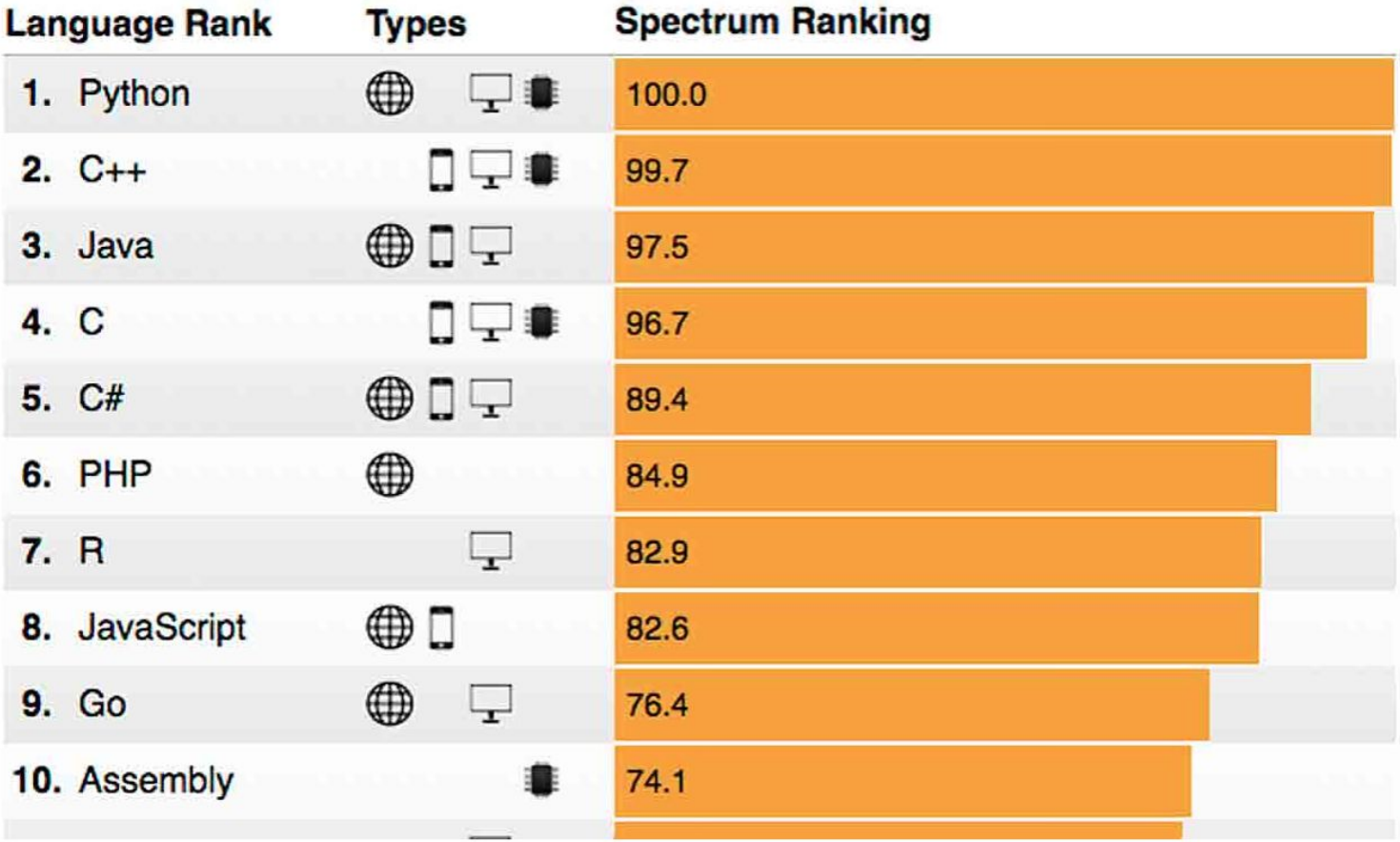
Python programming basic#1

경기과학고등학교 김종혜

TIOBE index

Jan 2019	Jan 2018	Change	Programming Language	Ratings	Change
1	1		Java	16.904%	+2.69%
2	2		C	13.337%	+2.30%
3	4	^	Python	8.294%	+3.62%
4	3	v	C++	8.158%	+2.55%
5	7	^	Visual Basic .NET	6.459%	+3.20%
6	6		JavaScript	3.302%	-0.16%
7	5	v	C#	3.284%	-0.47%
8	9	^	PHP	2.680%	+0.15%
9	-	^^	SQL	2.277%	+2.28%
10	16	^^	Objective-C	1.781%	-0.08%

