</code> 를 리턴한다.
<code>islower()</code>	문자열이 모두 소문자로만 되어있으면 <code>True</code> 를 리턴한다.
<code>isupper()</code>	문자열이 모두 대문자로만 되어있으면 <code>True</code> 를 리턴한다.
<code>lstrip()</code>	문자열의 왼쪽부분에 나타나는 공백문자를 제거한 문자열을

	리턴한다.
<b>rstrip()</b>	문자열의 오른쪽부분에 나타나는 공백문자를 제거한 문자열을 리턴한다.
<b>strip()</b>	문자열의 좌우에 나타나는 공백문자를 제거한 문자열을 리턴한다.
<b>lower()</b>	문자열을 모두 소문자로 변경한 문자열을 리턴한다.
<b>upper()</b>	문자열을 모두 대문자로 변경한 문자열을 리턴한다.
<b>replace(old, new)</b>	문자열 내에서 old 문자열을 모두 new 문자열로 변경한 문자열을 리턴한다.
<b>split(sep)</b>	문자열을 sep문자열을 기준으로 나눈 다음 나눠지 문자열들로 이뤄진 리스트를 리턴한다. (sep이 생략되면 공백을 기준으로 함)
<b>startswith(sub)</b>	문자열이 sub문자열로 시작되면 True를 리턴한다.

그러면 간단한 예제를 통해 사용방법을 연습하기로 하자. 영문으로 된 두 개의 문자열을 입력받은 후, 대소문자를 구분하지 않고 이 문자열이 동일한지를 알아보자.

```
str1 = input("첫 번째 문자열은? ")
str2 = input("두 번째 문자열은? ")
if str1.lower() == str2.lower() :
    print("같습니다.")
else :
    print("같지 않습니다.")
```



이번에는 리스트 형식의 변수에서 사용할 수 있는 주요 함수들을 살펴보기로 하자.

함수	설명
<b>append(item)</b>	리스트에 마지막 요소로 item을 추가한다.
<b>count(item)</b>	리스트 내에 item이라는 요소가 포함된 개수를 리턴한다.
<b>insert(index, item)</b>	리스트의 index위치에 item이라는 요소를 추가한다.

<code>index(item)</code>	리스트 내에 <code>item</code> 이라는 요소가 나타나는 위치를 리턴한다. 없으면 에러 발생
<code>pop()</code>	리스트의 마지막 요소를 리턴하고 삭제한다.
<code>remove(item)</code>	리스트 내에 <code>item</code> 이라는 값을 가진 요소를 제거한다.
<code>reverse()</code>	리스트의 모든 요소를 역순으로 재배치한다.
<code>sort()</code>	리스트의 모든 요소를 정렬한다.

이번에는 외부의 모듈을 가져다가 필요한 함수를 사용하는 방법을 설명한다. 수학관련 계산을 처리해주는 함수들을 사용하려면 프로그램 소스의 처음에 다음과 같은 구문을 넣어주어야 한다.

```
from math import *
```

math 모듈을 가지고 오면 다음과 같은 함수들을 사용할 수 있다.

함수	설명
<code>cos(x), sin(x), tan(x)</code>	$x$ 에 대한 코사인, 사인, 탄젠트 값을 리턴한다.
<code>ceil(x)</code>	실수 $x$ 의 천정값, 즉 $x$ 보다 작지 않은 최소 정수를 리턴한다.
<code>fabs(x)</code>	실수 $x$ 의 절대값을 리턴한다.
<code>floor(x)</code>	실수 $x$ 의 바닥값, 즉 $x$ 보다 크지 않은 최대 정수를 리턴한다.
<code>pow(x, y)</code>	$x$ 의 $y$ 제곱값을 리턴한다. $x^{**}y$ 와 동일하다.
<code>sqrt(x)</code>	실수 $x$ 의 제곱근을 리턴한다.
<code>pi</code>	원주율 값을 가지고 있는 상수이다.

그러면 간단한 예제를 통해 사용방법을 연습하기로 하자. 1부터 20까지의 숫자들의 제곱근 값을 구하는 구문을 작성해보면 다음과 같다.

```
from math import *
for i in range(1,21) :
    print("%d의 제곱근은 %.3f입니다."%(i, sqrt(i)))
```

위 프로그램의 실행 화면은 다음과 같다.

```

1의 제곱근은 1.000입니다.
2의 제곱근은 1.414입니다.
3의 제곱근은 1.732입니다.
4의 제곱근은 2.000입니다.
5의 제곱근은 2.236입니다.
6의 제곱근은 2.449입니다.
7의 제곱근은 2.646입니다.
8의 제곱근은 2.828입니다.
9의 제곱근은 3.000입니다.
10의 제곱근은 3.162입니다.
11의 제곱근은 3.317입니다.
12의 제곱근은 3.464입니다.
13의 제곱근은 3.606입니다.
14의 제곱근은 3.742입니다.
15의 제곱근은 3.873입니다.
16의 제곱근은 4.000입니다.
17의 제곱근은 4.123입니다.
18의 제곱근은 4.243입니다.
19의 제곱근은 4.359입니다.
20의 제곱근은 4.472입니다.
계속하려면 아무 키나 누르십시오 . . .

```

이번에는 절대값을 구하는 함수인 `fabs()`를 사용해 보자. 10개의 숫자를 입력받은 후에 이 숫자들 중에 절대값이 가장 큰 숫자를 찾아서 출력하는 프로그램을 만들려면 어떻게 하면 될까? 먼저 10 개짜리 실수형 숫자의 배열을 만든 다음에 반복문을 10번 돌려가면서 숫자를 하나씩 입력받으면서 배열에 저장한다. 그런 다음에 [Step D]에서 연습한 최댓값 구하는 해결방법을 이용하되 숫자들을 비교하기 전에 절댓값으로 바꾼 값을 비교하도록 하면 된다.

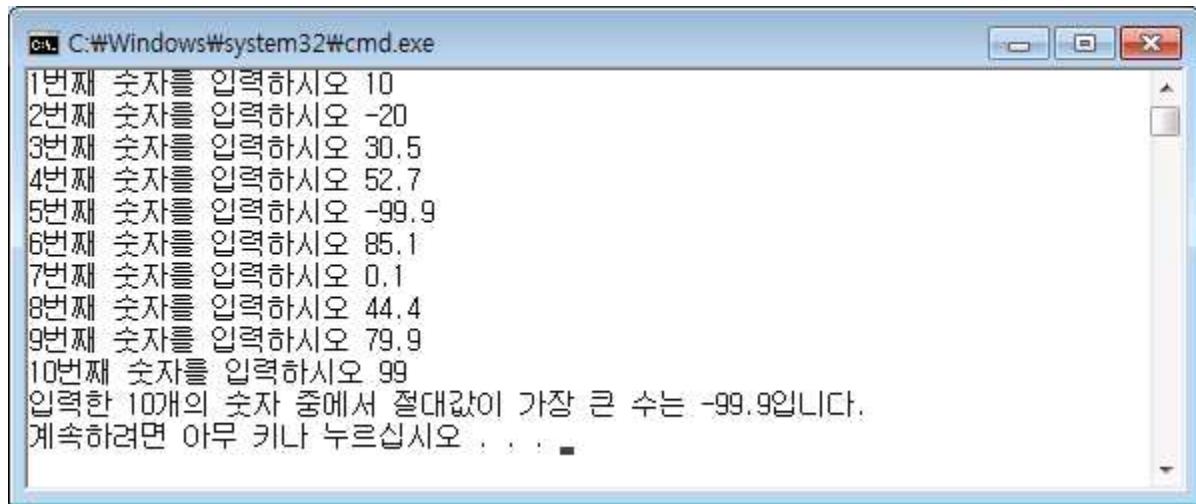
```

# -*- coding: utf-8 -*-
# 예제 ex_H.py
from math import *

number = []
for i in range(10) :
    newnum = eval(input("%d번째 숫자를 입력하시오 "%(i+1)))
    number.append(newnum)

max_index = 0          # 우선 첫번째 숫자를 가장 큰 수로 정한다.
for i in range(1,10) :   # 두 번째 숫자부터 열 번째 숫자까지 반복해가면서
    if fabs(number[i]) > fabs(number[max_index]) :      # 절대값을 서로 비교
        max_index = i          # max_index를 절대값이 큰 수의 인덱스로 변경.
print("10개의 숫자 중에서 절대값이 가장 큰 수는 %.1f입니다."%(number[max_index]))

```



다음은 임의의 숫자를 만들어 사용하는 랜덤 함수를 사용해보자. 랜덤 함수는 숫자를 정해주면 그 숫자만큼의 임의의 수를 자동으로 만들어 리턴하는 것이다. 이 함수들을 사용할 때에는 반드시 다음과 같이 모듈을 가져오도록 정의해야 한다.

```
from random import *
```

여기에서 사용되는 주요 함수는 `randint()`와 `random()`이다. 예를 들어 `randint(1,100)`을 호출하면 1에서 100사이의 숫자 100개 중에서 임의의 숫자를 리턴하게 된다. `random()` 함수는 0과 1사이의 임의의 실수값을 만들어 리턴한다.

그러면 간단한 예제를 통해 사용방법을 연습하기로 하자. 로또 복권은 1부터 45까지의 숫자 6개를 맞추는 것인데, 이 6개의 번호를 임의로 만들어보는 프로그램을 작성하려면 어떻게 할까?

그리고 1부터 45까지의 숫자를 임의로 만들기 위해서는 `randint(1,45)`를 사용하면 된다. 작성된 프로그램은 다음과 같다.

```
from random import *

for i in range(6) :
    print("%d번째 숫자는 : %d"%(i+1,randint(1,45)))
```

물론 위 프로그램에서는 1부터 45 사이의 임의의 숫자를 6개 만들어내기 때문에 아래 화면처럼 6 개의 숫자들 간에 겹치는 숫자가 나올 가능성이 있다. 절대로 겹치지 않게 만드는 방법은 실습문제를 통해 각자 해결해보도록 하라.

The image shows two separate windows, both titled "C:\Windows\system32\cmd.exe". Each window displays the output of a Python script. The script prints six lines of text, each consisting of a number followed by a colon and a space, then the word "숫자는", then a value. After these six lines, there is a message in Korean asking the user to press any key to continue. The two windows show slightly different values for the numbers.

```
1번째 숫자는 : 14  
2번째 숫자는 : 44  
3번째 숫자는 : 29  
4번째 숫자는 : 11  
5번째 숫자는 : 8  
6번째 숫자는 : 33  
계속하려면 아무 키나 누르십시오 . . .  
1번째 숫자는 : 15  
2번째 숫자는 : 14  
3번째 숫자는 : 14  
4번째 숫자는 : 43  
5번째 숫자는 : 24  
6번째 숫자는 : 12  
계속하려면 아무 키나 누르십시오 . . .
```

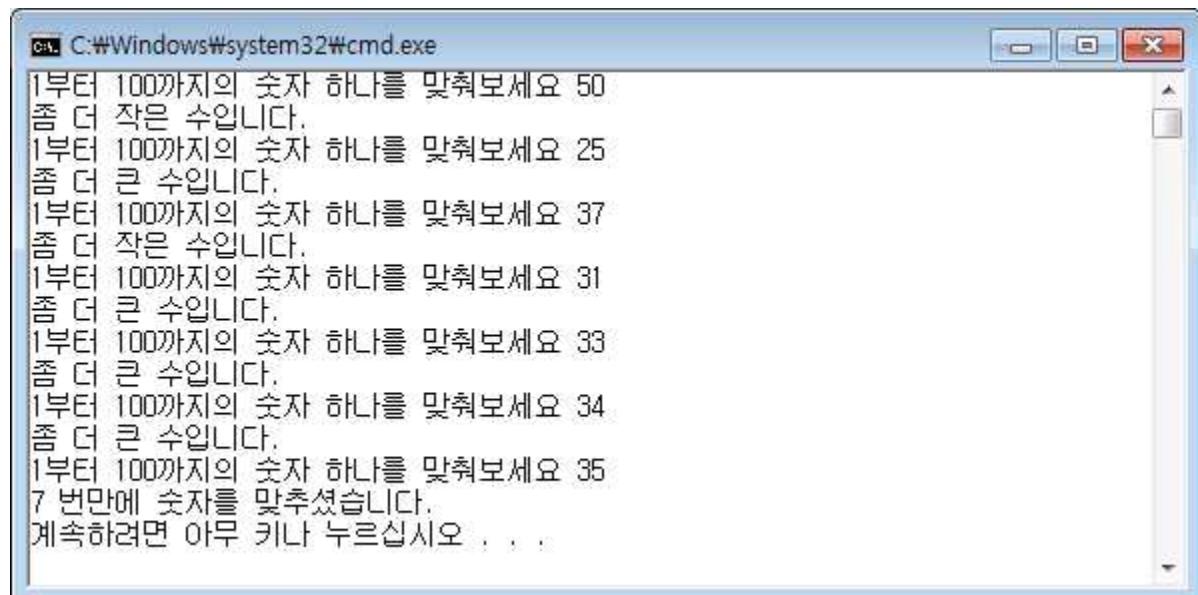
## ○ 실습 문제

### [H01] 숫자 알아 맞추기

1부터 100 사이의 임의의 숫자를 만든 후에 사용자로 하여금 이 숫자를 맞추도록 하라. 사용자가 제시한 숫자보다 큰 수인지 작은 수인지를 알려주면서 몇 번 만에 맞추었는지 출력하라.

변수는 다음과 같이 사용하라.

answer	# 컴퓨터가 만들어 낸 1부터 100사이의 임의의 숫자
number_try	# 사용자가 제시한 숫자
count	# 사용자가 맞추려고 시도한 횟수

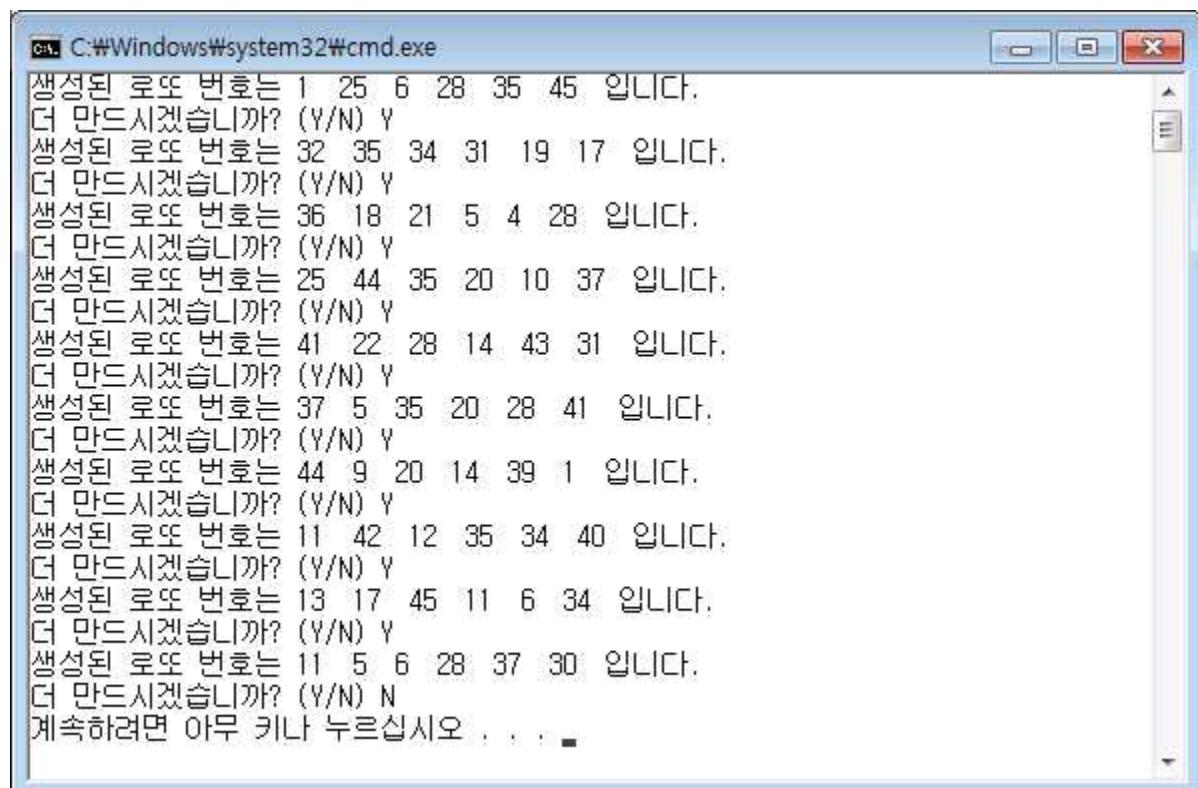


## [H02] 로또 번호 만들기

1부터 45사이의 임의의 숫자를 6개 만들어 로또 번호를 완성하라. 단, 6개의 번호 중에 중복되는 번호가 없도록 해야 한다. 출력한 후에 사용자에게 "더 만드시겠습니까? (Y/N)"를 물어보고 N을 입력할 때까지 계속해서 반복하라.

변수는 다음과 같이 사용하라.

```
lotto = []          # 컴퓨터가 만들어 낸 로또 번호 리스트
count            # 현재 만들어지고 있는 로또 번호의 순서(0, 1, 2, 3, 4, 5)
onemore          # 반복여부를 입력하는 문자 (Y/N)
i                # 반복문을 위한 변수
```

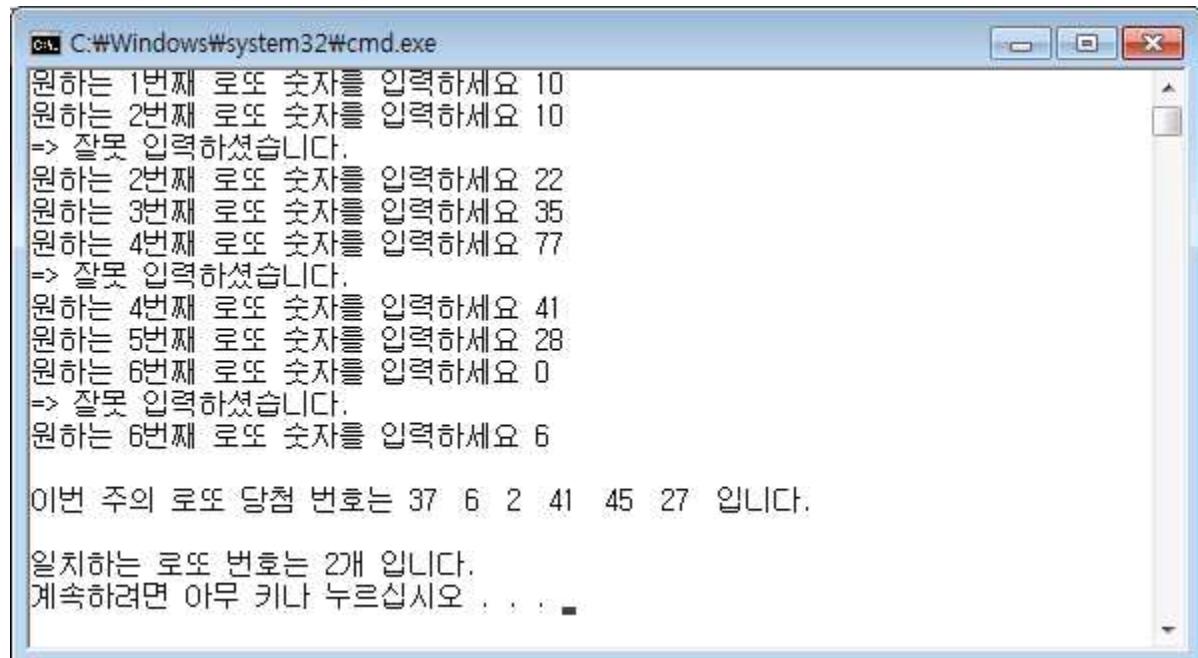


### [H03] 로또 번호 당첨 확인하기

사용자에게 1부터 45사이의 임의의 숫자를 6개 입력받은 후에, 프로그램에서 만든 임의의 로또 번호와 대조하여 몇 개의 숫자가 일치하는지 출력하라. 단, 사용자가 입력한 6개의 번호 중에 중복되는 번호가 없도록 입력받아야 하며, 프로그램에서 만든 임의의 로또 번호에서도 중복되는 번호가 없도록 해야 한다.

