# comment

자료형: 숫자형 (정수 integer, 소수floating point) / 문자형string (따옴표) / 불린 (True or False)

문자열: 연결 (문자열끼리만 연결 가능)

숫자형: 계산

추상화: 변수 (x,y값 저장) / 함수 (명령 저장)

p="Family" # 변수를 ‘정의’한 것

q="Custume"

print(p+q) # 여긴 따옴표가 안 들어가도 FamilyCustume으로 나옴

=은 수학처럼 양쪽이 같다는 의미아님. 오른쪽을 왼쪽에 지정해준다는 의미. 지정 연산자

Cf. == m 수학적으로 등호 의미

Print는 이미 함수.

우리가 함수를 정의하려면

Def Hello():

여기에 Hello 함수 정의

아 그리고 print 쓰는 거 잊지 않기

함수에 변화주기 파라미터

Def Hello(name):

Print(name)  
Hello(“Minjee”)

실행: Minjee

Def 쓰는 거 잊지 말고

파라미터 여러 개 쓰려면 그냥 수학에서 하나의 변수 더하듯이

Def Hello(x, y, z):

Print(x+y+z)

Hello(2, 3, 4)

이렇게만 해도 출력됨!

Hello(“가”, “나”, “다”)

문자열은 문자열로!

return: 말그대로 수학에d서의 함수 정의인데? + 함수를 즉시 끝내기 (리턴 후의 코드는 dead code)

print(x \* y)랑 return x \* y랑 똑같지 않음. 그냥 Return 함수는 콘솔에 아무것도 써지지 않음. 그냥 컴퓨터 내에서 값이 실행되긴만 할 뿐.. 그래서 print라고 해줘야 출력됨.

Return은 print처럼 밖에다 쓸 수 없음.. 함수 정의할 때는 쓸 수 있음..

1, print(print(“Hello”’) 실행: Hello / None

print ( 프린트함수(x)) ) 하면 프린트함수가 출력되고 print (None) 이 됨. 그럼 None이 출력됨.

2, print(return Hello) 실행: Hello

3, print(None) 실행: None

Def Hello(x, y, z):

Print(x+y+z)

Return “Hi”

Print(Hello(“가”, “나”, “다”))

x+y+z만 출력되고 None출력 안 됨. (return이 뒤에 있어서 그런듯.. 확실하지 않음)

둘째줄은 print함수에 들어가서 출력됨.

나머지 구하기: % 거듭제곱: 2 \*\* 3 은 2의 세제곱

나눗셈은 무조건 float 소수형이 무조건 세기 때문에 개수 새려면 //쓰거나 int로 감싸기

Floor division 버림나눗셈 // : 소수부분 버림, 소수형 포함하면 버림나눗셈 결과값이 소수형으로..

Round 반올림: 소수점 n째 자리면 (3.141592, n) 이렇게 두 번째 파라미터로 넣어주기

Ex) 4/2 float 4//2 int 4//2.0 float

Print(type(3)) 하면 무슨 형인지 나옴

문자열 곱셈 “Hello” \* 3

글자 내에 따옴표 포함되면 겹치는 모든 따옴표 앞에 역슬래시: \

**형 변환 type conversion**

문자열을 숫자형으로도 가능

Int float str 으로 감싸주기

**문자열 format**

: 문자열에 빈칸을 중괄호로 나타내기: ‘나는 {}번째 순서이다’. format(2)

중괄호 빈칸에 숫자, 문자열 다 가능

X, y 순서를 바꾸고 싶다면 {} 안에 0, 1, 2 … 순서를 적어주면 됨

{}에 순서를 넣어줄 때 하나에만 2 이렇게 넣으면 ok인데 하나에만 2:.2f이러면 안됨..

2:.2f means 순서2의 floating point가 소수점에서 2번째 자리까지 나오도록.

Cf. round(7.2534, 2) 실행 7.25

Round 쓸려면 .format 쓸 때 round 넣기

Formatting 다른 방법은 f”나는 {}이다.” 로 써도됨.

**Bool number** (and, or, not) True / False

숫자 비교할 때 수학적으로 같다. ==

!= means 같지 않다(다르다)

옵셔널 파라미터는 항상 마지막에.

Syntactic sugar: x=x+1 can be also said x+=1

X=x\*2 can be also said x\*=2

**변수 범위란**

Scope: 범위

Local 변수: 그 함수 내에서 정의된 변수. 파라미터도. <-> global 변수: 함수 외

함수 내외 다른 값으로 지정되었으면 함수내에서는 가장 최근에 지정된 함수 내 값을 출력. 그러나 함수 밖에서는 글로벌 변수로. 로컬변수부터 확인하고 없으면 글로벌변수.

변수 중에 바뀌지 않는 것은 상수 constant라고 함. 모든 글자를 대문자로 쓰기. 보는 입장에서 구분짓기 쉬우니까. 실행상에서는 상수를 바꿔도 바꾼대로 출력됨.