The 2018 Top Programming Languages(IEEE)

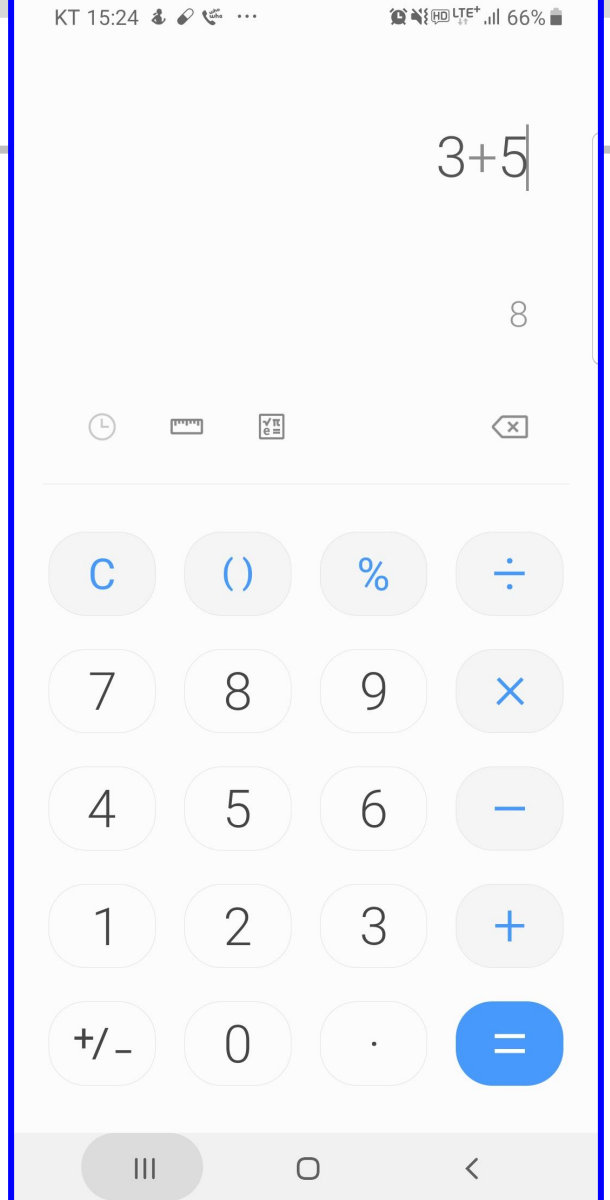


python programming

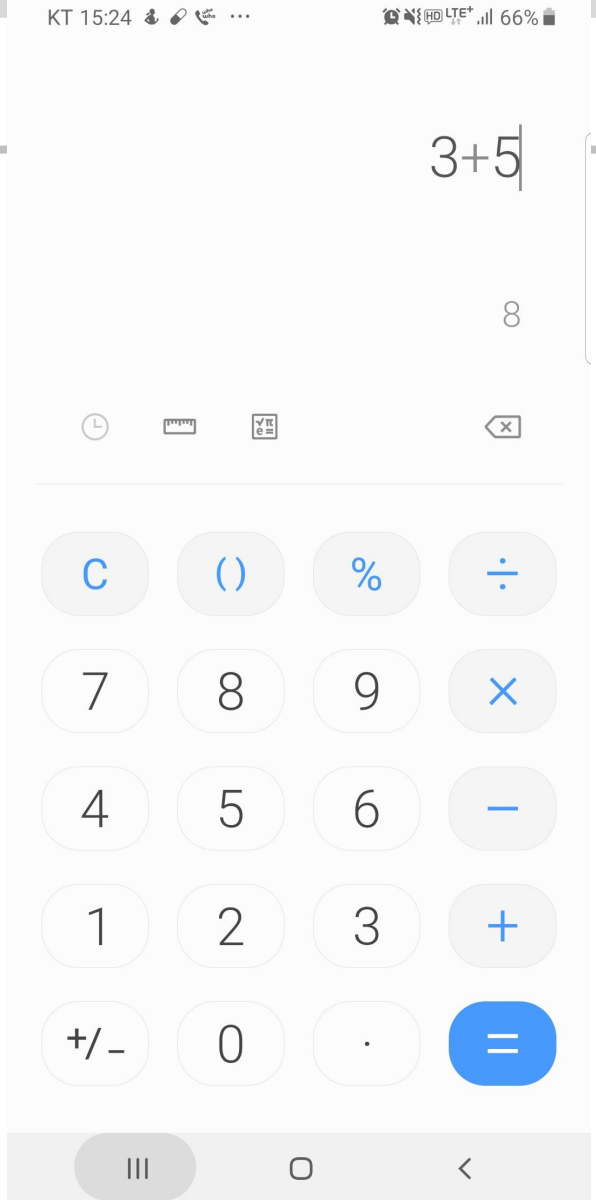
Python

- easy to learn and powerful
- used in many universities for introductory course
- used for web programming at Google.
- Large portions of Games(such as Civilization 4)