변수는 다음과 같이 사용하라.

```
lotto_com = []          # 컴퓨터가 만들어 낸 로또 번호 리스트
lotto_user =[]          # 사용자가 입력한 로또 번호 리스트
i                      # 반복문을 위한 변수
count                  # 현재 만들어지고 있는 로또 번호의 순서(0, 1, 2, 3, 4, 5)
match_count            # 일치하는 로또 번호의 갯수 (0~6)
```



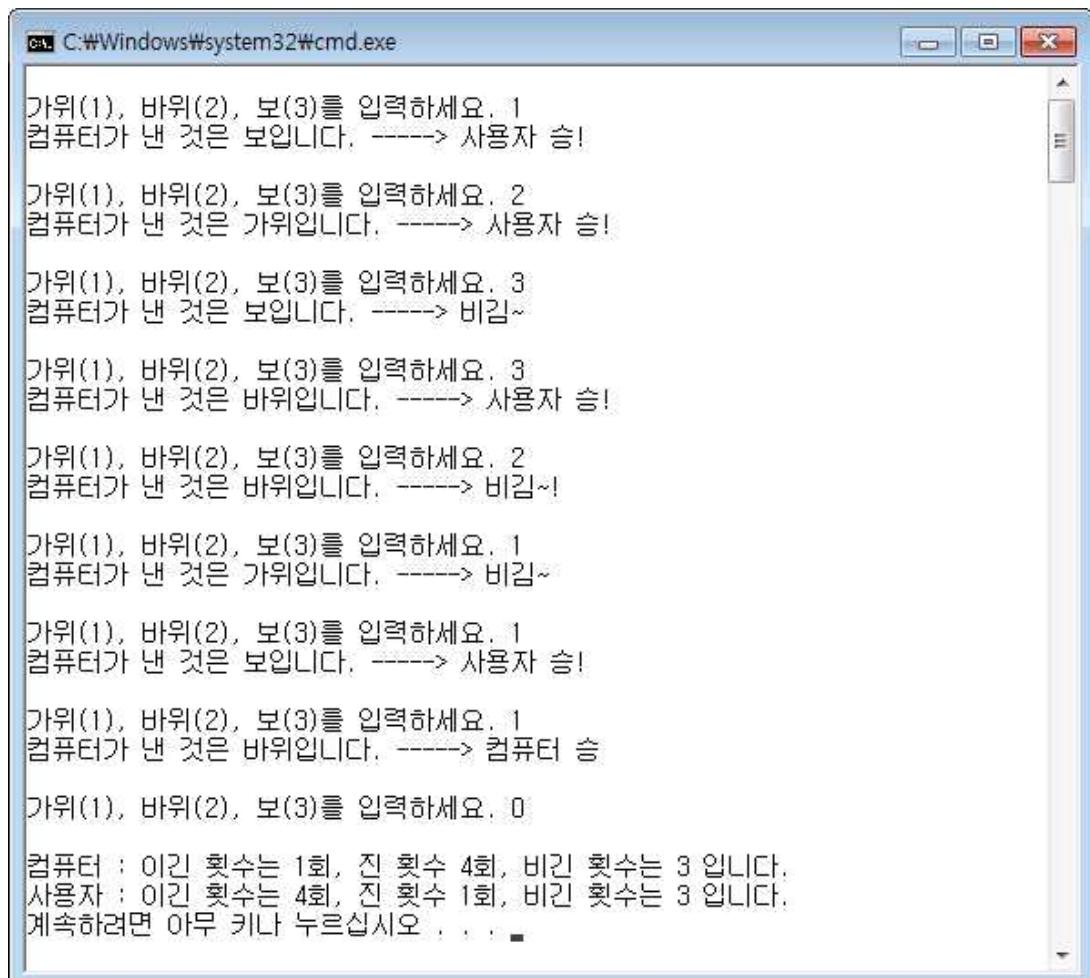
## [H04] 가위바위보 게임하기

다음과 같이 사용자와 컴퓨터가 가위바위보를 하는 프로그램을 만들어라.

- 1) 랜덤으로 가위(1), 바위(2), 보(3) 셋 중에 하나를 만든다.
- 2) 사용자에게 가위(1), 바위(2), 보(3) 중에 하나를 숫자로 입력받는다.
- 3) 사용자가 입력한 것과 컴퓨터가 만들어 낸 것을 비교하여  
    컴퓨터가 이기면, "컴퓨터 승!"  
    사용자가 이기면, "사용자 승!"  
    비기면, "비김~" 으로 출력한다.
- 4) 사용자가 0을 입력할 때까지 위 1)부터 3)을 계속 반복하다가, 끝나면 그동안 컴퓨터와 사용자가 이긴 횟수와 진 횟수, 비긴 횟수를 출력하라.

변수는 다음과 같이 사용하라.

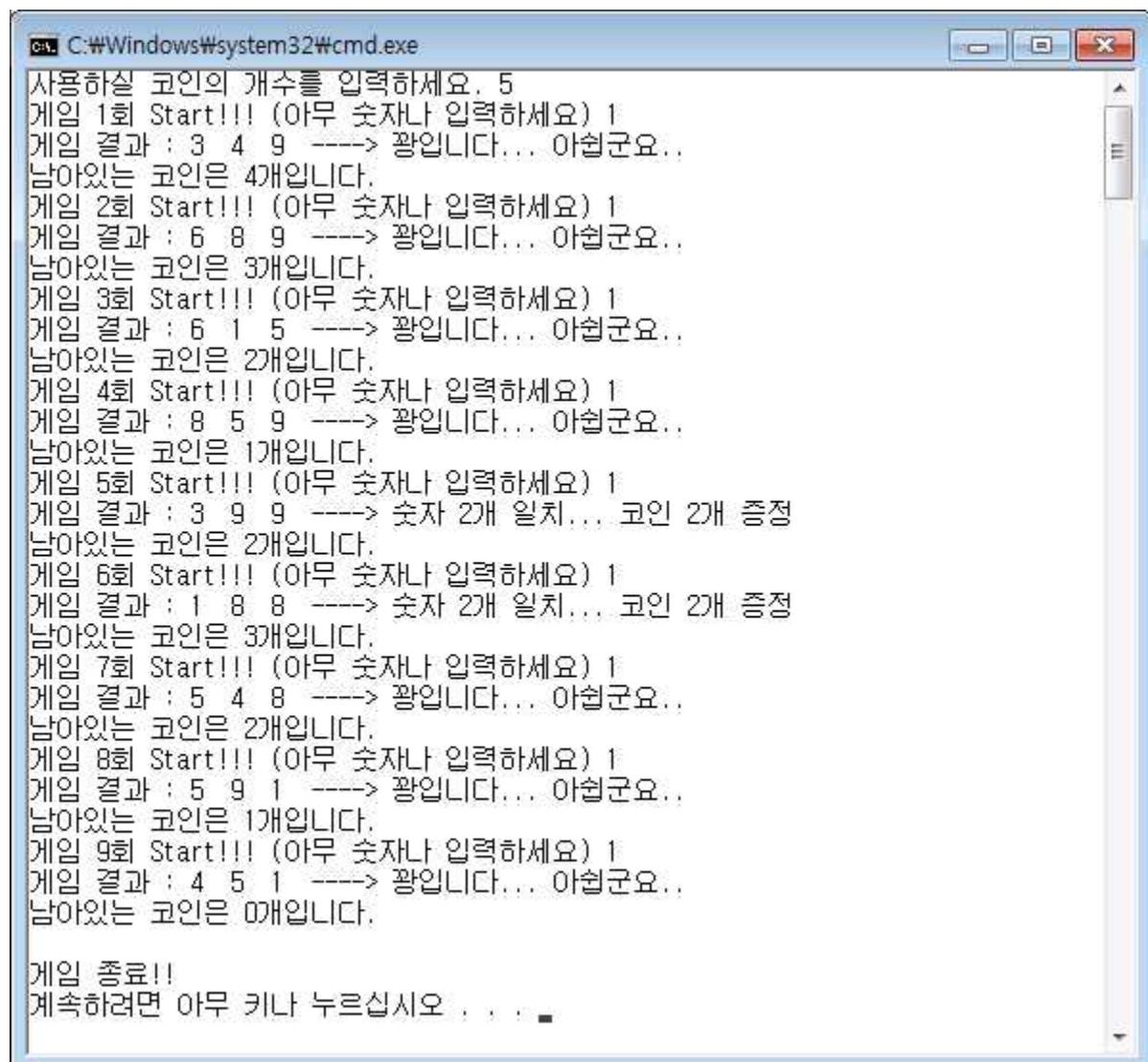
```
com_finger      # 컴퓨터가 낸 가위(1), 바위(2), 보(3)
my_finger      # 사용자가 낸 가위(1), 바위(2), 보(3)
score = []      # 결과 횟수 리스트 [[컴퓨터의 승, 무, 패], [사용자의 승, 무, 패]]
i, j           # 반복문을 위한 변수
```



## [H05] 슬롯머신 만들기

간단한 슬롯 머신을 만들어라. 사용자에게 코인의 개수를 숫자로 입력받은 다음에, 1부터 9까지의 숫자 3개를 랜덤으로 만들어 3개의 숫자가 같으면 상금으로 코인 4개, 2개의 숫자가 같으면 코인 2개를 더해준다. 한 번 할 때마다 코인을 1개 씩 소모하게 되고, 사용자의 코인이 모두 소모될 때까지 게임을 반복시켜라. 변수는 다음과 같이 사용하라.

```
coins           # 사용자의 코인 수. (최초에 입력받음)
number = []     # 랜덤하게 만들어진 슬롯 머신의 숫자 3개의 리스트
dummy          # 게임스타트를 위한 의미 없는 입력 숫자
i              # 반복문을 위한 변수
```

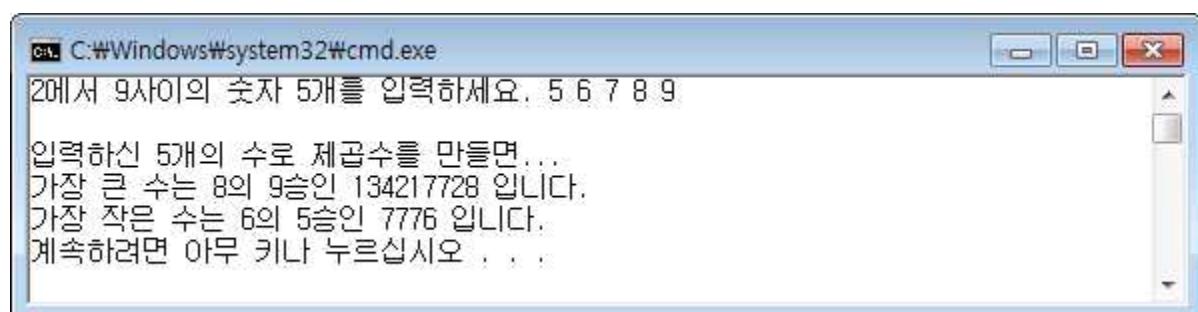
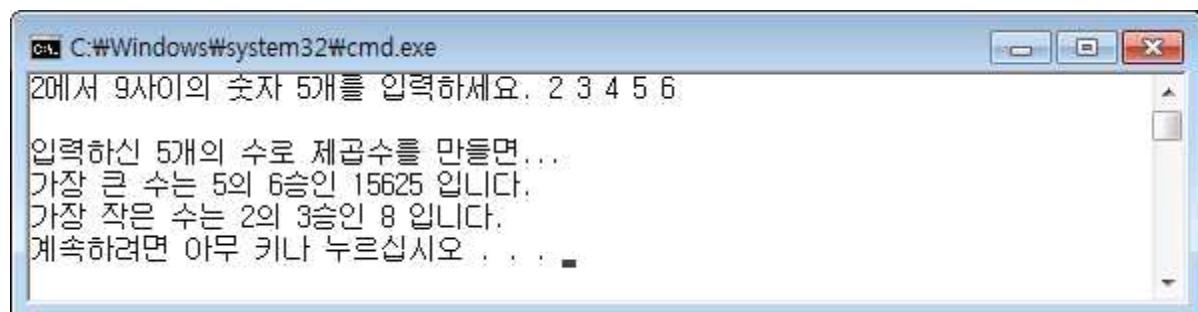


## [H06] 5개 숫자의 제곱수 조합 구하기

2에서 9사이의 숫자 5개를 입력받아 배열에 저장한 후, 이 5개의 숫자들 중 임의의 2개의 숫자  $a$ 와  $b$ 를 선택하여  $a^b$ 의 값을 계산하여 이 중에서 가장 큰 수와 가장 작은 수를 만들 수 있는 경우를 찾아내라. 계산할 때에는 함수 `pow()`를 사용하라. 예를 들어 2, 3, 4, 5, 6을 입력한 경우에는 가장 작은 수는  $2^3$ 이고, 가장 큰 수는  $5^6$ 이 된다.

변수는 다음과 같이 사용하라.

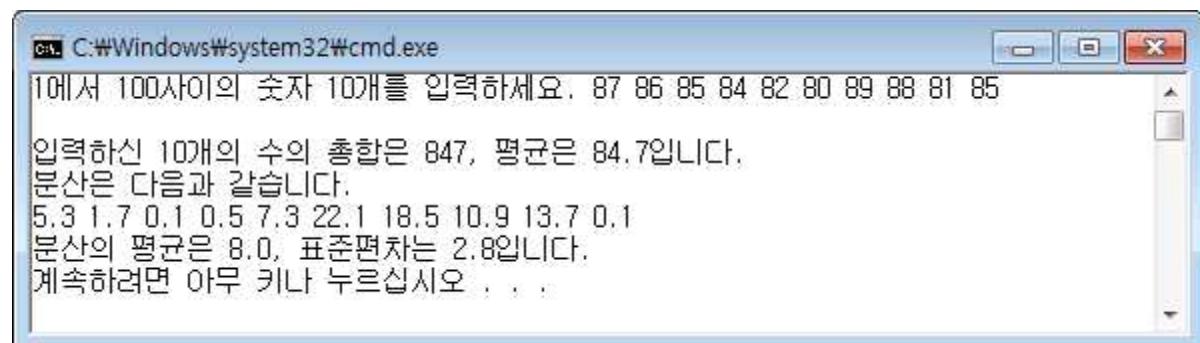
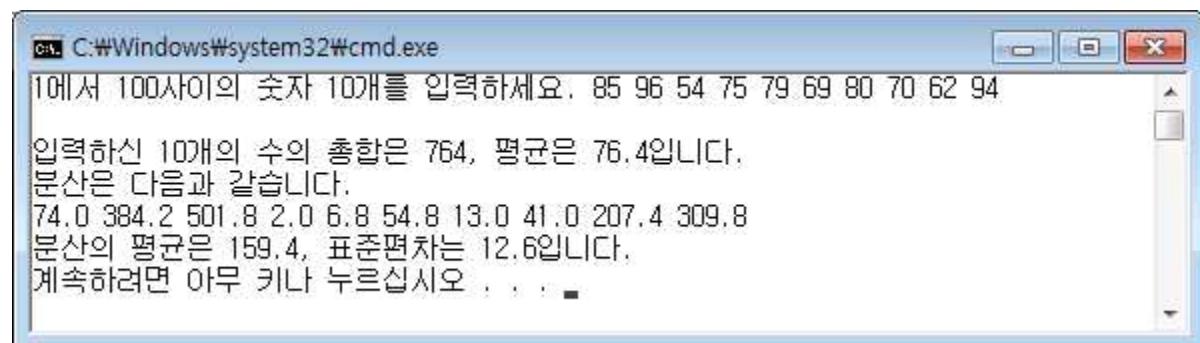
```
number = []          # 입력받은 5개의 숫자 리스트
pow_value = []       # 임의의 두 수 a, b로 만들 수 있는  $a^b$ 의 값들, 5 * 5 리스트
max, min           # 최댓값과 최솟값
max_a, max_b       # 최댓값을 만들어 내는 경우의 a와 b의 값
min_a, min_b       # 최솟값을 만들어 내는 경우의 a와 b의 값
i, j               # 반복문을 위한 변수
```



### [H07] 표준편차 구하기

1에서 100사이의 숫자 10개를 입력받아 배열에 저장한 후, 반복문을 사용하여 이 숫자들의 총합계와 평균은 구한 후, 표준편차를 구하라. 표준편차를 계산할 때에는 함수 `sqrt()`와 `pow()`를 사용하라. 표준편차는 각 숫자와 평균과의 차이 값을 제곱한 수 10개를 구한 후, 이들 평균값의 제곱근을 계산한 것이다. 변수는 다음과 같이 사용하라.

```
number = []          # 입력받은 10개의 숫자 리스트
total               # 총 합계
average             # 평균
bunsan = []         # 10개 숫자의 분산 값 (각각 (숫자-평균)의 제곱) 리스트
total_bunsan        # 분산의 총합
average_bunsan      # 분산의 평균
std_dev             # 표준편차 (분산 평균의 제곱근)
i                   # 반복문을 위한 변수
```



## [H08] 애너그램(Anagram) 판별하기

문자열 s1, s2 2개를 입력받아 이 두 문자열이 anagram인지 판별하라. anagram이란 두개의 문자열 속의 알파벳이 동일한 갯수만큼 포함된 것을 말한다. 이때, 알파벳을 제외한 문자들 즉 공백, 기호(, . ' " \_ - 등)는 제외해야 하며, 대소문자는 따로 구분하지 않는다.

