4. Function (함수)

Initial Commit: 2020.08.07
Private lesson (OOP Basics with Python, August 2020)
Reference: https://wikidocs.net/24
※ 문제의 난이도는 순차적으로 올라갑니다. 좌절하지 말고 답안과 비교하면서 공부하기 바랍니다 😃
※ 파라미터나 변수의 이름이 문제에 명시되지 않은 경우 임의로 설정해야 합니다!

(1-1) 정수를 입력받아 n 변수에 저장하고, 파라미터 x에 대해 x번만큼 Hello, World!를 출력하는 함수 rep를 정의하여 rep(n)을 호출하세요. rep의 return값은 없습니다.

(1-2) 1-1번 문제에 대한 추가 도전 과제입니다. rep 함수의 파라미터를 x가 아닌 n으로 바꾸어 실행하고, 그 결과를 1-1번의 결과와 비교해보세요.

> **[실행 예시]**
> Insert a number: 5
> Hello, World!
> Hello, World!
> Hello, World!
> Hello, World!
> Hello, World!

(2) 정수를 입력받아 a 변수에 저장하고, 파라미터의 세제곱을 return하는 함수 f를 정의하여 f(a)를 호출한 결과를 출력하세요.

> **[실행 예시]**
> Insert a number: 5
> 125

(3) 국어, 수학, 영어 점수를 정수로 입력받아 각각 korean, math, english 변수에 저장하고, 점수의 평균을 return하는 함수 avg와 해당 점수의 등급을 return하는 함수 cal을 정의하여, 세 점수에 의한 등급을 출력하세요.
(평균이 90점 이상이면 A등급, 80점 이상이면 B등급, 70점 이상이면 C등급, 그 외의 경우 D등급입니다.)

> Korean score: 95
> Math score: 80
> English score: 80
> B

(4) 두 정수를 입력받아 각각 a, b 변수에 저장하고, 두 파라미터의 최대공약수를 return하는 함수 g를 정의하여 g(a, b)를 호출한 결과를 출력하세요.

> **[실행 예시]**
> Insert number a: 12
> Insert number b: 18
> 6

(5-1) 양의 정수 n에 대한 팩토리얼은 n보다 작거나 같은 모든 양의 정수의 곱으로 정의되며, ! (느낌표)로 표시합니다.
(Ex. 6! = 6x5x4x3x2x1=720)
정수를 입력받아 n 변수에 저장하고, 파라미터의 팩토리얼을 return하는 facto 함수를 정의하여 facto(n)을 호출한 결과를 출력하세요.

(5-2) 5-1번 문제에 대한 추가 도전 과제입니다. facto 함수를 **재귀함수**로 정의하여 동일한 결과를 얻으세요.

> **[실행 예시]**
> Insert a number: 4
> 24

(6-1) 피보나치 수열은 일반적으로 1, 2, 3, 5, 8, 13... 과 같이 n번째 원소가 (n-1)번째 원소와 (n-2)번째 원소의 합인 수열으로 정의됩니다.
정수를 입력받아 n 변수에 저장하고, 피보나치 수열의 (파라미터)번째 원소를 return하는 fibo 함수를 **재귀함수**로 정의하여 fibo(n)을 호출한 결과를 출력하세요.

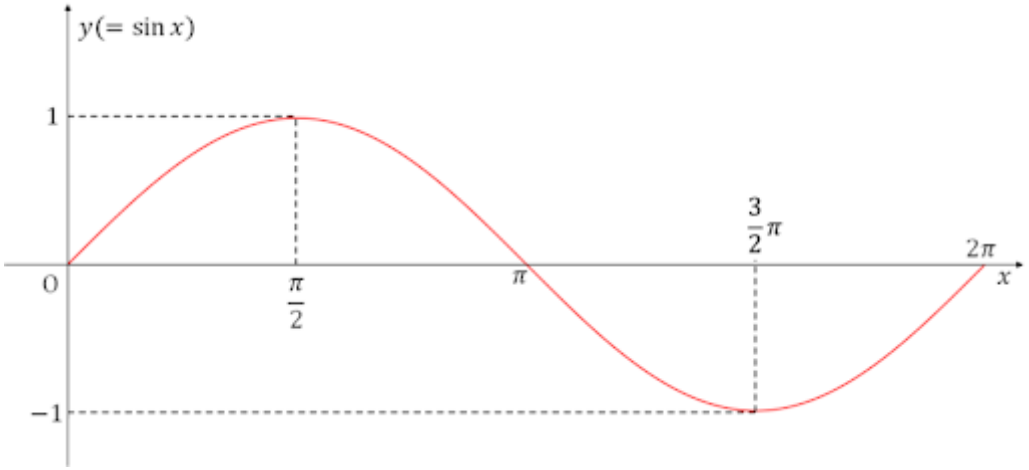
(6-2) 6-1번 문제에 대한 추가 도전 과제입니다. 재귀함수로 정의된 fibo 함수는 파라미터가 큰 값 (50 이상)일 때 처리 시간이 유의미하게 증가합니다. fibo 함수의 시간 복잡도를 계산해보고, 처리 시간을 감소시킬 수 있는 아이디어를 생각해보세요.