- Deep learning

Programming



Programming

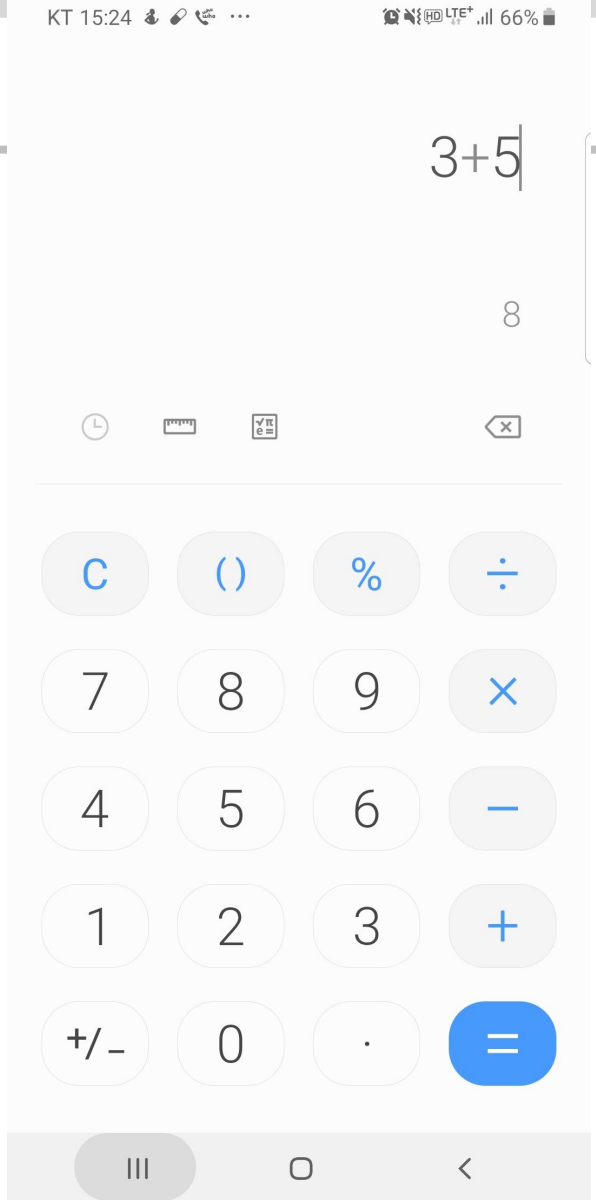



```
a=3
b=5
print(a+b)
```

```
8
```

a, b : 변수(값 저장공간)
print() : 출력함수

Programming




```
a=int(input())
b=int(input())
print(a+b)
```

```
 3
 5
8
```

```
a=3
b=5
print(a+b)
```

```
8
```