참고할 함수 : 문자열의 lower(), isalpha(), 리스트의 sort()

변수는 다음과 같이 사용하라.

```
s1, s2          # 입력받은 문자열 2개  
list1 = [], list2 = []    # 문자열을 리스트로 변경할 때 사용하는 리스트 변수  
ch, i, j        # 반복문을 위한 변수
```

```
C:\Windows\system32\cmd.exe  
첫번째 문자열은? Mother-in-law  
두번째 문자열은? Woman Hitler  
애너그램입니다.  
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\Windows\system32\cmd.exe  
첫번째 문자열은? Women Hitler  
두번째 문자열은? Mother-in-law  
애너그램이 아닙니다.  
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\Windows\system32\cmd.exe  
첫번째 문자열은? Election results  
두번째 문자열은? Lies-let's recount  
애너그램입니다.  
계속하려면 아무 키나 누르십시오 . . .
```

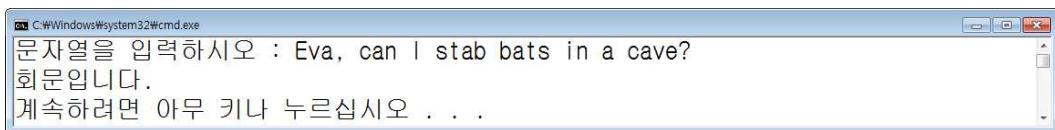
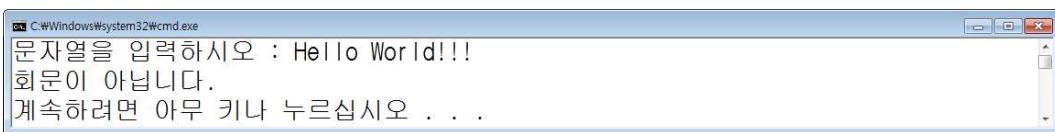
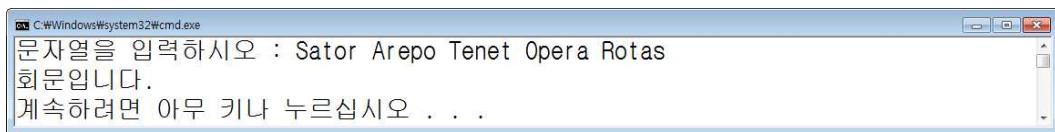
### [H09] 회문(Palindrome) 판별하기

회문(Palindrome)인지 검사하는 프로그램을 작성하라. 회문이란 거꾸로 읽어도 동일한 알파벳 순서가 되는 문장이다. 단, 알파벳 이외의 문자는 제외하고, 대소문자는 가리지 않는다.

참고할 함수 : 문자열의 lower(), isalpha(), 리스트의 reverse()

변수는 다음과 같이 사용하라.

```
str                      # 입력받은 문자열
list1 = [], list2 = []    # 문자열을 리스트로 변경할 때 사용하는 리스트 변수
ch, i, j                 # 반복문을 위한 변수
```



## [H10] 문자열 분석하기

문자열을 입력받아서 이 문자열을 분석한 결과를 출력하는 함수를 만들라.

분석결과의 항목은 다음과 같다.

- 1)문자열 총 개수, 2)공백(빈칸, 탭) 수,
- 3)대문자 수, 4)소문자 수, 5)숫자 갯수, 6)그외나머지문자들의 수

참고할 함수 : 문자열의 isupper(), islower(), isalpha(), isdigit()

변수는 다음과 같이 사용하라.

str	#입력받은 문자열
count_all	#총 문자 개수
count_upper	#대문자 개수
count_lower	#소문자 개수
count_space	#공백 개수
count_number	#숫자 개수
count_etc	#기타 문자 개수
ch	#반복문을 위해 사용하는 문자변수 (for ch in str:)

```
C:\Windows\system32\cmd.exe
문자열을 입력하시오 : Hello World!! I'm 25 years old...
총 문자 개수 : 33
대문자 개수 : 3
소문자 개수 : 17
공백 개수 : 5
숫자 개수 : 2
기타 문자 개수 : 6
계속하려면 아무 키나 누르십시오 . . .
```

## [Step H'] 딕셔너리 사용하기

이번 단계는 파이썬에서 제공하는 딕셔너리(Dictionary) 형식의 변수를 사용하는 방법을 활용하기로 한다. 딕셔너리 즉, 사전이라는 말에서 알 수 있듯이 어떤 데이터가 단어와 뜻처럼 키(key)와 값(value)의 쌍으로 묶여져 있는 경우에 이런 데이터를 다루기 위한 형식이다. 딕셔너리는 리스트처럼 순서를 매겨서 추가하거나 가져오는 방식이 아니라 특정한 키에 해당되는 값을 찾는 방식으로 사용한다.

예를 들어 단어를 키로 하고, 뜻을 값으로 갖는 딕셔너리는 다음과 같이 정할 수 있다.

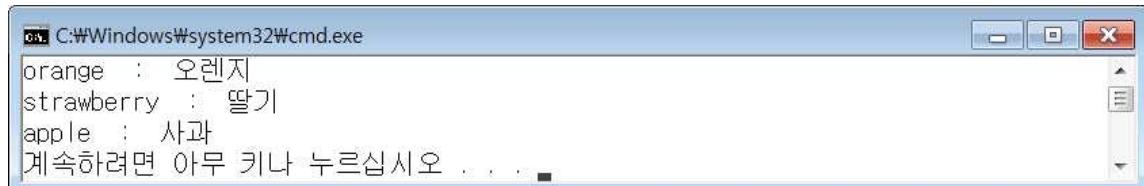
```
EngDic = {"apple": "사과", "orange": "오렌지", "strawberry": "딸기"}
```

"apple" 을 키로 가지는 값을 출력하기 위해서는 다음과 같이 하면 된다. 인덱스를 통해 리스트를 접근하듯이 딕셔너리는 키를 통해 접근하는 것이다.

```
print(EngDic["apple"])
```

이렇게 만든 딕셔너리에서 모든 키를 끄집어내서 키와 값을 화면에 출력하려면 다음과 같이 하면 된다.

```
EngDic = {"apple": "사과", "orange": "오렌지", "strawberry": "딸기"}  
for item in EngDic.keys():  
    print(item, " : ", EngDic[item])
```



화면에서 볼 수 있듯이 딕셔너리는 내용이 순차적으로 들어가지 않기 때문에 `keys()` 함수를 사용해서 키 리스트를 만들 때에 입력한 순서와 상관없다는 것을 염두에 두어야 한다.

다음과 같이 딕셔너리에서 값을 리스트 형식으로 만들 수도 있다.

```
EngDic = {"check" : ["검사하다", "조사하다"], "assign" : ["할당하다", "배정하다"], "involved" :  
    ["관계된", "연루된"], "engage" : ["종사하다", "관여하다"], "execute" : ["수행하다", "실행하다"]}
```

이렇게 만들어진 딕셔너리의 모든 키와 값을 화면에 출력하려면 다음과 같이 값 리스트에서 반복문을 사용해야 한다.

```
EngDic = {"check" : ["검사하다", "조사하다"], "assign" : ["할당하다", "배정하다"], "involved" : ["관계된", "연루된"], "engage" : ["종사하다", "관여하다"], "execute" : ["수행하다", "실행하다"]}
for item in EngDic.keys():
    print(item, " : ")
    for val in EngDic[item] :
        print(val, end=" ")
    print()
```

```
involved :
관계된 연루된
execute :
수행하다 실행하다
engage :
종사하다 관여하다
check :
검사하다 조사하다
assign :
할당하다 배정하다
계속하려면 아무 키나 누르십시오 . . .
```

딕셔너리에 대해서 사용할 수 있는 주요 함수들은 다음과 같다.

함수	설명
clear()	딕셔너리 내의 모든 요소를 제거한다.
get(k, d)	딕셔너리 내에 키 k가 들어 있으면 그에 대한 값을 리턴하고, 없는 경우에는 d를 리턴한다. (d가 생략되면 None 리턴)
has_key(k)	딕셔너리 내에 키 k가 들어 있으면 True를 리턴한다.
items()	딕셔너리 내의 모든 키와 값을 (키, 값)의 리스트로 만들어 리턴한다.
keys()	딕셔너리 내의 모든 키를 리스트로 만들어 리턴한다.
values()	딕셔너리 내의 모든 값을 리스트로 만들어 리턴한다.
pop(k,d)	딕셔너리 내에 키 k가 들어 있으면 그에 대한 값을 리턴하고 삭제한다. 없는 경우에는 d를 리턴한다.
popitem()	딕셔너리 내에서 임의의 (키, 값) 튜플을 리턴하고 제거한다.

## 영역 3 : 함수 만들어 사용하기

영역 3에서는 프로그램을 제작할 때에 필수 능력이라고 할 수 있는 함수 제작을 연습한다. 프로그램에서 처리해야 하는 작업들을 내용과 성격에 따라 서브 함수로 제작하여 역할을 분담하는 기술은 반드시 터득해야 하는 기술이다. 영역 3의 연습을 마치고 나면 영역 1과 영역 2에서 습득한 프로그래밍의 제어구조 기술을 메인 함수와 서브 함수 모두에 골고루 적용할 수 있게 될 것이다.

함수는 별도로 작업해야 하는 프로그램 구문들을 따로 정해놓은 것이며, 함수를 호출할 때에 전달해야 하는 데이터가 있으면 이를 파라미터로 전달할 수 있다. 그리고 함수의 동작이 마친 후에 결과와 값을 돌려받아야 하는 경우에는 리턴 값을 명시해서 돌려줄 수 있다. 물론 함수에 전달할 데이터가 없는 경우에는 파라미터 없이 함수를 제작하면 되며, 리턴할 결과 값이 없는 경우에는 리턴 값이 없다고 명시하면 된다.

영역 3의 실습 문제에서는 사용해야 하는 변수를 알려주지 않는다. 여러분이 문제를 충분히 읽고 분석한 후, 적절하게 필요한 변수들을 선언하여 사용하기 바란다.

영역 3는 다음과 같이 3개의 단계로 구성된다.

Step I : 파라미터 또는 리턴값만 있는 함수 만들기

Step J : 파라미터와 리턴 값이 모두 있는 함수 만들기

Step K : 파일 사용하기

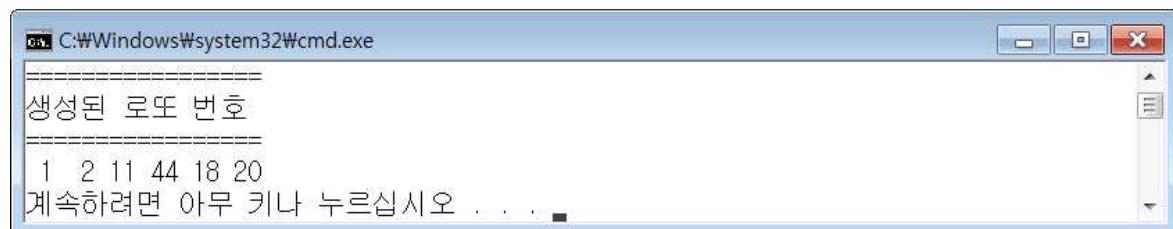
## [Step I] 파라미터 또는 리턴값만 있는 함수 만들기

이번 단계에서는 가장 기본적인 함수들을 만드는 연습을 하려고 한다. 함수를 호출하게 되면 뭔가 정해진 작업을 수행하기만 하면 함수의 역할이 끝나는 경우이다. 이런 경우에는 함수의 리턴 값이 없다고 말한다. 예를 들어 다음 예제와 같이 화면에 구분선을 출력하는 경우가 여러 번 있다고 가정한다면, 이 작업이 필요할 때마다 해당 구문을 일일이 프로그램에 넣기보다는 함수로 만들어 호출하게 하면 프로그램을 읽거나 만들 때에 좀 더 편리하다. 소스에서 보는 바와 같이 메인 함수 위에 함수의 내용을 구현하였다.

```
# -*- coding: utf-8 -*-
# 예제 ex_I1.py
from random import *

def print_header() :          # 함수 시작
    print ("=*17)
    print ("생성된 로또 번호")
    print ("=*17)

# 메인함수 시작
lotto = set()                 # 로또 번호를 보관하기 위한 집합 생성
while len(lotto) < 6:           # 만들어진 번호의 갯수가 6개가 되면 종료
    lotto.add(randint(1,45))
print_header()                  # 함수 호출
for item in lotto:
    print("%2d"%item, end=" ")
print()
```



이번에는 파라미터가 있고 리턴 값이 없는 함수를 만들어보자. 위의 예제에서 `print_header()` 함수는 테두리를 등호('=')로 출력하고 있는데, 이 함수에 테두리의 모양문자를 지정하는 값을 파라미터로 주어서 출력 모양을 바꿔보자. 그러면 함수의 내용은 다음과 같이 바뀌어야 한다.

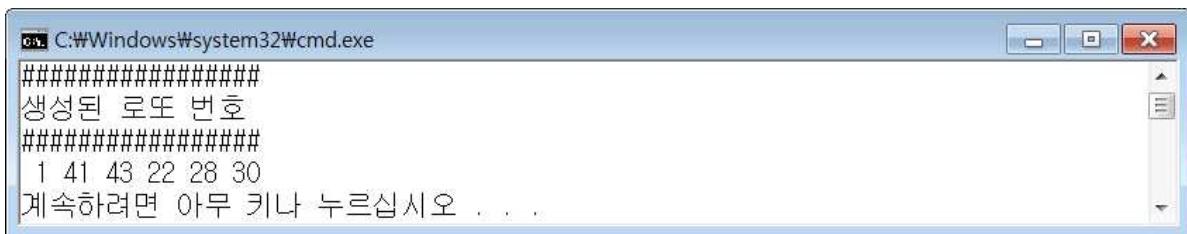
```
def print_header(ch) :          # 변경된 함수 원형 정의, 파라미터로 ch가 추가되었다.
```

이렇게 바뀐 함수를 사용하도록 변경된 프로그램은 다음과 같다.

```
# coding: euc-kr
# 예제 ex_I2.py
from random import *

def print_header(ch) :      # 함수 시작
    print(ch*17)
    print("생성된 로또 번호")
    print(ch*17)

# 메인함수 시작
lotto = set()              # 로또 번호를 보관하기 위한 집합 생성
while len(lotto) < 6:        # 만들어진 번호의 갯수가 6개가 되면 종료
    lotto.add(randint(1,45))
print_header("#")           # 함수 호출시 파라미터로 테두리에 사용할 문자를 지정하였다.
for item in lotto:
    print("%2d"%item, end=" ")
print()
```



이번에는 파라미터 없이 리턴 값만 돌려주는 함수를 만들어보자. 예를 들어 입력받아야 하는 데이터가 바르게 입력되도록 점검해야 하는 일이 반복된다면 이 작업을 함수로 제작해서 편리하게 불러 오면 된다. 성적을 처리하는 프로그램에서 점수를 입력받을 때에 반드시 0에서 100사이의 정수를 입력받아야 하는데 이 범위에 맞는 수인지 검사하는 함수를 만들어보자. 이 함수를 사용하여 국어, 영어, 수학 점수를 입력받아 총점과 평균을 계산하는 프로그램은 다음과 같다.

```
# -*- coding: utf-8 -*-
# 예제 ex_I3.py
def GetScore() :          # 함수 시작
    while 1 :
        jumsu = int(input("점수를 입력하시오. (0~100) "))
        if 0<=jumsu<=100 : return jumsu
        else: print "잘못된 범위의 수입니다.",
```

```
# 메인함수 시작
class_name = ["국어", "영어", "수학"]
class_score = []
sum = 0
for i in range(3):
    print class_name[i],
    score = GetScore()
    class_score.append(score)
    sum += score
average = sum / 3.0
print("3과목의 총점은 %d, 평균은 %.1f입니다."%(sum, average))
```



## ○ 실습 문제

### [I01] 메뉴판 보여주는 함수 만들기

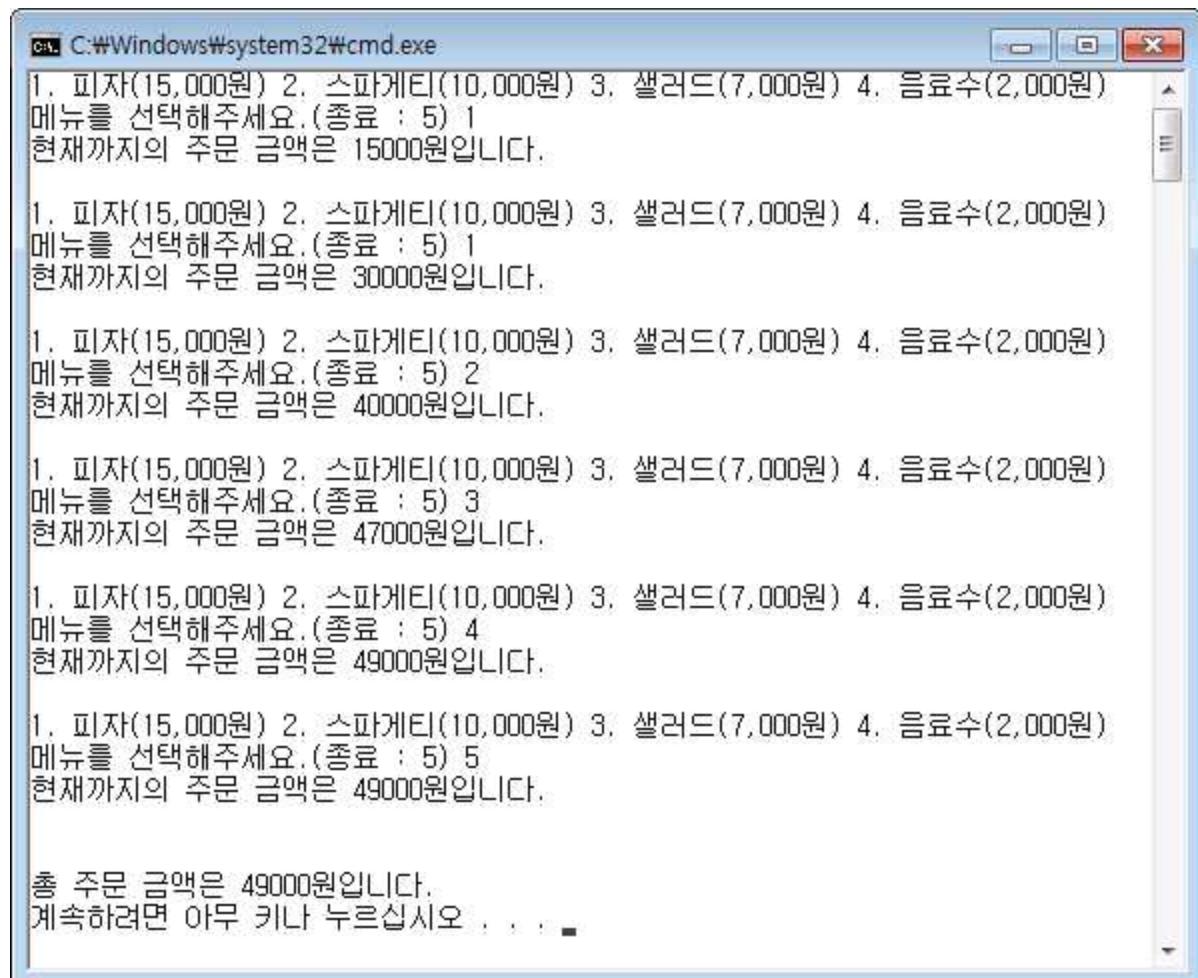
어떤 식당의 메뉴판을 보여주는 함수를 만들어라. 메인 함수에서 이 함수를 호출하여 메뉴판을 보여준 다음 메뉴번호를 입력받아 해당 메뉴의 가격을 합산하되 메뉴선택 종료를 의미하는 5값을 입력 받을 때까지 계속 반복하여 메뉴를 선택하게 하고, 선택종료 후 선택한 메뉴의 총 합계금액을 출력하라.