Pep8에 나와있는 스타일 규칙임.

들여쓰기는 스페이스 4개, 함수 정의 사이는 빈 줄 2개, 괄호 옆 띄어쓰기X

지정연산자 앞뒤 띄기 but 우선순위 강조를 위해선 붙이기, 코멘트 앞 스페이스 2개 이상

Return False If ~~ else False

그런데 그냥 2==0 자체가 False인 불린이니까 False로 나옴.

If elif elif elif elif else 하면 여러가지 경우의 수 중 하나만 가능

**<While 반복문>**

조건부분(불린) 그리고 콜론: 쓰고

실행부분은 들여쓰기

몇 번 실행됐는지 순서매기려면 실행될 때마다 값이 수정되도록

1~100 까지 자연수 중 몇 배수만 출력하려면 I = 1 로 시작하고, I += 1은 if문의 ox와 연관없이 한 번마다 무조건 실행돼야하므로 if문 밖에 쓰기!

While문 안에서 반복되는 거임.

함수의 마지막 값을 다른 데에 쓰고 싶으면 함수 마지막에 값 리턴해주기.

단, 함수 아니고 그냥 while문으로 변수를 변화시켰다면 그냥 그 값 써주면 됨.

함수 내 로컬 변수 있고 while문 있어서 값이 변한 수에 리턴해주면 변한 마지막 값을 리턴.

{0:.0f} 항상 잊어버림… 0번부터 그리고 땡땡해주고 소수점 찍고 몇 번째 자리까지 floating이다.

천 단위로 나눌 때 \_ 써도 됨.

아님 .format(int(blabla)) 해도 됨.

While문에서 그냥 갑자기 그만하고 싶다 그럼 break문

While문 뒤에 print라던지 무시하고 다음 반복으로 넘어가고 싶다면 continue문

<**프로그래밍과 데이터(3)>**

**리스트: 변수에 대괄호로 리스트를 저장할 수 있음**

문자열은 각 단어마다 따옴표해주기

출력하면 리스트 그대로 출력됨.

인덱싱: 리스트의 몇 번째 인덱스를 데려오는 것

Names = [‘alegra’, ‘Minji’, ‘gigi’]

Print(Names[1]) 실행: Minji

첫 번째가 인덱스 0

**음수 인덱스**는 거꾸로 감. -1은 gigi -2는 Minji

그러나 개수 이상 양수나 음수로는 못감

**리스트 슬라이싱**: 리스트 구간으로 자르기

names[2:] 인덱스 2부터 끝까지

Names[0:2] 인덱스 0부터 1까지 (마지막 인덱스는 하나 전까지)

**인덱스에 저장된 요소 바꾸기**

Names[0] = ‘Hana’ 당근 지정연산자 사용하면 됨

**인덱스 찾기**

Print(names.index(“Minji”))

**리스트의 요소 개수 찾기**

Len(리스트) # length

**리스트에 요소 추가/삭제하기**

Names.append(jake)하면 jake가 맨 뒤에 추가됨. 추가연산

Names.insert(1, kim)하면 인덱스1이 되도록 추가됨. 삽입연산.

Del names[2]하면 인덱스2가 지워짐

**리스트 정렬하기**

Sorted: 새로운 함수에 작은 순서대로 정렬된 함수 저장

Sort: 기존 함수를 작은 순서대로 정렬만 하고 리턴은 안함.

Reverse=true를 파라미터로 넣으면 반대로 정렬됨

New\_list = sorted(names, reverse=true)

Names.sort(reverse=true)

반대로 정렬하기: Names.reverse()

\*# 값을 변환하고 싶으면 A = A + 1 이 식 사용하기

\*# while문 반복될 때 I += 1 해야하는지 주의

Print(7 in numbers)

True 나오면 7이 numbers 안에 있는 거임

**리스트 인덱스로 원소 프린트하기**

Print(Numbers[i])

Numbers[ [0,1] [2,3] ] 이런 식이면

Print[0]하면 [0,1] 나오고

Print[0][1]하면 1나옴

**<for문>**

For문은 while문에서 리스트의 요소들이 하나씩 순서대로 변수에 저장되는 것

For number in range():

Print number

Range(start, stop) : range(3,10)은 3~9까지 의미

Range(stop) : range(10)은 0~9까지 의미

Range(start, stop, step) : range(3, 17, 3)하면 3,6,9,12,15로 나옴. Start부터 3씩 커짐, stop은 안나옴

**리스트 뒤집기**

numbers[right], numbers[left] = numbers[left], numbers[right]

**<dictionary>**

리스트랑 비슷한데 중괄호 사용하고, 리스트는 0,1,2~ 나가지만 dic은 4: 35, 5: 20 이런 식으로 순서 없이 나타낼수 있음. 리스트랑 똑같이 dic[4] 하면 35임. 그렇게 추가하기.

For key, value in dict.items():

For key in dict.key()