a, b : 변수(값 저장공간)
input() : 입력함수
int() : 정수형으로 변환
print() : 출력함수

python programming-1

1. variable(변수) : 값 저장공간
2. data type(데이터형) : 정수, 실수, 문자열 등
3. String(문자열) : “ “, ‘ ‘
4. standard I/O (표준입출력) : 키보드, 모니터
5. operator(연산자) : 산술, 논리, 관계

python programming-1

- variable(변수) : 값 저장공간
- data type(데이터형) : 정수, 실수, 문자열 등
- standard I/O (표준입출력) : 키보드, 모니터
- operator(연산자) : 산술, 논리, 관계

```
a=int(input())
b=int(input())
print(a+b)
```

```
 3
 5
8
```

- variable(변수) :
- data type(데이터형) :
- 표준 입력 :
- 표준 출력 :
- 연산자 :

variable

- 변수 : 값 저장공간
- 변수를 미리
  선언하지 않음

- **변수의 데이터형은**
  대입되는 자료에
  따라 **자동으로**
  결정됨

```
>>> a=10
>>> type(a)
<class 'int'>
>>> a=10.5
>>> type(a)
<class 'float'>
>>> a='hello'
>>> type(a)
<class 'str'>
>>> a='h'
>>> type(a)
<class 'str'>
```

variable

- 변수 여러개를 한꺼번에 만들 수 있음

```
>>>x,y,z=10,20,30
>>>x
10
>>>y
20
>>>z
30
```

```
>>>x=y=z=10
>>>x
10
>>>y
10
>>>z
10
```

variable

- 변수 여러 개에 변수를 각각 할당 가능
- 두 변수의 값을 바꿀 때, 콤마(,)로 처리 가능

```
>>>a,b=10,20
>>>x,y=a,b
>>>x
10
>>>y
20
```

```
>>>x=10
>>>y=20
>>>x,y=y,x
>>>x
20
>>>y
10
```

String

' '(작은 따옴표) 또는
" "(큰 따옴표)로 묶어서 표현

\+ : 문자열 연결
* : 문자열 반복

len() : 문자열 길이 구하기

```
>>> str1='Hello World'
>>> str2="Hello World"
>>>
>>> 'Hello'+'World'
'HelloWorld'
>>> 'Hello'*3
'HelloHelloHello'
>>>
>>> str1+str2
'Hello WorldHello World'
>>>
>>> len('hello world')
11
```

String

String + 정수/실수 => error!

```
>>> "Hello" + 10
Traceback (most recent call last):
  File "<pyshell#1>", line 1, in <module>
    "Hello" + 10
TypeError: must be str, not int
>>> "Hello" + str(10)
'Hello10'
>>>
```

String

```
>>> str="Hello world!"
>>> str
'Hello world!'
>>> str[3]
'l'
>>> str[1:5]
'ello'
>>> str[:5]
'Hello'
>>> str[5:]
' world!'
```

0	1	2	3	4	5	6	7	8	9	10	11
H	e	l	l	o		w	o	r	l	d	!

표준 입력(input())

1. 변수 = input() (리턴값 : 문자열)

2. 한번에 여러개 입력받기 (리턴값 : 문자열)
   - 변수1, 변수2 = input().split() # spacebar 구분
   - 변수1, 변수2 = input().split('기준문자')
   - 변수1, 변수2 = input('문자열').split()
   - 변수1, 변수2 = input('문자열').split('기준문자')

표준 입력(input())

3. 한 번에 여러개 입력받기

○ map() 함수를 이용하면 입력된 데이터를 모두 정수/실수로 변환해줌 (정수 : int, 실수 : float)

- 변수1, 변수2 = map(int, input().split())
- 변수1, 변수2 = map(int, input().split('기준문자'))
- 변수1, 변수2 = map(int, input('문자열').split())
- 변수1, 변수2 = map(int, input('문자열').split('기준문자'))