단, 사용할 메뉴는 다음과 같다.

1. 피자(15,000원)
2. 스파게티(10,000원)
3. 샐러드(7,000원)
4. 음료수(2,000원)
5. 종료

함수 선언부는 다음과 같다.

```
def ShowMenu() :
    파라미터) 없음
    리턴 값) 없음
    수행내용) 메뉴판 출력
```

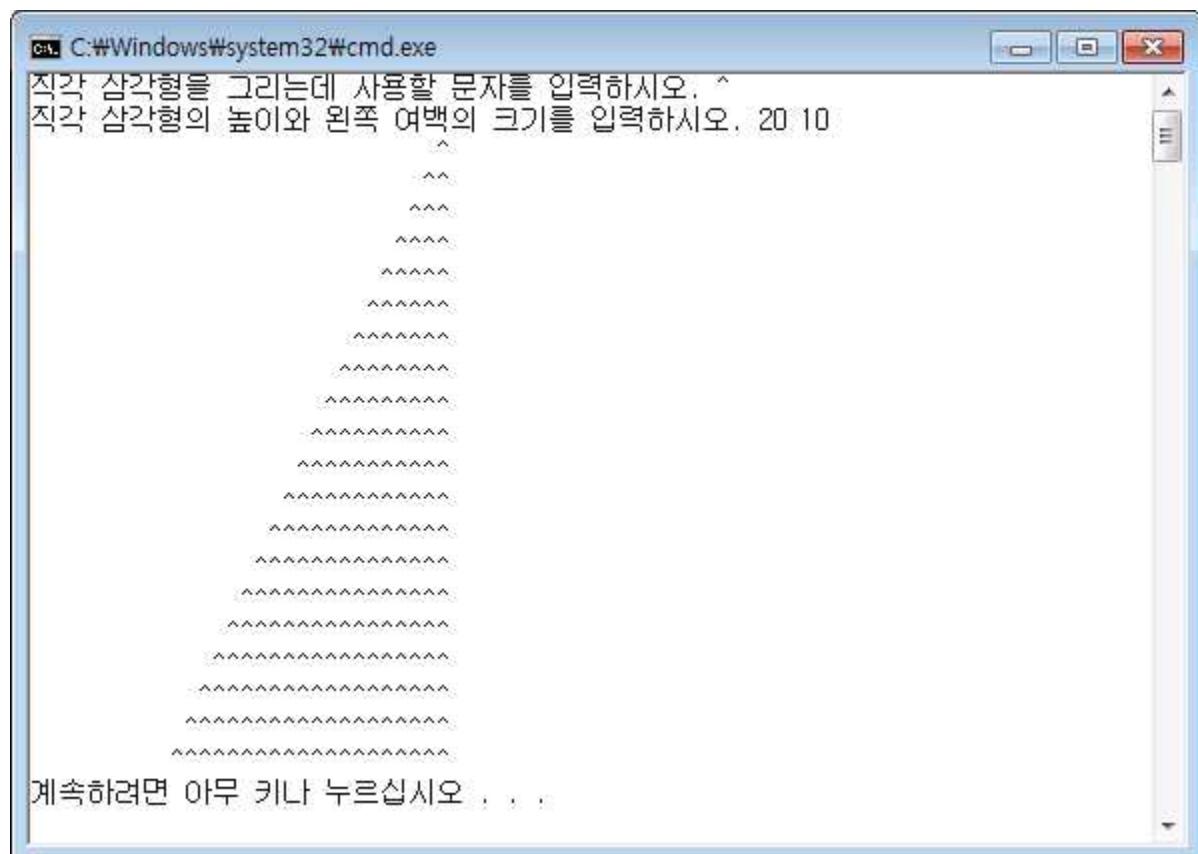


## [I02] 빈칸과 함께 특정 문자를 개수만큼 찍는 함수 만들기

파라미터로 문자 하나와 숫자 두 개를 넘겨주면 한 줄에 첫 번째 숫자만큼 빈칸을 출력한 후, 바로 이어서 두 번째 숫자만큼 넘겨받은 문자를 화면에 출력하는 함수를 만들어라. 그리고 사용자로부터 모양(문자 하나)과 높이, 여백을 입력받은 후, 이 함수를 사용해서 입력한 크기만큼의 여백과 높이를 갖는 우직각 삼각형을 입력한 문자모양으로 화면에 출력시켜라.

함수 선언부는 다음과 같다.

```
def PrintCharWithBlank(blanks, size, ch) :  
    파라미터) blank : 빈칸의 개수, size : 출력할 문자의 개수, ch : 출력할 문자  
    리턴 값) 없음  
    수행내용) blanks 숫자만큼 빈칸 출력, size 개수만큼 ch 문자 출력 후 줄바꿈
```



### [103] 비만 판정

10명의 신장(cm단위)과 체중(kg단위)를 입력받으면서 AskBiman() 함수를 통해 이들이 비만도를 출력하고 다음 기준에 따라 비만여부를 판정하여 출력하라.

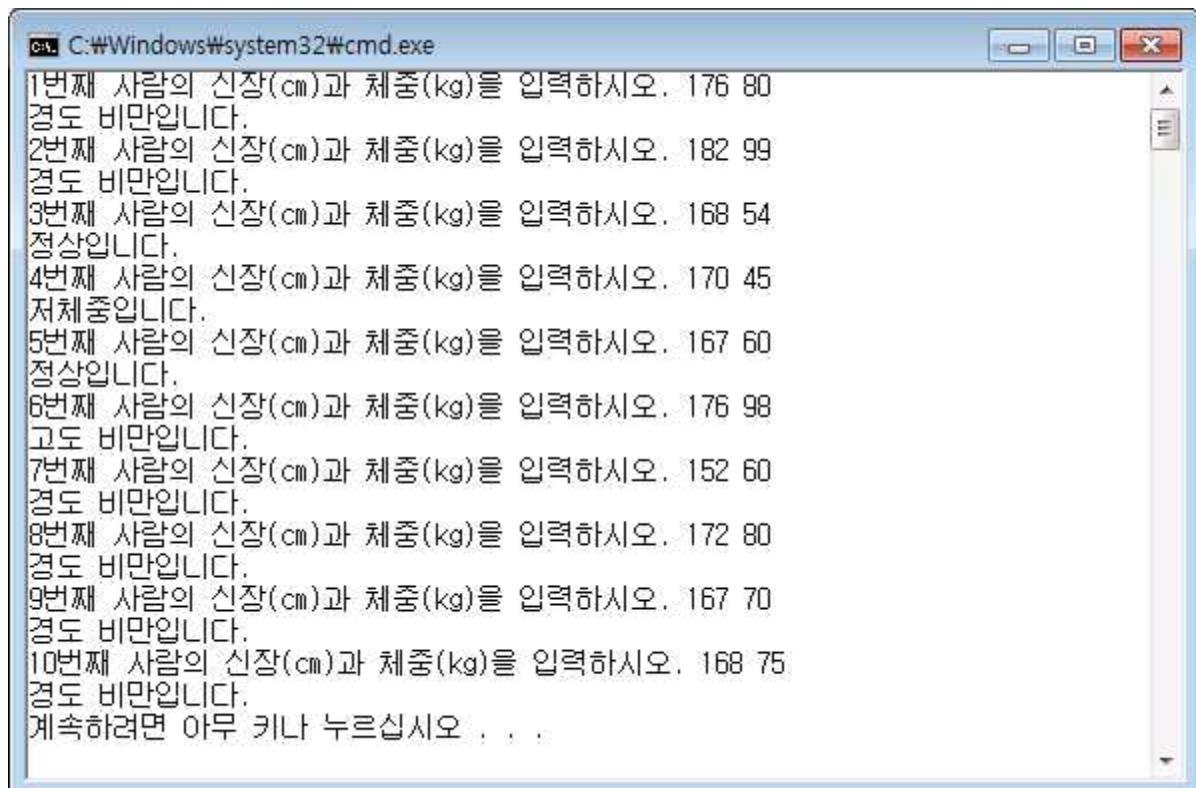
비만도 수치 = 체중(kg) / (신장(m)의 제곱) 으로 계산한다. 이 때, 신장은 미터 단위로 환산해야 함을 유의하라.

비만도 수치에 따른 비만도 판정

1. 18.5 미만 : 저체중
  2. 18.5 ~ 23 미만 : 정상체중
  3. 23~25미만 : 과체중
  4. 25~30미만 : 경도비만
  5. 30이상 : 고도비만

함수 선언부는 다음과 같다.

```
def AskBiman(height, weight) :  
    파라미터 height : 신장(cm), weight : 체중(kg)  
    리턴 값) 없음  
    수행내용) 비만도 계산 후 판정결과 출력
```



### [I04] 메뉴 번호 받아오는 함수 만들기

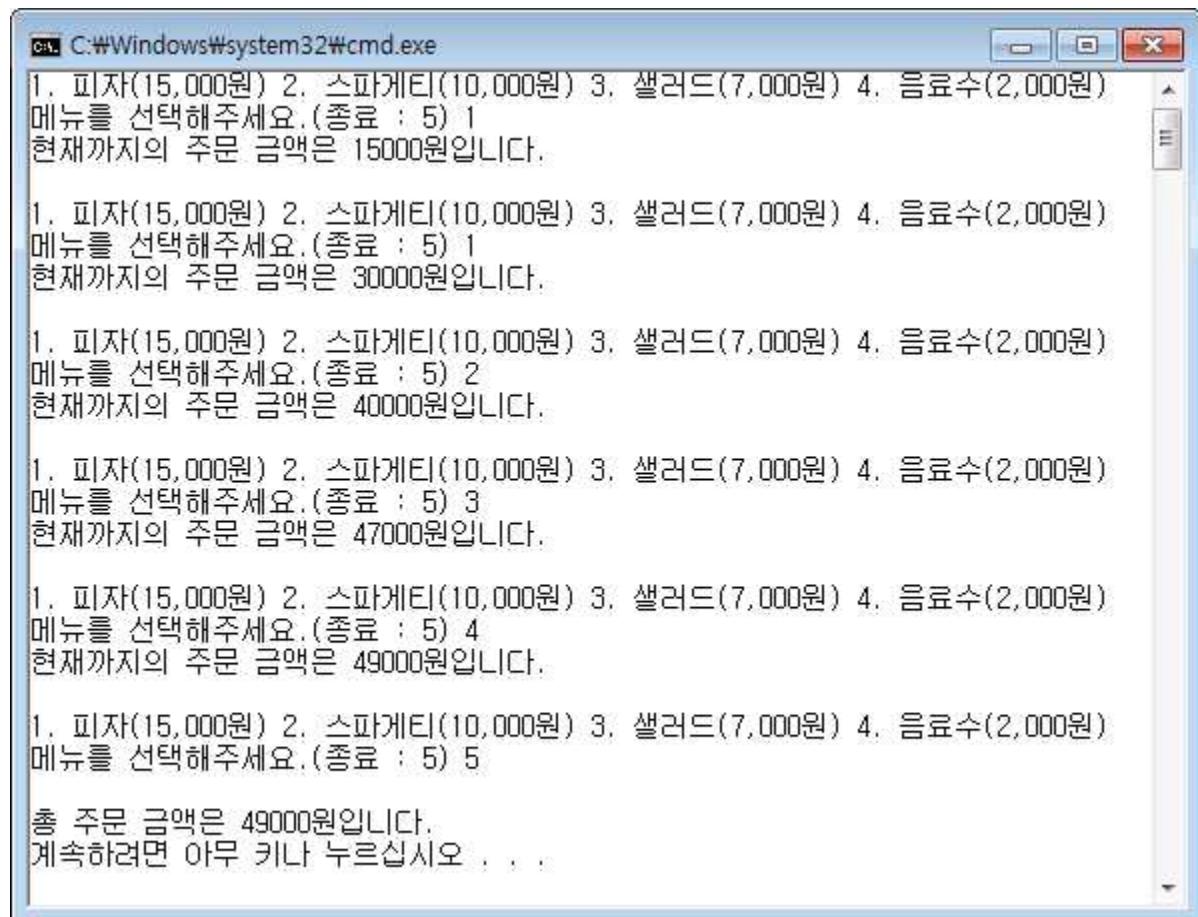
어떤 식당의 메뉴판을 보여준 후에 메뉴번호를 입력받아 그 가격을 리턴하는 함수를 만들어라. 메인 함수에서 이 함수를 호출하여 리턴 받은 가격을 합산하되 메뉴선택 종료를 의미하는 5를 리턴 받을 때까지 계속 반복하여 메뉴를 선택하게 하고, 선택종료 후 선택한 메뉴의 총 합계금액을 출력하라.

단, 사용할 메뉴는 다음과 같다.

1. 피자(15,000원), 2. 스파게티(10,000원), 3. 샐러드(7,000원), 4. 음료수(2,000원), 5. 종료

함수 선언부는 다음과 같다.

```
def SelectMenu() :  
    파라미터) 없음  
    리턴 값) 1~4를 선택하면 선택한 메뉴의 가격, 5를 선택하면 -1
```



### [I05] 최댓값 리턴하는 함수 만들기

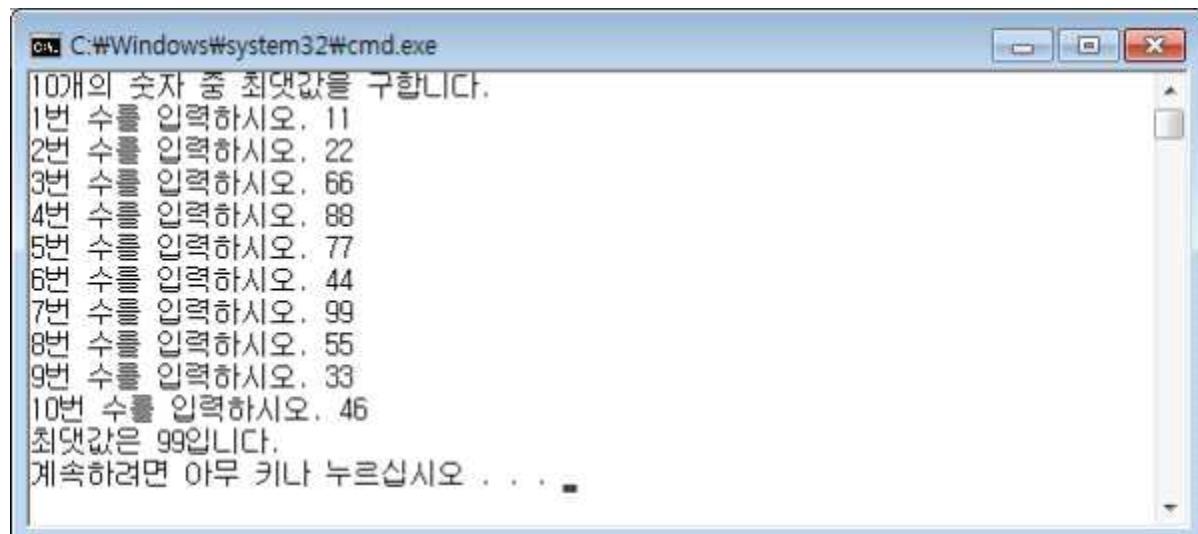
숫자 10개를 입력받아 이 중 최댓값을 찾아서 리턴하는 함수 MaxOfTen()을 만들고, 이 함수를 이용하여 10개의 숫자 중 최댓값을 출력하라.

함수 선언부는 다음과 같다.

```
def MaxOfTen():
```

파라미터) 없음

리턴 값) 최댓값



### [I06] 임의의 숫자를 만들어 구간을 리턴하는 함수 만들기

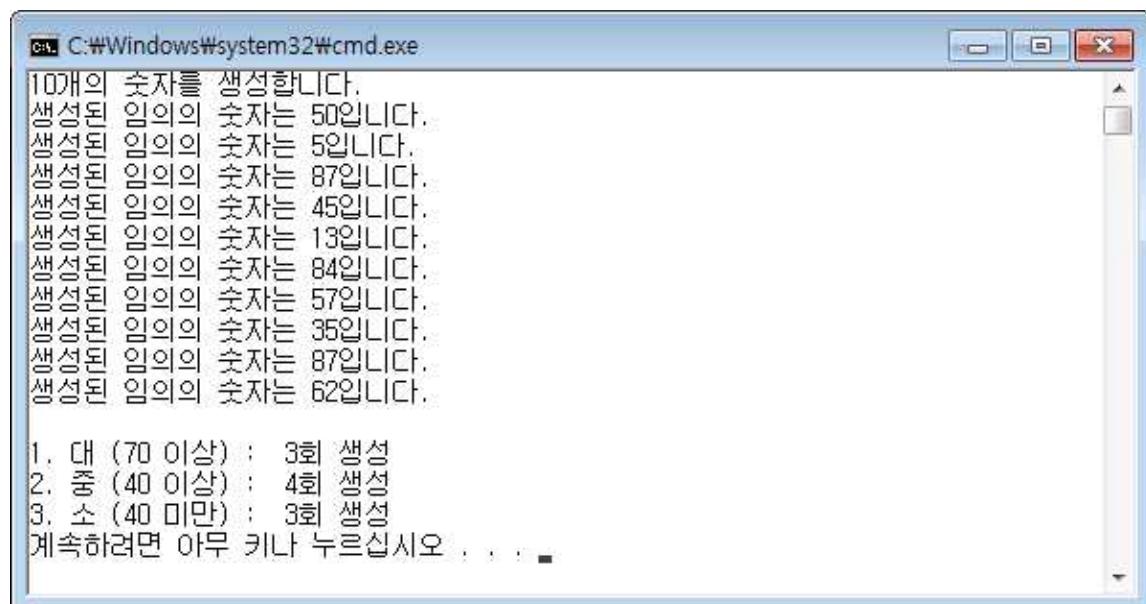
1부터 100사이의 임의의 숫자를 만들어서 대(70 이상), 중(40이상~70미만), 소(40미만) 셋 중에 하나를 판정하여 결과를 리턴하는 함수 GetRandom()을 만들어라. 그리고 이 함수를 이용해서 임의의 숫자 10개에 대해 대, 중, 소가 각각 몇 번씩 포함되어 있는지 개수를 출력하라.