> **[실행 예시]**
> Insert a number: 7
> 21

(7) 양의 정수를 입력받아 n 변수에 저장하고, n에 대한 다음 문제의 답을 return하는 함수 stair을 정의하여 호출한 결과를 출력하세요.
"n개의 칸을 가지는 계단이 있습니다. 당신은 계단을 한 번에 1칸 또는 2칸씩 오를 수 있습니다. 계단을 모두 오르는 경우의 수를 구하세요."
예를 들어 3칸의 계단을 오르는 경우의 수는 (1, 1, 1), (1, 2), (2, 1)로 3이고, 4칸의 계단을 오르는 경우의 수는 (1, 1, 1, 1), (2, 1, 1), (1, 2, 1), (1, 1, 2), (2, 2)로 5입니다.

> **[실행 예시]**
> Insert a number: 10
> 89

(Ex) 직각이 아닌 한 내각의 크기가 x인 직각삼각형에 대한 빗변의 길이와 높이의 비로 정의되는 함수 sin x (사인 x)는 물리학에서 물체의 원형 궤도 운동을 수치적으로 분석하기 위해 필수적이므로 그 값을 정밀하게 알아내는 것이 매우 중요합니다. 아래 그림은 y = sin x의 그래프입니다.



그러나 공학용 계산기나 수치해석 프로그램은 물리적으로 sin x의 값을 정확하게 알 수 없으므로 '최대한 비슷하게' 결과를 얻기 위한 방법을 사용하는데요, 그 수학적인 기반은 테일러 급수에서 비롯됩니다. 테일러 급수는 analytic한 임의의 함수를 무한개의 항을 가지는 다항식 (무한급수)의 형태로 나타낼 수 있음을 보장해줍니다. 아래 식은 sin x를 포함한 몇 가지 함수에 대한 테일러 급수입니다.

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n}$$

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1}$$

sin x의 테일러 급수를 살펴보면, 1항은 x, 2항은 -x^3/(3!)...으로 특정한 규칙을 가진 무한개의 항으로 이루어져 있음을 알 수 있습니다. 무한개의 항을 더하는 것은 상식적으로 어려우므로 컴퓨터는 임의의 정수 n을 지정하여 테일러 급수의 n개의 항 만을 계산해 그 결과를 출력합니다. 이와 같이 테일러 급수의 첫 n개의 항만을 더한 식을 n차 테일러 다항식이라고 합니다. 예를 들어서 sin x의 1차 테일러 다항식은 x와 같고, 2차 테일러 다항식은 x - x^3/(3!)으로 정의될 것입니다. 일반적으로 n이 증가할수록 정확도는 증가합니다.
두 정수를 입력받아 변수 n, x에 저장하고, x에 대해 sin x의 n차 테일러 다항식을 계산한 값을 return하는 함수 sine을 **재귀함수로** 정의하여 sine(n, x)를 호출한 결과를 출력하세요. (Hint: 계산을 용이하게 만들기 위해 함수를 추가로 선언하고 이용하는 것을 추천합니다 😃)

> **[실행 예시 1]**
> Determine the degree of sin calculation (n): 3

Insert a number (x): 1
0.8416666666666667

[실행 예시 2]
Determine the degree of sin calculation (n): 7
Insert a number (x): 1
0.8414709848086585