표준 입력(input())

```
>>> a=input()
10
>>> a
'10'
>>> type(a)
<class 'str'>
>>> b,c=input().split()
10 20
>>> b
'10'
>>>
>>> c
'20'
>>> d,e=input().split(",")
30,40
>>> d
'30'
>>> e
'40'
```

```
>>> a,b=map(int,input().split())
10 20
>>> a
10
>>> b
20
>>> type(a)
<class 'int'>
>>> type(b)
<class 'int'>
```

표준 출력(print())

print(변수)	- 엔터를 포함해 출력됨
print(값1, 값2, 값3) print(변수1, 변수2, 변수3)	- 변수나 값을 콤마(,)로 구분해서 넣으면 각 값이 공백으로 띄어져서 한 줄로 출력됨.
print(변수,end="")	- 엔터를 삭제하고 출력됨
print(변수1, 변수2, sep="문자")	- 여러개의 값 사이에 공백이 아닌 다른 문자를 넣고

표준 출력(print())

```
>>> print(1,2,3)
1 2 3
>>> print(5,6,sep=",")
5,6
>>> print(10,20,sep="*")
10*20
```

표준 출력(print())

print('%d' % n)	변수n의 값을 10진수로 출력
print('%.2f' % f)	변수f의 값을 소수점 이하 셋째 자리에서 반올림하여 둘째 자리까지 출력
print('%o' % n)	변수n의 값을 8진수로 출력
print('%x' % n)	변수n의 값을 16진수 소문자로 출력
print('%X' % n)	변수n의 값을 16진수 대문자로 출력

표준 입출력

CodeUp # 1038

- 정수 2개를 입력받아 합을 출력하시오.

```
1  a,b=input().split()
2  x=int(a)
3  y=int(b)
4  print(x+y)
```

```
1  a,b=map(int,input().split())
2  print("%d"%(a+b))
```

operator

산술연산자	+(더하기), -(빼기), *(곱하기), **(거듭제곱), /(나누기), //(몫), %(나머지)
비교연산자 True / False	>, <, >=, <=, ==(값의 같음), !=(값의 다름), is(객체의 같음 주소 비교), is not(객체의 다름 주소 비교)
논리연산자 True / False	and, or, not

operator _ 산술연산자

```
>>> 3+5
8
>>> 5/3
1.6666666666666667
>>> 5//3
1
>>> 5%3
2
>>> 5**3
125
```

- python은 실수를 부동소수점으로 표현함.
- 부동소수점 반올림 오차 (rounding error)
  - 실수를 근삿값으로 표현하면서 발생하는 문제

```
>>> 0.1+0.2
0.30000000000000004
>>> 0.1+0.2 == 0.3
False
```

operator

할당연산자를 사용할 때

값이 저장될 변수에는 값이 반드시 들어있어야 함.

```
>>> a=10
>>> b=20
>>> c+=b
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
    c+=b
NameError: name 'c' is not defined
>>>
```

operator _ 비교 연산자

- 비교연산자 결과 : True / False
- >, <, >=, <=,
- ==(값의 같음)
- !=(값의 다름)

```
>>> a=10
>>> b=20
>>> type(a)
<class 'int'>
>>> type(b)
<class 'int'>
>>> a==b
False
>>> a!=b
True
```