함수 선언부는 다음과 같다.

```
def GetRandom():
```

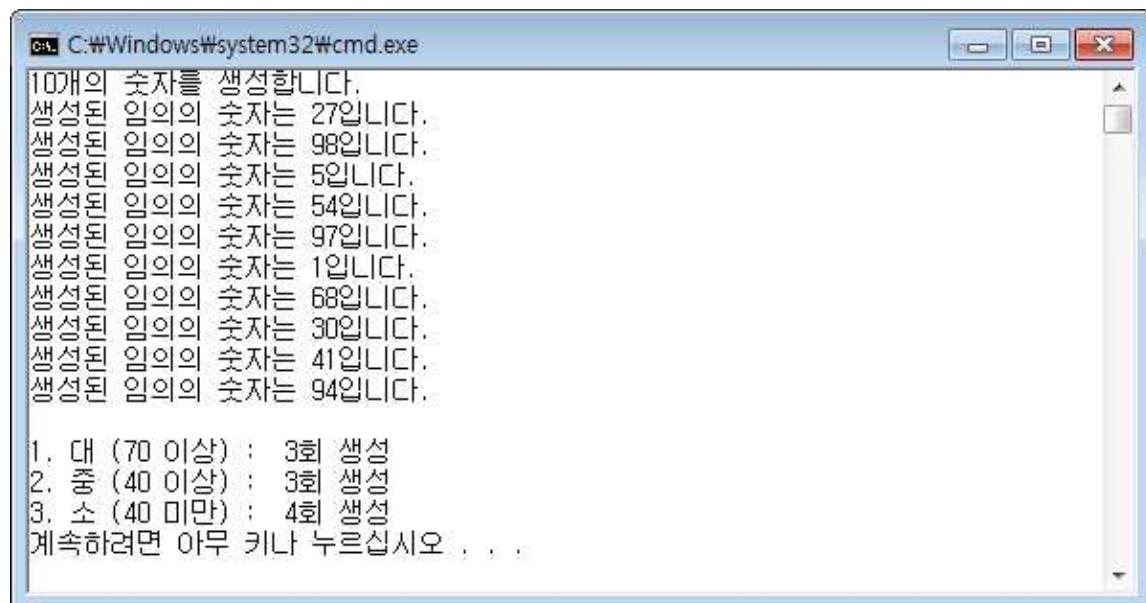
파라미터) 없음

리턴 값) 임의로 만들어낸 숫자가 속하는 구간번호 0.대(70 이상), 1.중(40이상~70미만), 2.소(40미만)



```
C:\Windows\system32\cmd.exe
10개의 숫자를 생성합니다.
생성된 임의의 숫자는 5입니다.
생성된 임의의 숫자는 5입니다.
생성된 임의의 숫자는 87입니다.
생성된 임의의 숫자는 45입니다.
생성된 임의의 숫자는 13입니다.
생성된 임의의 숫자는 84입니다.
생성된 임의의 숫자는 57입니다.
생성된 임의의 숫자는 35입니다.
생성된 임의의 숫자는 87입니다.
생성된 임의의 숫자는 62입니다.

1. 대 (70 이상) : 3회 생성
2. 중 (40 이상) : 4회 생성
3. 소 (40 미만) : 3회 생성
계속하려면 아무 키나 누르십시오 . . .
```



```
C:\Windows\system32\cmd.exe
10개의 숫자를 생성합니다.
생성된 임의의 숫자는 27입니다.
생성된 임의의 숫자는 98입니다.
생성된 임의의 숫자는 5입니다.
생성된 임의의 숫자는 54입니다.
생성된 임의의 숫자는 97입니다.
생성된 임의의 숫자는 1입니다.
생성된 임의의 숫자는 68입니다.
생성된 임의의 숫자는 30입니다.
생성된 임의의 숫자는 41입니다.
생성된 임의의 숫자는 94입니다.

1. 대 (70 이상) : 3회 생성
2. 중 (40 이상) : 3회 생성
3. 소 (40 미만) : 4회 생성
계속하려면 아무 키나 누르십시오 . . .
```

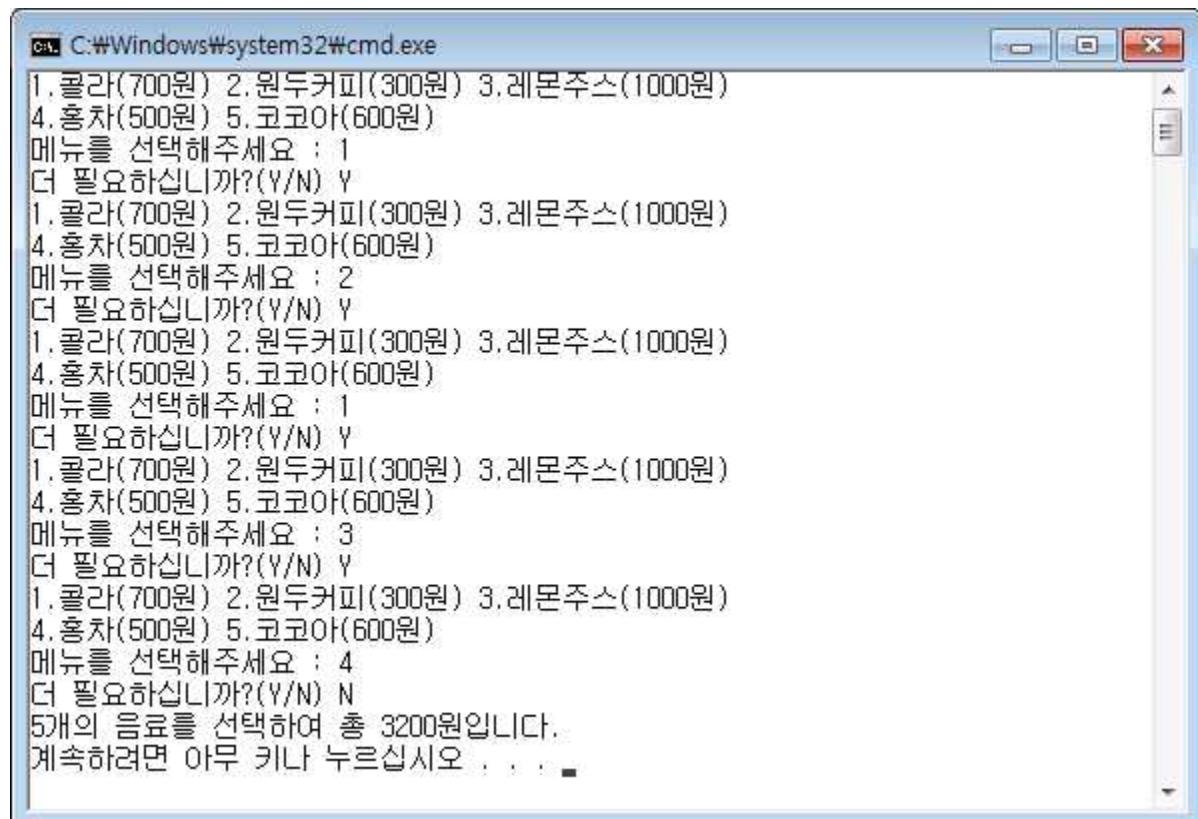
### [I07] 자판기에서 선택한 음료 가격을 리턴하는 함수 만들기

자판기의 메뉴를 보여주고 선택하게 하여 선택된 음료의 가격을 리턴하는 함수 SelectCan()을 만들어라.  
그리고 이 함수를 이용해서 자판기에서 음료를 반복해서 선택하게 하여 총 음료의 개수와 가격을 출력하라.  
자판기의 음료 종류와 가격은 다음과 같다.

1.콜라(700원) 2.원두커피(300원) 3.레몬주스(1000원) 4.홍차(500원) 5.코코아(600원)

함수 선언부는 다음과 같다.

```
def SelectCan():
    파라미터) 없음
    리턴 값) 선택한 음료의 가격
```



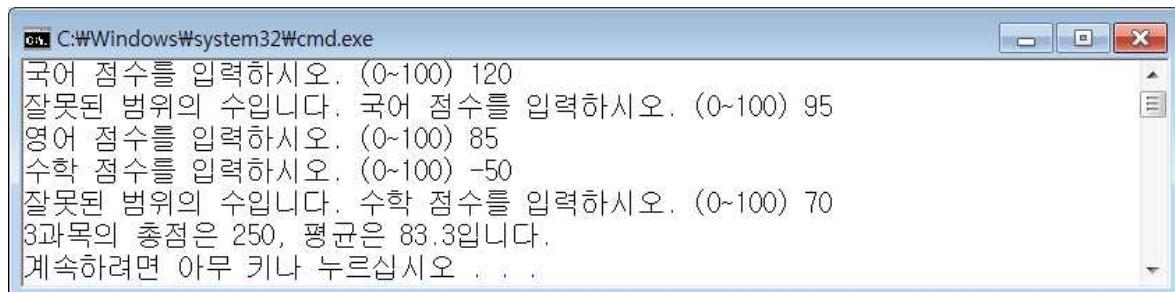
## [Step J] 파라미터와 리턴 값이 모두 있는 함수 만들기

프로그램을 만들 때에 가장 일반적으로 사용하는 함수의 형태는 파라미터와 리턴 값이 모두 있는 함수이다. 함수를 호출할 때에 함수 실행에 필요한 값을 파라미터로 전달하고, 실행이 끝나면 그 결과로 얻어내야 하는 값을 리턴받는 것이다. 이번 단계에서는 이런 형식의 함수를 제작하는 연습을 하도록 한다.

앞의 [Step I]에서 만들어 본 GetScore() 함수를 어떤 과목의 점수를 입력받아야 하는지를 명시해서 호출하도록 변형해보도록 하자. 즉, 파라미터로 과목의 이름을 전달하는 것이다. 이를 위해서는 함수의 선언부를 다음과 같이 바꾸어야 한다. 진하게 표시한 부분이 이전 소스에서 변경된 부분이다.

```
# -*- coding: utf-8 -*-
# 예제 ex_J1.py
def GetScore(name) :           # 파라미터로 과목명(name)을 전달받는다.
    while 1 :
        jumsu = int(input("%s 점수를 입력하시오. (0~100) "%name))
        if 0<=jumsu<=100 : return jumsu
        else: print("잘못된 범위의 수입니다.", end=" ")

# 메인함수 시작
class_name = ["국어", "영어", "수학"]
class_score = []
sum = 0
for i in range(3):
    score = GetScore(class_name[i])
    class_score.append(score)
    sum += score
average = sum / 3.0
print("3과목의 총점은 %d, 평균은 %.1f입니다."%(sum, average))
```

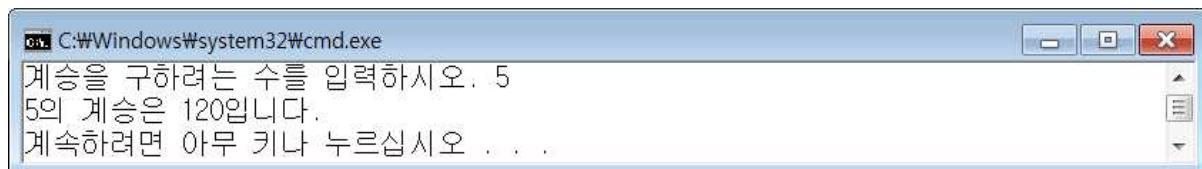


이번에는 함수에서 자기 자신을 다시 호출하는 함수, 즉 재귀 호출 함수를 만들어보자. 함수가 자기 자신을 다시 호출하는 방식이 왜 필요할까?

함수를 제작하다보면 함수의 구현 내용이 함수 자신을 사용하도록 정의되는 경우가 있다. 예를 들어 계승(factorial)을 구하는 경우를 생각해보자. 계승이란 1부터 자기 자신의 수까지를 모두 곱하는 것이다. 즉 5의 계승은  $1*2*3*4*5$  인 120이다. 먼저 반복문을 사용해서 계승을 구하는 함수를 만들어보면 다음과 같다. 구현내용에서 보는 바와 같이 1부터 구하려는 수까지 모두 곱해나가면 되는 것이다. 아래 구문을 쉽게 이해할 수 있을 것이다.

```
# -*- coding: utf-8 -*-
# 예제 ex_J2.py
def FactorialNoRecur(n) :
    result = 1
    for i in range(1, n+1) :
        result = result * i
    return result

# 메인함수 시작
num = int(input("계승을 구하려는 수를 입력하시오. "))
print("%d의 계승은 %d입니다."%(num, FactorialNoRecur(num)))
```



그러면 이제 같은 작업을 수행하는 함수를 재귀 호출 함수로 만들어보도록 하자. 재귀 호출 함수는 자신을 불러야 하기 때문에 계승의 원리를 다음과 같이 생각해야 한다.

$$\begin{aligned} n\text{의 계승} &= n * (n-1) * (n-2) * \dots * 2 * 1 \\ &= n * \{(n-1) * (n-2) * \dots * 2 * 1\} \\ &= n * \{(n-1)\text{의 계승}\} \end{aligned}$$

이렇게 해서 계승의 계산 원리 속에 자신보다 하나 적은 수의 계승을 넣을 수 있다. 물론 계승을 무한대로 재귀 호출 할 수 없으며, 기본이 되는 조건이 있어야 한다. 여기에서는  $n$ 의 값이 1인 경우에 더 이상 자기 자신을 호출하지 않도록 해 준다. 즉 다음과 같이  $n$ 의 계승을 재귀 호출의 형태로 정의할 수 있다.

n의 계승	= $n * \{(n-1)\text{의 계승}\}$	( $n$ 이 1보다 큰 경우)
또는	= 1	( $n$ 이 1이하인 경우)

이렇게 정의한 계승을 재귀 호출 함수로 만들어 보면 다음과 같다.

```
# -*- coding: utf-8 -*-
# 예제 ex_J3.py
def FactorialRecur(n) :
    if n <= 1 : return 1
    else : return n * FactorialRecur(n-1)
# 메인함수 시작
num = int(input("계승을 구하려는 수를 입력하시오. "))
print("%d의 계승은 %d입니다."%(num, FactorialRecur(num)))
```

이렇게 재귀 호출 함수를 만들 때에는 반드시 함수 자기 자신을 사용하여 정의할 수 있어야 하며, 또한 더 이상의 재귀 호출이 일어나지 않는 기본 조건을 넣어 주어야 한다.

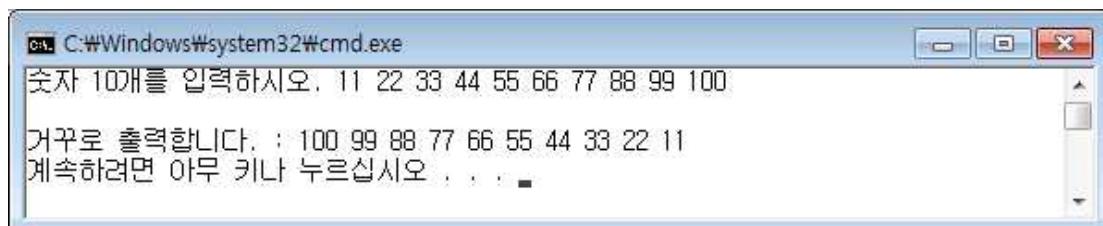
비슷한 예로 1부터 n까지의 숫자들을 모두 더한 값을 알아내는 작업을 재귀 호출 함수로 만든다면 다음 구문처럼 하면 된다.

```
def SumToOne(n) :
    if n == 1 : return 1 # n이 1인 경우
    else : return (n + SumToOne(n-1)) # n이 1보다 큰 경우
```

10개의 숫자가 들어 있는 리스트를 역순으로 출력하는 재귀 호출 함수는 다음과 같다.

```
# -*- coding: utf-8 -*-
# 예제 ex_J4.py
def PrintReverse(numbers, position) : # numbers 숫자 리스트, position 출력할 마지막 위치
    print(numbers[position],end=" ") # 가장 마지막 숫자 출력
    if position > 0 : PrintReverse(numbers, position-1) # 출력위치 1개 감소 후 출력

# 메인함수 시작
numlist = input("숫자 10개를 입력하시오. ").split()
for i in range(10):
    numlist[i] = int(numlist[i])
print("거꾸로 출력합니다. :",end=" ")
PrintReverse(numlist, 9)
```



## ○ 실습 문제

### [J01] 나이 계산 및 연령대 판정

[G01] 문제를 참고하여 최대 100명까지 사람들의 2012보다 큰 년도가 입력되기 전까지 태어난 년도를 입력받도록 하라. 입력이 끝나면 AskAge() 함수를 사용해서 지금까지 입력된 사람들의 나이를 모두 출력하고, 연령대 별로 각각 몇 명인지 출력하라.

AskAge()에서는 태어난 년도를 입력하면 나이를 출력한 후, 유아, 어린이, 청소년, 청년, 중년, 노년 여부를 판정하여 연령대 번호를 리턴한다.

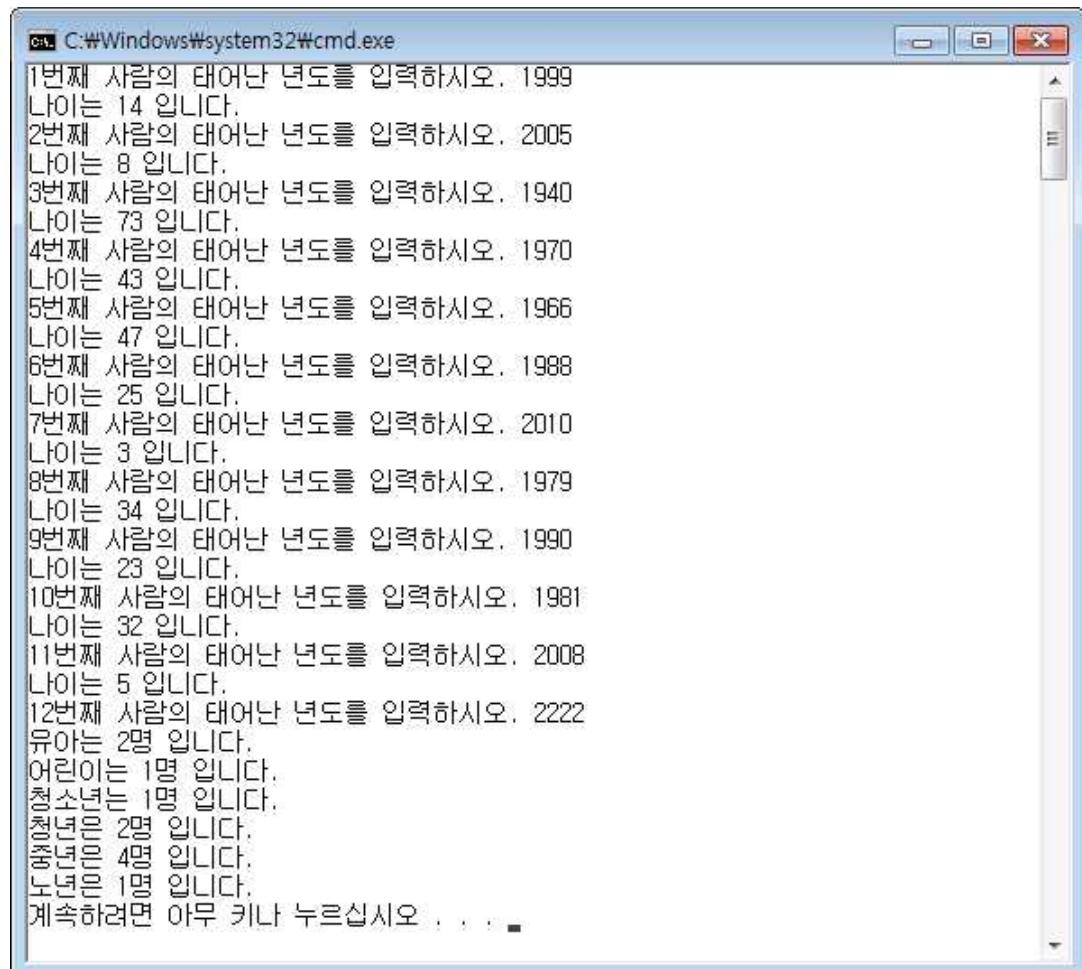
단, 나이 =  $2012 - \text{태어난 년도} + 1$ 로 계산하고 연령대 구분은 다음과 같이 판정한다.

7세 미만 : 유아, 7세 이상 ~ 13세미만 : 어린이, 13세 이상 ~ 20세 미만 : 청소년,

20세 이상 ~ 30세 미만 : 청년, 30세 이상 ~ 60세 미만 : 중년, 60세 이상 : 노년

함수의 선언부는 다음과 같다.

```
def AskAge(birthyear) :
    파라미터) birthyear : 태어난 년도
    리턴값) 계산한 나이에 따른 연령대 번호 (0.유아, 1.어린이, 2.청소년, 3.청년, 4.중년, 5.노년)
```



## [J02] 심사점수 계산

심사점수를 10개를 입력받아 리스트에 저장한 후, 이 리스트를 파라미터로 하여 가장 큰 점수를 구하는 `Max()`와 가장 작은 점수를 구하는 `Min()`을 사용하여 10개의 점수 중 최대점수와 최소점수를 제외한 8개의 점수에 대한 평균을 계산하여 출력하라.

함수의 선언부는 다음과 같다.

```
def Max(num) :
```

파라미터) num : 숫자 리스트

리턴값) 숫자 리스트에서 가장 큰 값

```
def Min(num) :
```

파라미터) num : 숫자 리스트

리턴값) 숫자 리스트에서 가장 작은 값

The screenshot shows a Windows command-line interface (cmd.exe) window titled 'C:\Windows\system32\cmd.exe'. The window contains the following text:

```
1번 심사점수를 입력하시오. 7.5
2번 심사점수를 입력하시오. 9.9
3번 심사점수를 입력하시오. 8.4
4번 심사점수를 입력하시오. 6.4
5번 심사점수를 입력하시오. 5.8
6번 심사점수를 입력하시오. 8.0
7번 심사점수를 입력하시오. 6.9
8번 심사점수를 입력하시오. 7.0
9번 심사점수를 입력하시오. 7.3
10번 심사점수를 입력하시오. 8.9
가장 큰 점수와 가장 작은 점수를 제외한 8개의 점수에 대한 평균은 7.6 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

### [J03] 물의 온도 구간 개수 판정

물의 온도를 10회 입력받은 후, 이 물이 각각 어느 정도의 온수인지 AskWater()를 통해 판정하여 그 결과를 출력하라. 출력할 내용은 입력된 10개의 온도 값, 냉수 입력 횟수, 미온수 입력 횟수, 온수 입력 횟수, 끓는 물 입력 횟수를 각각 출력하라.

단, 온수의 판정 구간은 다음과 같이 판정한다.

0도 ~ 25도 미만 : 냉수

25도 ~ 40도 미만 : 미온수

40도 ~ 80도 미만 : 온수

80도 이상 : 끓는 물

함수의 선언부는 다음과 같다.

```
def AskWater(degree) :
    파라미터) degree: 온도
    리턴 값) 온도 판정 번호 (0.냉수, 1.미온수, 2.온수, 3.끓는 물)
```

```
C:\Windows\system32\cmd.exe
1번째 물의 온도를 입력하시오. 15.5
2번째 물의 온도를 입력하시오. 20.5
3번째 물의 온도를 입력하시오. 27.7
4번째 물의 온도를 입력하시오. 35.0
5번째 물의 온도를 입력하시오. 40.5
6번째 물의 온도를 입력하시오. 66.7
7번째 물의 온도를 입력하시오. 90.1
8번째 물의 온도를 입력하시오. 100.0
9번째 물의 온도를 입력하시오. 77.7
10번째 물의 온도를 입력하시오. 28.0
1번 물의 온도는 15.5도 입니다.
2번 물의 온도는 20.5도 입니다.
3번 물의 온도는 27.7도 입니다.
4번 물의 온도는 35.0도 입니다.
5번 물의 온도는 40.5도 입니다.
6번 물의 온도는 66.7도 입니다.
7번 물의 온도는 90.1도 입니다.
8번 물의 온도는 100.0도 입니다.
9번 물의 온도는 77.7도 입니다.
10번 물의 온도는 28.0도 입니다.
냉수 입력 횟수 2 입니다.
미온수 입력 횟수는 3 입니다.
온수 입력 횟수는 3 입니다.
끓는 물 입력 횟수는 2 입니다.
계속하려면 아무 키나 누르십시오 . . .
```

### [J04] 연중 날짜 계산 함수를 이용한 날짜 간격 세기

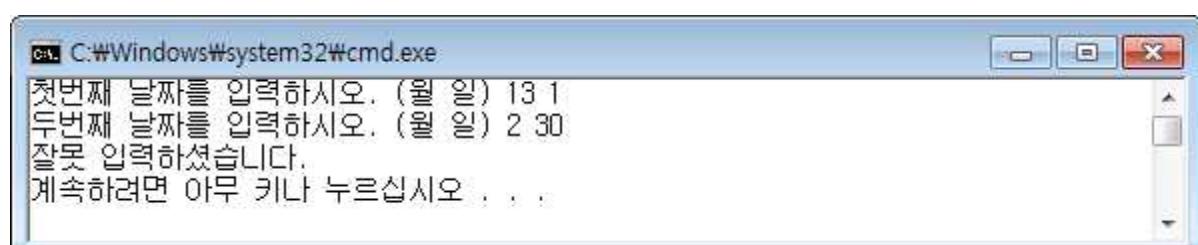
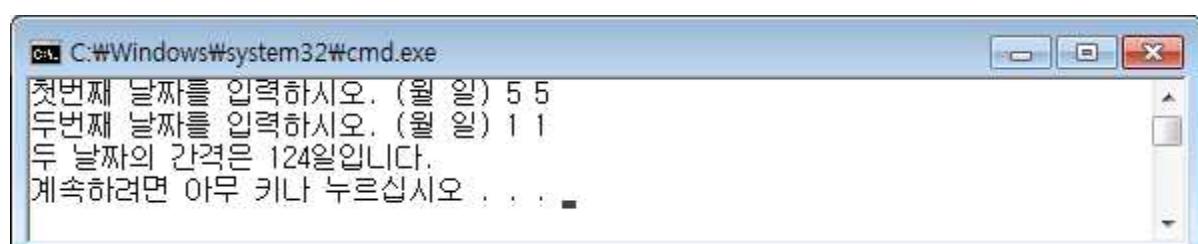
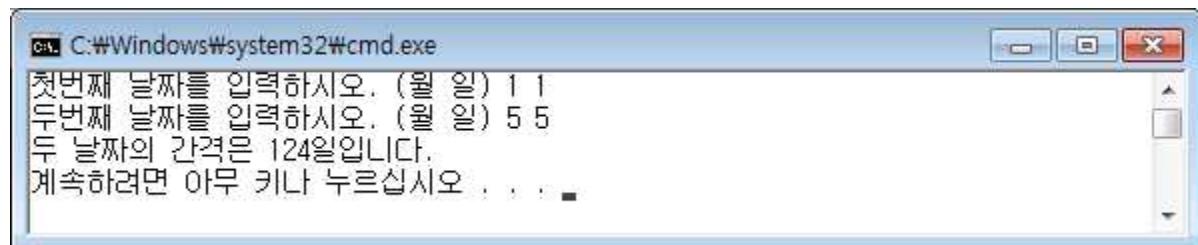
날짜 2개를 입력받은 후, 이 2개의 날짜 간격은 며칠인지 계산하여 출력하라. 단, 월과 일로 파라미터로 넘기면 이 날짜가 1년 중 몇 번째 날에 해당되는지 리턴하는 함수 CalcDate() 함수를 만들어 사용하라. 이 함수에서 매 월의 날 수 계산 시 다음과 같이 매월의 날 수를 리스트로 만들어 이를 이용하여 계산하라.

```
monthdays = [31,28,31,30,31,30,31,31,30,31,30,31]      # 1~12월의 날 수
```

함수의 선언부는 다음과 같다.

```
def CalcDate(month, day) :  
    파라미터) month : 월, day : 일
```

리턴값) 1년 중 해당 날짜가 몇 번째 날인지의 결과 값 (1~365)



## [J05] 주차 관리 시스템

주차장에서 차량들의 주차 관리 시스템을 만들어라. 차량마다 주차를 시작한 시간을 시와 분으로 입력받고, 주차를 종료한 시간을 시와 분으로 입력받은 후, 이를 CalcParking() 함수에 파라미터로 넘겨 주차요금을 리턴받도록 하라. 차량이 더 있는지 물어서 더 이상 차량이 없을 때까지 반복해서 요금을 계산하되 반복이 끝나면 지금까지 계산한 차량의 수량과 총 주차요금을 화면에 출력하라. 주차요금은 10분당 500원으로 한다.

함수의 선언부는 다음과 같다.

```
def CalcParking(start_h, start_m, end_h, end_m) :
    파라미터) start_h : 주차시작 시, start_m : 주차시작 분, end_h : 주차종료 시, end_m : 주차종료
    분
    리턴값) 주차시작 시간(시, 분)부터 종료 시간(시, 분)까지의 주차요금(원)
```



The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the following interaction:

```

1번 차량 주차 시작 시각 (시 분) : 10 30
1번 차량 주차 종료 시각 (시 분) : 11 15
주차요금 : 2500원
더 입력하시겠습니까?(Y/N) Y
2번 차량 주차 시작 시각 (시 분) : 9 10
2번 차량 주차 종료 시각 (시 분) : 15 10
주차요금 : 18000원
더 입력하시겠습니까?(Y/N) Y
3번 차량 주차 시작 시각 (시 분) : 12 10
3번 차량 주차 종료 시각 (시 분) : 14 55
주차요금 : 8500원
더 입력하시겠습니까?(Y/N) Y
4번 차량 주차 시작 시각 (시 분) : 11 00
4번 차량 주차 종료 시각 (시 분) : 11 05
주차요금 : 500원
더 입력하시겠습니까?(Y/N) N
주차차량 4대의 총 주차 요금은 29500원입니다.
계속하려면 아무 키나 누르십시오 . .

```

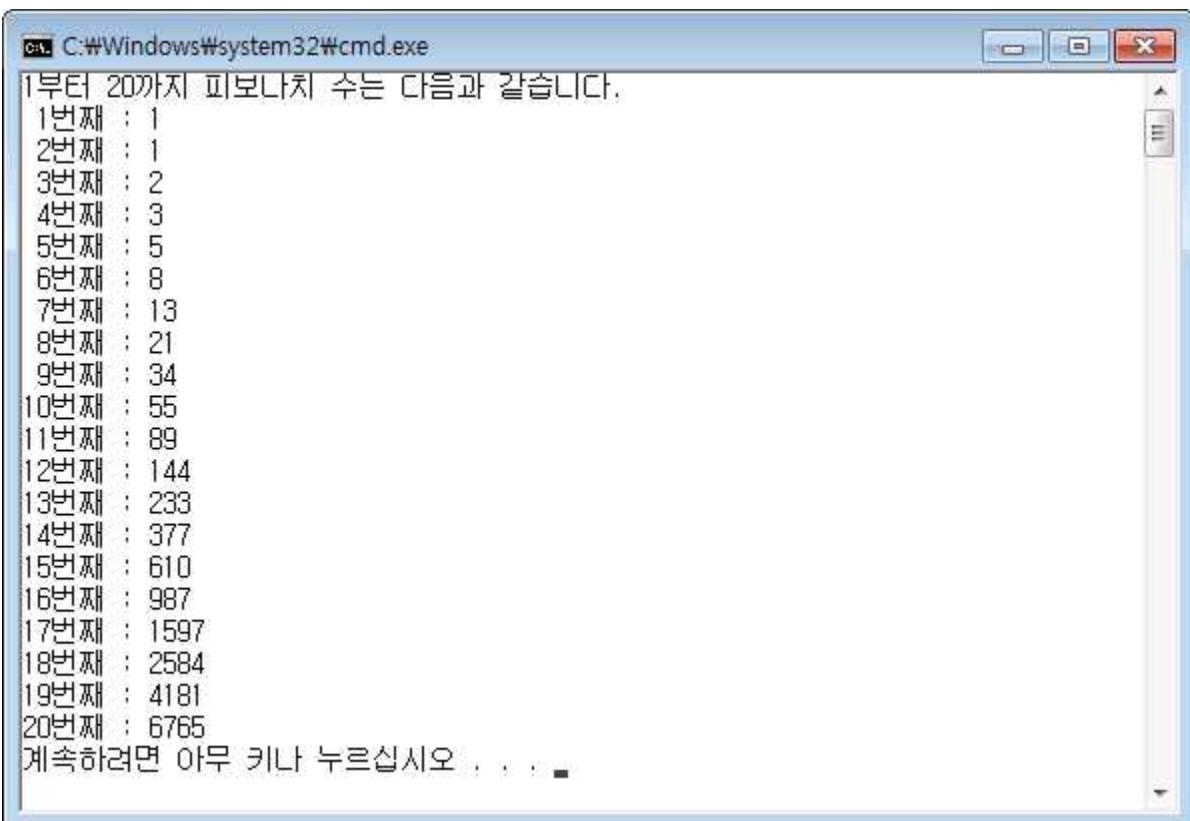
### [J06] 피보나치 수 구하기

n이 1부터 20까지 증가하는 경우 각각의 피보나치 수  $\text{fibonacci}(n)$ 을 출력하라. 피보나치 수는 다음과 같이 정의한다.

$$\text{fibonacci}(n) = \begin{cases} 1 & (n = 1 \text{ 또는 } n = 2 \text{인 경우}) \\ \text{fibonacci}(n-1) + \text{fibonacci}(n-2) & (n > 2 \text{인 경우}) \end{cases}$$

단, 함수의 선언부는 다음과 같이 사용하라.

```
def fibonacci(n) :
```



The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the following text:

```
1부터 20까지 피보나치 수는 다음과 같습니다.
1번째 : 1
2번째 : 1
3번째 : 2
4번째 : 3
5번째 : 5
6번째 : 8
7번째 : 13
8번째 : 21
9번째 : 34
10번째 : 55
11번째 : 89
12번째 : 144
13번째 : 233
14번째 : 377
15번째 : 610
16번째 : 987
17번째 : 1597
18번째 : 2584
19번째 : 4181
20번째 : 6765
계속하려면 아무 키나 누르십시오 . . .
```

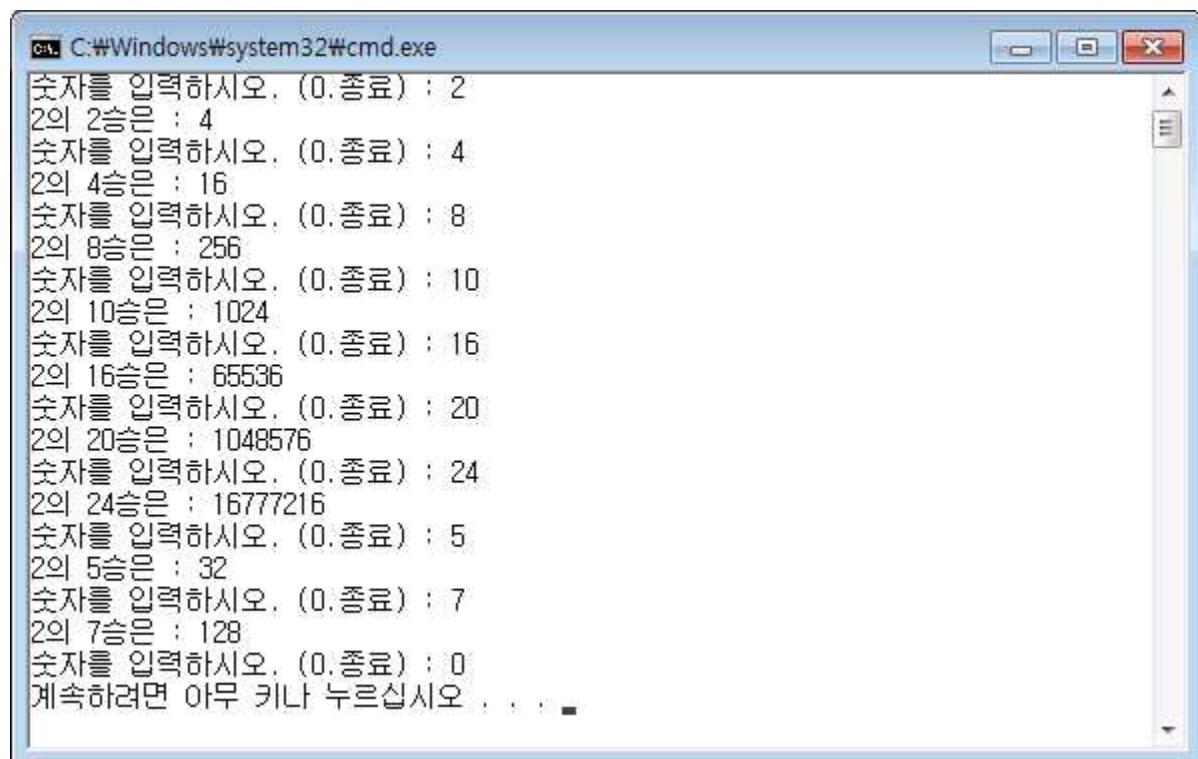
### [J07] 2의 제곱수 구하기

반복해서 임의의 숫자  $n$ 을 입력받은 후  $2^n$ 을 계산하여 출력하되, 재귀함수를 이용하여 계산하라. 이 때 사용할 재귀함수 poweroftwo()의 정의는 다음과 같다.

$$\text{poweroftwo}(n) = \begin{cases} 1 & (n = 0 \text{인 경우}) \\ 2 \times \text{poweroftwo}(n-1) & (n > 0 \text{인 경우}) \end{cases}$$

단, 함수의 선언부 다음과 같이 사용하라.

```
def poweroftwo(n) :
```



The screenshot shows a Windows command prompt window titled 'cmd.exe' with the path 'C:\Windows\system32\cmd.exe'. The window displays the output of a Python script that repeatedly prompts the user for a number and prints its value as a power of 2. The user inputs numbers from 2 to 7, and the script outputs the corresponding powers of 2. The script then asks if the user wants to continue, and the user responds with '0' to exit.

```
C:\Windows\system32\cmd.exe
숫자를 입력하시오. (0.종료) : 2
2의 2승은 : 4
숫자를 입력하시오. (0.종료) : 4
2의 4승은 : 16
숫자를 입력하시오. (0.종료) : 8
2의 8승은 : 256
숫자를 입력하시오. (0.종료) : 10
2의 10승은 : 1024
숫자를 입력하시오. (0.종료) : 16
2의 16승은 : 65536
숫자를 입력하시오. (0.종료) : 20
2의 20승은 : 1048576
숫자를 입력하시오. (0.종료) : 24
2의 24승은 : 16777216
숫자를 입력하시오. (0.종료) : 5
2의 5승은 : 32
숫자를 입력하시오. (0.종료) : 7
2의 7승은 : 128
숫자를 입력하시오. (0.종료) : 0
계속하려면 아무 키나 누르십시오 . . .
```

### [J08] Ackermann 수 구하기

Ackermann's Function A 는 다음과 같이 정의된다.  $A( i, j )$  를 재귀 호출 함수로 만들고, 이 함수를 이용하여  $A(0,0)$  에서  $A(3,3)$  의 값을 구하라.

#### Ackermann's Function A

$A(0, j) = j+1$                        $i = 0$  이고,  $j \geq 0$  인 경우

$A(i, 0) = A(i-1, 1)$                $i > 0$  이고  $j = 0$  인 경우

$A(i, j) = A(i-1, A(i, j-1))$      $i$  와  $j$  모두 0보다 큰 경우

단, 함수의 선언부는 다음과 같이 사용하라.

```
def Ackermann(i, j) :
```

The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window displays the following output:

```
Ackermann(0, 0) = 1
Ackermann(0, 1) = 2
Ackermann(0, 2) = 3
Ackermann(0, 3) = 4
Ackermann(1, 0) = 2
Ackermann(1, 1) = 3
Ackermann(1, 2) = 4
Ackermann(1, 3) = 5
Ackermann(2, 0) = 3
Ackermann(2, 1) = 5
Ackermann(2, 2) = 7
Ackermann(2, 3) = 9
Ackermann(3, 0) = 5
Ackermann(3, 1) = 13
Ackermann(3, 2) = 29
Ackermann(3, 3) = 61
계속하려면 아무 키나 누르십시오 . . .
```

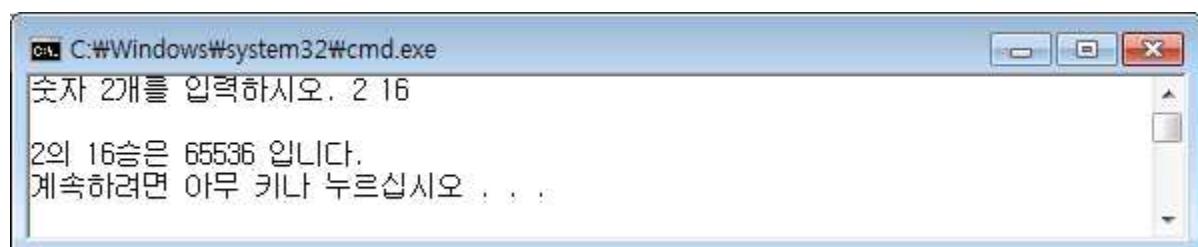
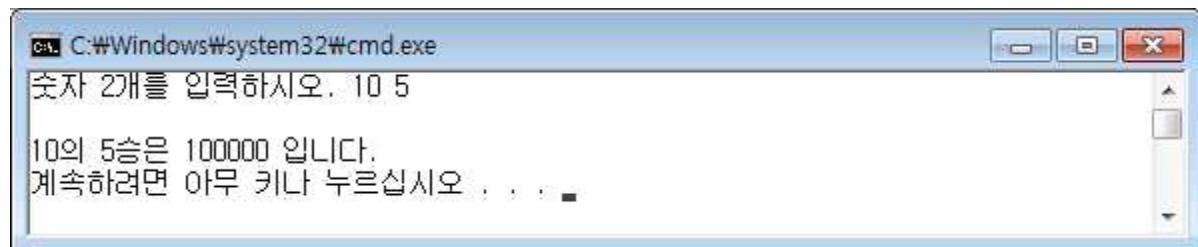
### [J09] pow() 함수 만들기

$n^a$ 를 계산할 수 있는 math 모듈 내의 pow() 함수와 같은 일을 하는 power() 함수를 재귀 호출을 이용하여 만들어라. 그리고 숫자 2개(num1, num2)를 입력받아  $\text{num1}^{\text{num2}}$  를 계산하라.  
단, power() 함수는 다음과 같이 정의된다.

$$\text{power}(n, a) = \begin{cases} 1 & (a=0 \text{인 경우}) \\ n & (a=1 \text{인 경우}) \\ \text{power}(n, \frac{a}{2}) \times \text{power}(n, \frac{a}{2}) & (a > 1 \text{이고 짝수인 경우}) \\ \text{power}(n, \frac{a}{2}) \times \text{power}(n, \frac{a}{2}) \times n & (a > 1 \text{이고 홀수인 경우}) \end{cases}$$

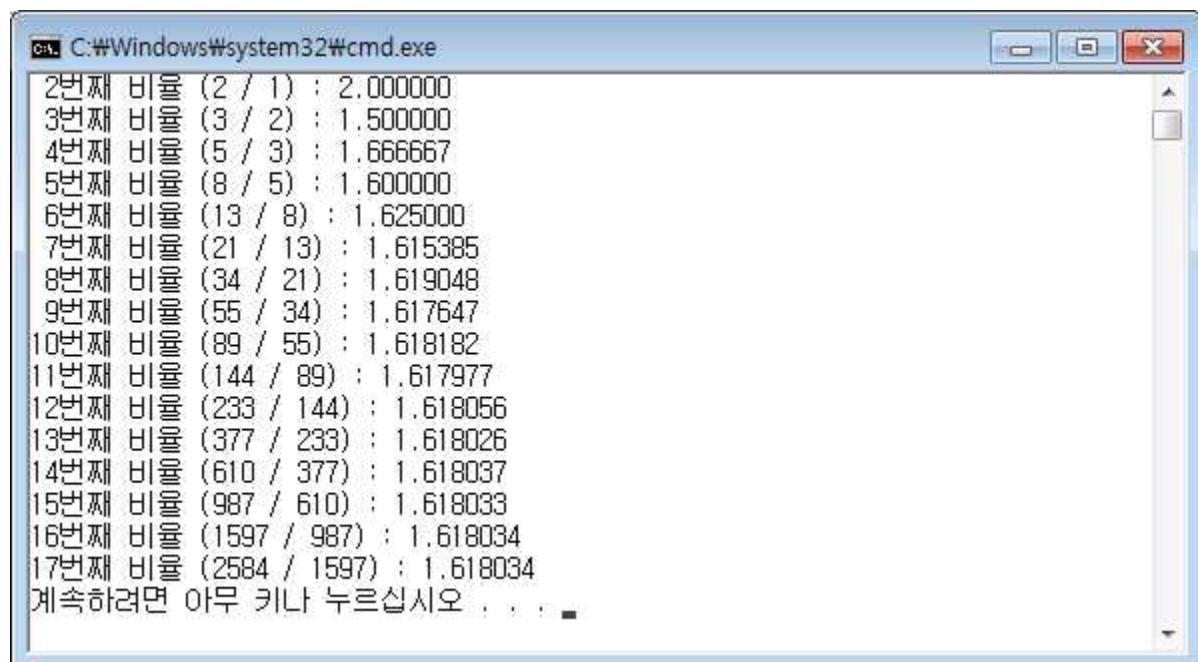
단, 함수의 선언부는 다음과 같이 사용하라.

```
def power( num1, num2 ) :
```



### [J10] 피보나치 수열로 황금비율 구하기

문제 [J06]에서 제작한 피보나치 함수 `fibonacci(n)`를 사용하여 황금비율을 찾아내라.  $n$ 번째 황금비율이란 피보나치 수열의 연속적인 2개의 숫자의 비율( $n+1$ 번째수 /  $n$ 번째 수)로 정한다.  
단, 계산한 비율이 직전의 비율에 비해 그 차이가 백만분의 1보다 작게 되면 멈추도록 하라.



```
C:\Windows\system32\cmd.exe
2번째 비율 (2 / 1) : 2.000000
3번째 비율 (3 / 2) : 1.500000
4번째 비율 (5 / 3) : 1.6666667
5번째 비율 (8 / 5) : 1.600000
6번째 비율 (13 / 8) : 1.625000
7번째 비율 (21 / 13) : 1.615385
8번째 비율 (34 / 21) : 1.619048
9번째 비율 (55 / 34) : 1.617647
10번째 비율 (89 / 55) : 1.618182
11번째 비율 (144 / 89) : 1.617977
12번째 비율 (233 / 144) : 1.618056
13번째 비율 (377 / 233) : 1.618026
14번째 비율 (610 / 377) : 1.618037
15번째 비율 (987 / 610) : 1.618033
16번째 비율 (1597 / 987) : 1.618034
17번째 비율 (2584 / 1597) : 1.618034
계속하려면 아무 키나 누르십시오 . . .
```

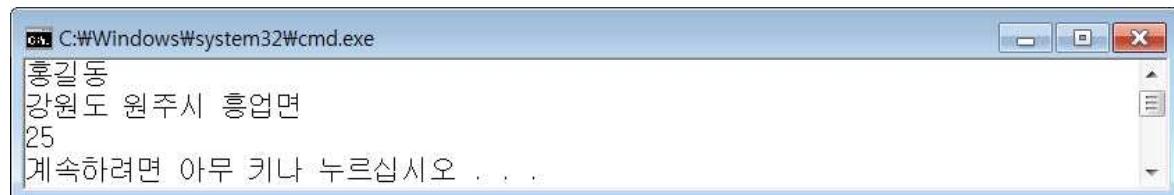
## [Step K] 파일 사용하기

이번 단계에서는 텍스트 파일을 사용하여 데이터를 읽고 쓰는 방법을 연습하도록 하겠다. 파일을 사용하기 위해서는 파일 변수를 사용해야 하는데, 이는 특정 텍스트 파일을 읽거나 쓰기 위해서 여는 함수인 `open()`을 사용해 만들 수 있다. 파일로부터 읽어오는 구문을 살펴보자. 데이터 파일인 `data.txt`에는 다음과 같은 내용이 들어 있다고 가정한다.

```
홍길동
강원도 원주시 흥업면
25
```

위 파일로부터 데이터를 읽어들여 화면에 출력하는 프로그램은 다음과 같다.

```
# -*- coding: utf-8 -*-
# 예제 ex_K1.py
file = open("data.txt", "r")           # 텍스트 파일 data.txt 를 읽는 용도로 연다.
lines = file.readlines()               # 파일의 모든 줄을 읽어들인다.
for line in lines:                   # 읽어들인 모든 줄의 각각의 줄에 대해 반복
    print(line.strip())              # 각각의 줄의 좌우 공백을 없앤 후에 출력한다.
file.close()                         # 파일을 닫는다.
```



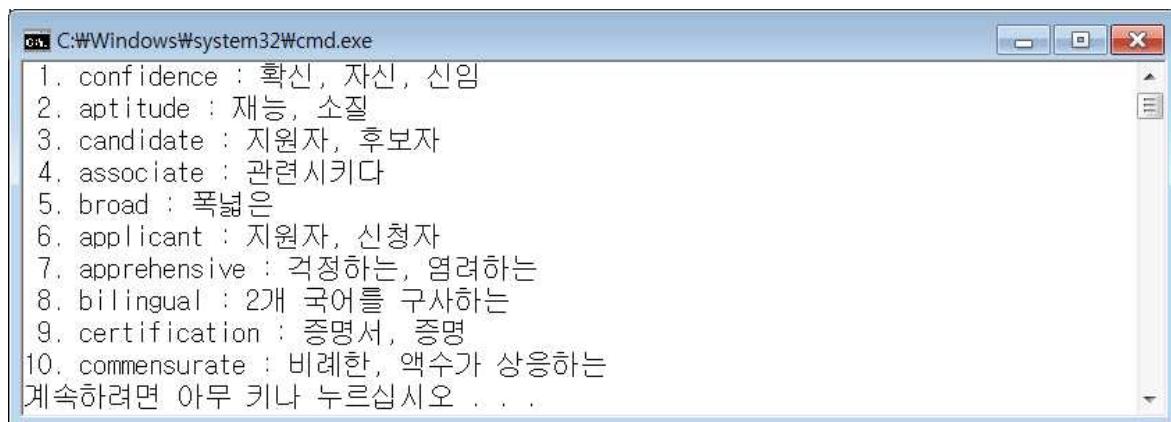
텍스트 파일의 모든 라인을 읽지 않고, 한 줄씩 읽어들일 수도 있다. 다음 프로그램은 `word.txt`에 들어있는 영어단어와 뜻 10개를 한 줄씩 읽어 들이면서 딕셔너리에 저장한다. `word.txt`의 내용은 다음과 같이 영어단어와 한글 뜻 사이에 콜론(":")으로 나뉘어져 있다.

```
applicant:지원자, 신청자
apprehensive:걱정하는, 염려하는
aptitude:재능, 소질
associate:관련시키다
bilingual:2개 국어를 구사하는
broad:폭넓은
candidate:지원자, 후보자
certification:증명서, 증명
commensurate:비례한, 액수가 상응하는
```

confidence:확신, 자신, 신임

프로그램의 소스는 다음과 같다.

```
# -*- coding: utf-8 -*-
# 예제 ex_K2.py
EngDic = {} # 딕셔너리 초기화
file = open("word.txt", "r")
while 1 :
    line = file.readline().strip().split(":") # 파일에서 한 줄을 읽어 공백을 제거하고,
                                                # 콜론 문자를 기준으로 분할한다.
    if len(line) == 2 : # 단어와 뜻 2개로 분리된 경우에만
        EngDic[line[0]] = line[1] # 딕셔너리 EngDic에 넣는다.
    else:
        break
file.close()
num = 1
for word in EngDic.keys() : # 딕셔너리의 모든 키에 대해서
    print("%2d.%s,%s:%s" % (num, word, ":", EngDic[word])) # 키와 값을 화면에 출력한다.
    num += 1
```



파일에 데이터를 넣는 방법은 파일을 쓰기 용도로 연 다음에 적절한 `write()`와 같은 파일 쓰기 함수들을 사용하면 되는데, 다음 프로그램은 10개의 영어단어와 뜻으로 된 딕셔너리의 내용을 텍스트 파일에 저장한다.

```
# coding: euc-kr
# 예제 ex_K3.py
EngDic = {"applicant" : "지원자, 신청자", "apprehensive" : "걱정하는, 염려하는", "aptitude" : "재능, 소질", "associate" : "관련시키다", "bilingual" : "2개 국어를 구사하는", "broad" : "폭넓은",
```

```
"candidate" : "지원자, 후보자", "certification" : "증명서, 증명", "commensurate" : "비례한, 액수가 상응하는", "confidence" : "확신, 자신, 신임"}  
file = open("word2.txt", "w")  
num = 1  
for word in EngDic.keys() :  
    line = "%2d. %s : %s\n"%(num,word,EngDic[word])  
    file.write(line)  
    num += 1  
file.close()
```

새로 만들어진 word2.txt의 내용은 다음과 같다.

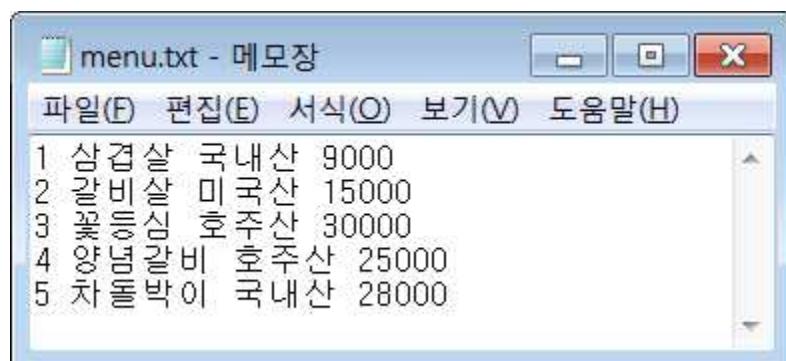
1. confidence : 확신, 자신, 신임
2. commensurate : 비례한, 액수가 상응하는
3. bilingual : 2개 국어를 구사하는
4. aptitude : 재능, 소질
5. associate : 관련시키다
6. certification : 증명서, 증명
7. broad : 폭넓은
8. applicant : 지원자, 신청자
9. apprehensive : 걱정하는, 염려하는
10. candidate : 지원자, 후보자

## ○ 실습 문제

### [K01] 메뉴판 저장하기

화면과 같이 5개의 메뉴내용을 입력받은 후, 화면에 출력하고, menu.txt 파일에 저장하는 프로그램을 제작하고 테스트하라.

```
C:\Windows\system32\cmd.exe
1번째 메뉴의 번호, 메뉴명, 원산지, 가격을 입력하시오. 1 삼겹살 국내산 9000
2번째 메뉴의 번호, 메뉴명, 원산지, 가격을 입력하시오. 2 갈비살 미국산 15000
3번째 메뉴의 번호, 메뉴명, 원산지, 가격을 입력하시오. 3 꽃등심 호주산 30000
4번째 메뉴의 번호, 메뉴명, 원산지, 가격을 입력하시오. 4 양념갈비 호주산 25000
5번째 메뉴의 번호, 메뉴명, 원산지, 가격을 입력하시오. 5 차돌박이 국내산 28000
메뉴번호 메뉴이름 원산지 1인분가격
 1  삼겹살  국내산  9000
 2  갈비살  미국산 15000
 3  꽃등심  호주산 30000
 4  양념갈비  호주산 25000
 5  차돌박이  국내산 28000
menu.txt에 저장하였습니다. 계속하려면 아무 키나 누르십시오 . . .
```



## [K02] 메뉴판 읽어오기

위의 [K01]문제에서 저장한 menu.txt 파일에서 메뉴 정보를 읽어와서 화면에 출력하라.



C:\Windows\system32\cmd.exe

```
menu.txt에서 메뉴정보를 읽어들였습니다.
메뉴번호 메뉴이름 원산지 1인분가격
1    삼겹살  국내산    9000
2    갈비살  미국산    15000
3    꽃등심  호주산    30000
4    양념갈비  호주산    25000
5    차돌박이  국내산    28000
계속하려면 아무 키나 누르십시오 . . .
```

### [K03] 좌표 저장하기

화면과 같이 10개의 좌표 값(x,y)을 입력받아 point.txt 파일에 저장하라. 그리고 저장한 파일을 다시 읽어들여서 리스트로 만든 후, 이 좌표가 몇사분면에 위치하는지를 알아내는 함수를 통해 각 사분면에 속하는 좌표의 갯수를 출력하라.

함수 선언부는 다음과 같다.

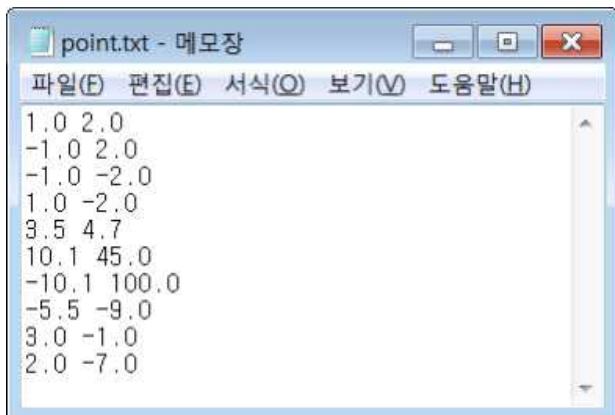
```
def get_area(x, y) :
```

파라미터 : x좌표, y좌표

리턴값 : 좌표가 위치한 사분면 (1,2,3,4, 만일 x 또는 y가 0이면 리턴값은 0이다.)

The screenshot shows a command-line interface window titled 'cmd C:\Windows\system32\cmd.exe'. The window contains the following text:

```
1번째 좌표의 x, y값을 입력하시오. --> 1.0 2.0
2번째 좌표의 x, y값을 입력하시오. --> -1.0 2.0
3번째 좌표의 x, y값을 입력하시오. --> -1.0 -2.0
4번째 좌표의 x, y값을 입력하시오. --> 1.0 -2.0
5번째 좌표의 x, y값을 입력하시오. --> 3.5 4.7
6번째 좌표의 x, y값을 입력하시오. --> 10.1 45.0
7번째 좌표의 x, y값을 입력하시오. --> -10.1 100.0
8번째 좌표의 x, y값을 입력하시오. --> -5.5 -9.0
9번째 좌표의 x, y값을 입력하시오. --> 3.0 -1.0
10번째 좌표의 x, y값을 입력하시오. --> 2.0 -7.0
1번째 좌표는 1사분면에 위치합니다.
2번째 좌표는 2사분면에 위치합니다.
3번째 좌표는 3사분면에 위치합니다.
4번째 좌표는 4사분면에 위치합니다.
5번째 좌표는 1사분면에 위치합니다.
6번째 좌표는 1사분면에 위치합니다.
7번째 좌표는 2사분면에 위치합니다.
8번째 좌표는 3사분면에 위치합니다.
9번째 좌표는 4사분면에 위치합니다.
10번째 좌표는 4사분면에 위치합니다.
1사분면의 좌표는 모두 3개입니다.
2사분면의 좌표는 모두 2개입니다.
3사분면의 좌표는 모두 2개입니다.
4사분면의 좌표는 모두 3개입니다.
point.txt에 저장하였습니다.
계속하려면 아무 키나 누르십시오 . . .
```



## [K04] 좌표 불러오기

위의 [K03]문제에서 저장한 point.txt 파일로부터 10개의 좌표를 읽어오는 함수를 제작하고, 이를 사용하여 각각의 좌표가 몇 사분면에 위치하는 지와 각 사분면에 속한 좌표의 수를 출력하라.

함수 선언부는 다음과 같다.

```
def load_point(filename) :  
    파라미터 : 좌표가 들어있는 파일명  
    리턴값 : 좌표 리스트
```

C:\Windows\system32\cmd.exe

```
point.txt에서 좌표정보를 읽어들였습니다.  
1번째 좌표( 1.0, 2.0)는 1사분면에 위치합니다.  
2번째 좌표( -1.0, 2.0)는 2사분면에 위치합니다.  
3번째 좌표( -1.0, -2.0)는 3사분면에 위치합니다.  
4번째 좌표( 1.0, -2.0)는 4사분면에 위치합니다.  
5번째 좌표( 3.5, 4.7)는 1사분면에 위치합니다.  
6번째 좌표( 10.1, 45.0)는 1사분면에 위치합니다.  
7번째 좌표(-10.1, 100.0)는 2사분면에 위치합니다.  
8번째 좌표( -5.5, -9.0)는 3사분면에 위치합니다.  
9번째 좌표( 3.0, -1.0)는 4사분면에 위치합니다.  
10번째 좌표( 2.0, -7.0)는 4사분면에 위치합니다.  
1사분면의 좌표는 모두 3개입니다.  
2사분면의 좌표는 모두 2개입니다.  
3사분면의 좌표는 모두 2개입니다.  
4사분면의 좌표는 모두 3개입니다.  
계속하려면 아무 키나 누르십시오 . . .
```

### [K05] 사용자 목록 저장하기

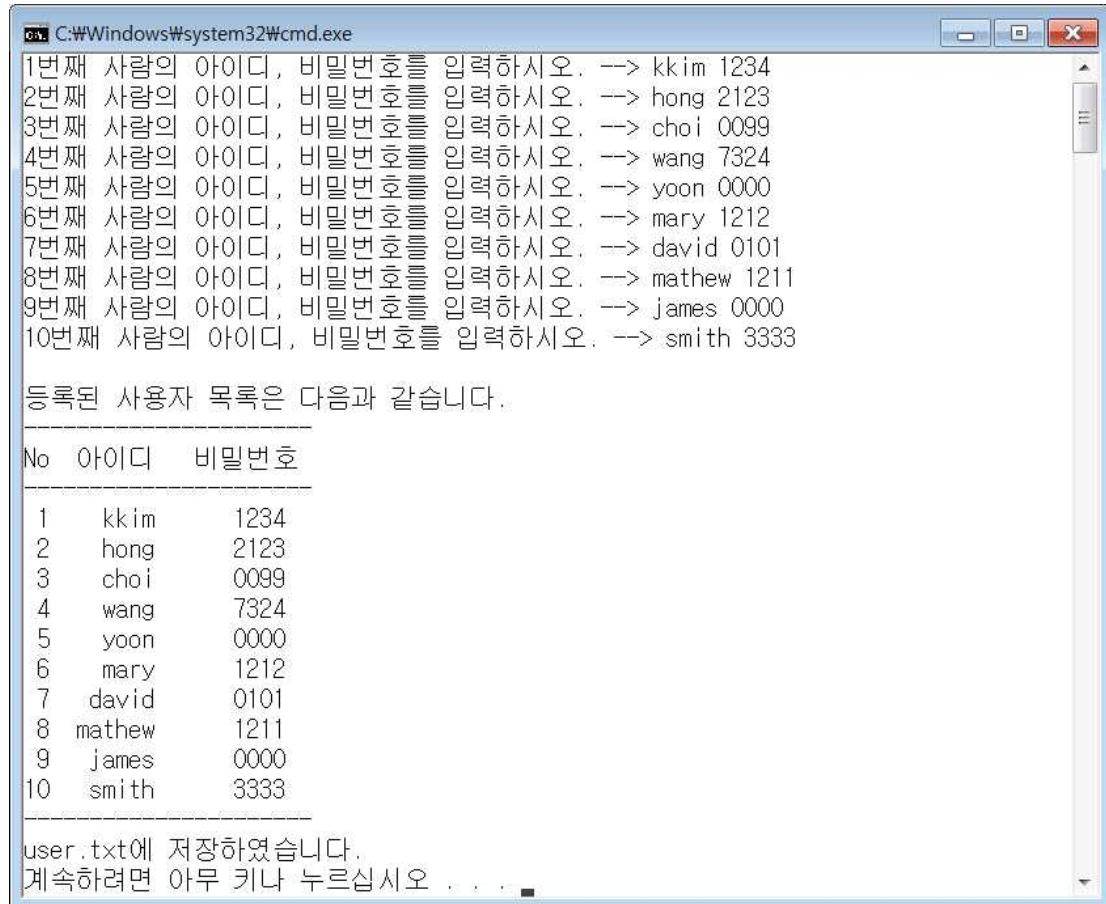
화면과 같이 10명의 아이디와 비밀번호를 입력받아 리스트로 만든 후, 이 리스트를 출력하고 파라미터로 넘겨 모든 사용자 정보를 텍스트 파일인 user.txt 파일에 저장하라.

함수 선언부는 다음과 같다.

```
def save_list(userlist, filename) :
```

파라미터 : 사용자 목록, 저장할 파일명

리턴값 : 없음



The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The window contains the following text:

```

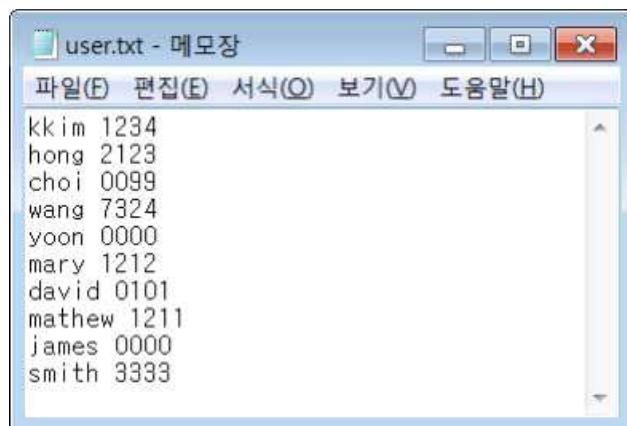
1번째 사람의 아이디, 비밀번호를 입력하시오. --> kk im 1234
2번째 사람의 아이디, 비밀번호를 입력하시오. --> hong 2123
3번째 사람의 아이디, 비밀번호를 입력하시오. --> cho i 0099
4번째 사람의 아이디, 비밀번호를 입력하시오. --> wang 7324
5번째 사람의 아이디, 비밀번호를 입력하시오. --> yoon 0000
6번째 사람의 아이디, 비밀번호를 입력하시오. --> mary 1212
7번째 사람의 아이디, 비밀번호를 입력하시오. --> david 0101
8번째 사람의 아이디, 비밀번호를 입력하시오. --> mathew 1211
9번째 사람의 아이디, 비밀번호를 입력하시오. --> james 0000
10번째 사람의 아이디, 비밀번호를 입력하시오. --> smith 3333

```

등록된 사용자 목록은 다음과 같습니다.

No	아이디	비밀번호
1	kk im	1234
2	hong	2123
3	cho i	0099
4	wang	7324
5	yoon	0000
6	mary	1212
7	david	0101
8	mathew	1211
9	james	0000
10	smith	3333

user.txt에 저장하였습니다.  
계속하려면 아무 키나 누르십시오 . . .



## [K06] 사용자 목록 불러오기

위의 [K05]문제에서 저장한 user.txt 파일로부터 사용자 정보를 읽어와서 이를 딕셔너리로 만든 후에 화면에 출력하라.

함수 선언부는 다음과 같다.

```
def load_list(filename) :
```

파라미터 : 읽을 파일명

리턴값 : 읽어들인 사용자 정보 리스트

The screenshot shows a Windows Command Prompt window titled 'cmd.exe' with the path 'C:\Windows\system32'. The window displays the following text:

```
C:\Windows\system32\cmd.exe
user.txt에서 10명의 사용자 정보를 읽어들였습니다.

등록된 사용자 목록은 다음과 같습니다.

No 아이디 비밀번호
1 kkim 1234
2 hong 2123
3 choi 0099
4 wang 7324
5 yoon 0000
6 mary 1212
7 david 0101
8 matthew 1211
9 james 0000
10 smith 3333

계속하려면 아무 키나 누르십시오 . . .
```

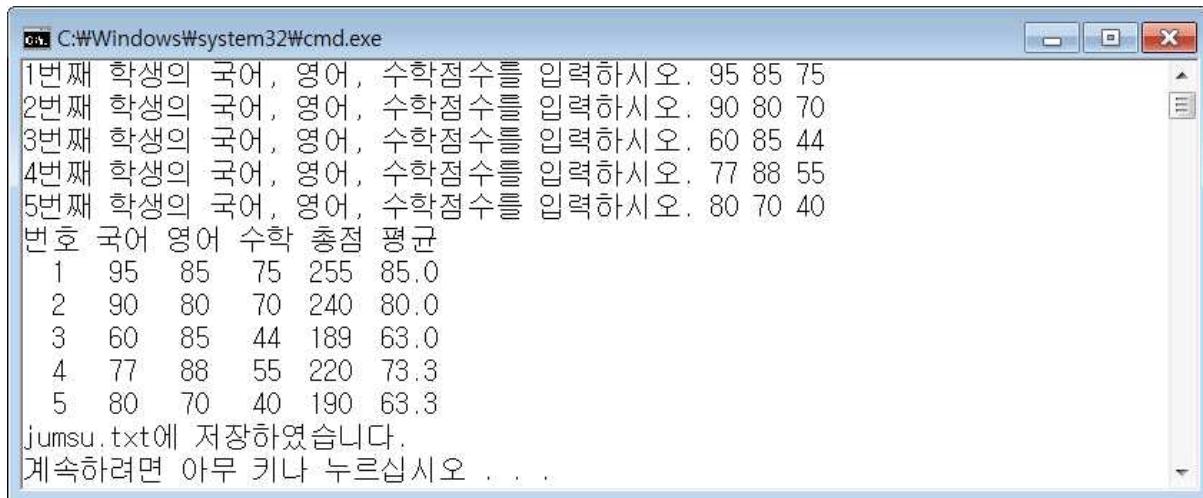
The window has a standard Windows title bar and a scroll bar on the right side. The text is displayed in a monospaced font.

### [K07] 학생 점수 결과 저장하기

5명의 학생들의 국어, 영어, 수학 점수를 입력받아 총점과 평균을 계산하고 이 정보를 리스트로 만든 후 이 리스트를 파라미터로 넘겨받아 텍스트 파일에 저장하는 함수를 만들고 테스트하라.

함수 선언부는 다음과 같다.

```
def save_jumsu(jumsulist filename) :  
    파라미터 : 점수 목록(국,영,수,총점,평균), 파일명  
    리턴값 : 없음
```



## 실습문제 목록

[A01] 나이 계산 .....	8
[A02] 온도 변환 .....	8
[A03] 직사각형 넓이 계산 .....	9
[A04] 아파트 평형 계산 .....	9
[A05] 날짜 계산 .....	10
[A06] 점수 계산 .....	10
[A07] 파일 용량 계산 .....	11
[B01] 나이 계산 및 미성년자 판정 .....	15
[B02] 온도 상호 변환 .....	16
[B03] 직사각형 넓이 계산 및 정사각형 판정 .....	17
[B04] 아파트 평형 계산 및 종류 판정 .....	18
[B05] 날짜 계산 .....	19
[B06] 점수 계산 .....	20
[B07] 파일 전송 시간 계산 .....	21
[B08] 다양한 조건 판정 .....	22
[B09] 비만 판정 .....	23
[C01] 나이 계산 및 연령대 판정 .....	26
[C02] 물의 온도 구간 판정 .....	27
[C03] 직사각형 형태 판정 .....	28
[C04] 아파트 평형 계산 및 종류 판정 .....	29
[C05] 연중 날짜 계산 .....	30
[C06] 점수 계산 .....	31
[C07] 파일 전송 시간 계산 .....	32
[C08] 3개의 수 중 최댓값과 최솟값 구하기 .....	33
[C09] 소득세 계산 .....	34
[C10] 간단한 사칙연산 계산기 .....	35
[C11] 윤년 판정하기 .....	36
[D01] 1부터 숫자 더하기 .....	41
[D02] 입력 받은 숫자들 중에서 가장 큰 수와 가장 작은 수 구하기 .....	42
[D03] 입력 받은 숫자들의 총합계와 평균 값 구하기 .....	43
[D04] 미성년자 숫자 세기 .....	44
[D05] 직사각형 형태 개수 세기 .....	45
[D06] 아파트 평형 계산 및 종류 판정 .....	46

[D07] 1차 함수의 좌표 구하기 .....	47
[D08] 2차 함수의 좌표 구하기 .....	48
[D09] 원하는 구구단의 단 출력하기 .....	49
[D10] 두 수의 배타적 배수 출력하기 .....	50
[E01] 입력한 숫자 크기의 정사각형 출력하기 .....	54
[E02] 입력한 숫자 크기의 높이를 갖는 우직각 삼각형 출력하기 .....	55
[E03] 입력한 숫자 크기의 높이를 갖는 이등변 삼각형 출력하기 .....	56
[E04] 홀수단 또는 짝수단의 구구단 출력하기 .....	57
[E05] 홀수단 또는 짝수단의 구구단을 열의 개수를 맞추어 출력하기 .....	58
[E06] 2차원 숫자 출력하기 .....	59
[F01] 두 번째로 큰 수의 순서 찾기 .....	63
[F02] 심사점수 계산 .....	64
[F03] 5명의 국, 영, 수 3과목 점수의 과목별 총점, 평균값 구하기 .....	65
[F04] 5명의 국, 영, 수 3과목 점수의 학생별 총점, 평균값 구하기 .....	66
[F05] 비만 판정 .....	67
[F06] 5층 아파트의 거주자 숫자 구하기 .....	68
[F07] 5층 아파트의 층별, 호수별 거주자 숫자 구하기 .....	69
[F08] 겹치지 않는 숫자 10개 입력 받기 .....	70
[F09] 리스트를 이용한 연중 날짜 계산 .....	71
[G01] 나이 계산 및 연령대 판정 .....	74
[G02] 물의 온도 구간 판정 .....	75
[G03] 점수 계산 .....	76
[G04] 부동산 중개 수수료 계산기 .....	77
[G05] PC방 이용료 계산기 .....	78
[G06] 쇼핑몰 매출 계산기 .....	79
[G07] 놀이공원 매표소 .....	80
[H01] 숫자 알아 맞추기 .....	87
[H02] 로또 번호 만들기 .....	88
[H03] 로또 번호 당첨 확인하기 .....	89
[H04] 가위바위보 게임하기 .....	90
[H05] 슬롯머신 만들기 .....	91
[H06] 5개 숫자의 제곱수 조합 구하기 .....	92
[H07] 표준편차 구하기 .....	93
[H08] 애너그램(Anagram) 판별하기 .....	94
[H09] 회문(Palindrome) 판별하기 .....	95
[H10] 문자열 분석하기 .....	96
[I01] 메뉴판 보여주는 함수 만들기 .....	103
[I02] 빈칸과 함께 특정 문자를 개수만큼 찍는 함수 만들기 .....	104
[I03] 비만 판정 .....	105

[I04] 메뉴 번호 받아오는 함수 만들기 .....	106
[I05] 최댓값 리턴하는 함수 만들기 .....	107
[I06] 임의의 숫자를 만들어 구간을 리턴하는 함수 만들기 .....	108
[I07] 자판기에서 선택한 음료 가격을 리턴하는 함수 만들기 .....	109
[J01] 나이 계산 및 연령대 판정 .....	113
[J02] 심사점수 계산 .....	114
[J03] 물의 온도 구간 개수 판정 .....	115
[J04] 연중 날짜 계산 함수를 이용한 날짜 간격 세기 .....	116
[J05] 주차 관리 시스템 .....	117
[J06] 피보나치 수 구하기 .....	118
[J07] 2의 제곱수 구하기 .....	119
[J08] Ackermann 수 구하기 .....	120
[J09] pow() 함수 만들기 .....	121
[J10] 피보나치 수열로 황금비율 구하기 .....	122
[K01] 메뉴판 저장하기 .....	126
[K02] 메뉴판 읽어오기 .....	127
[K03] 좌표 저장하기 .....	128
[K04] 좌표 불러오기 .....	130
[K05] 사용자 목록 저장하기 .....	131
[K06] 사용자 목록 불러오기 .....	133
[K07] 학생 점수 결과 저장하기 .....	134