operator _ 관계 연산자

- 관계 연산자 결과 : True / False

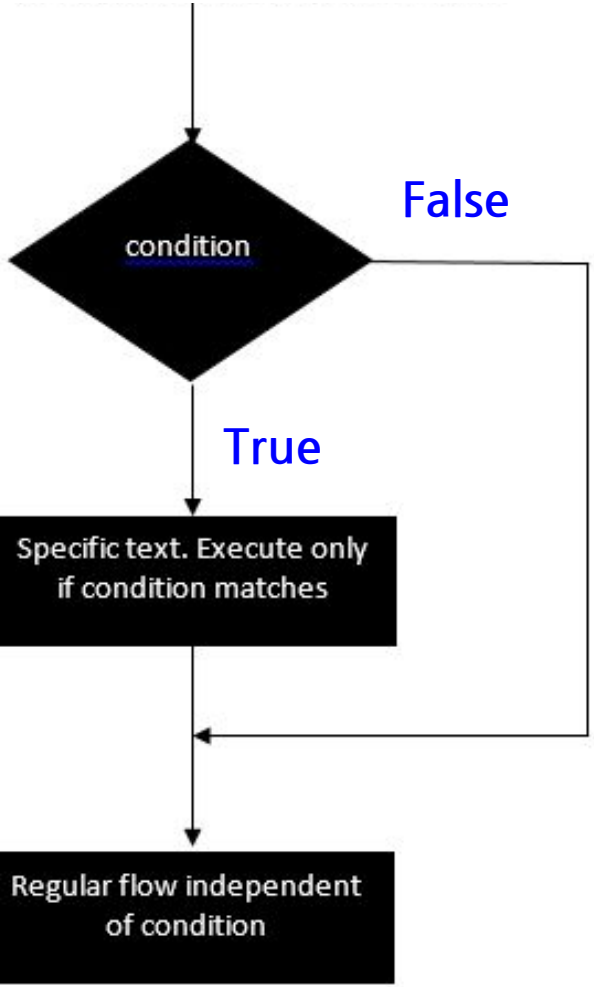
- and
- or
- not

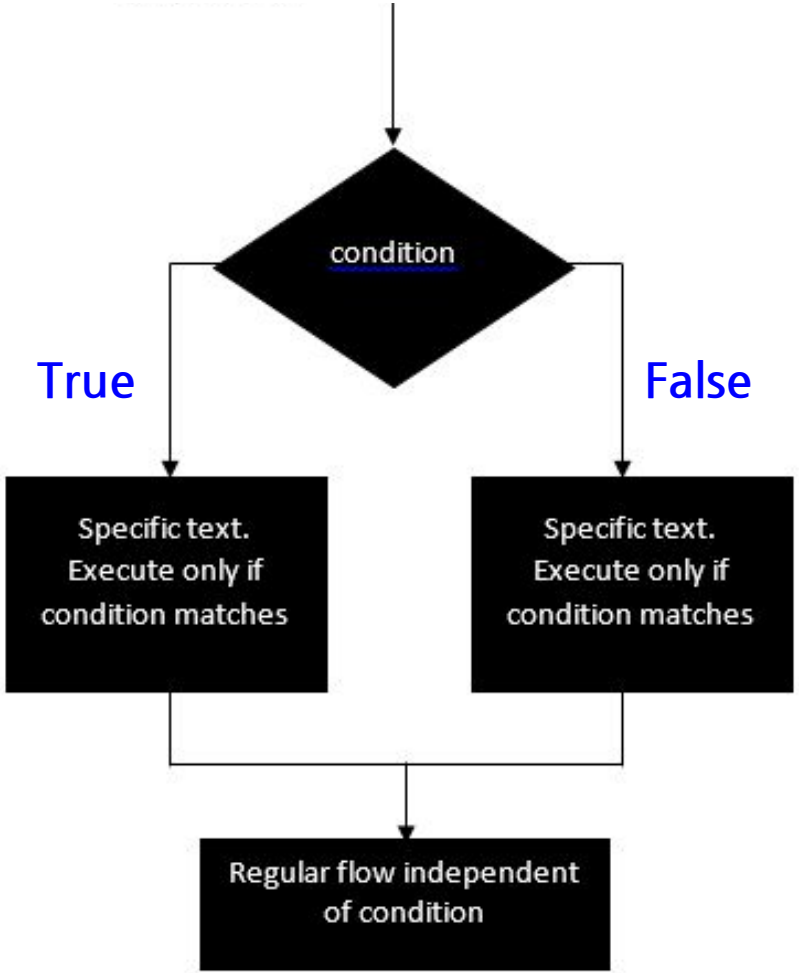
```
>>> a=10
>>> b=20
>>> c=10
>>> a==b or a==c
True
>>> a==b and a==c
False
>>> not(a==b) and a==c
True
```

python programming-2

1. condition(조건, 판단)
2. repeat (반복)

condition


condition
False
True
Specific text. Execute only if condition matches
Regular flow independent of condition


condition
True
False
Specific text. Execute only if condition matches
Specific text. Execute only if condition matches
Regular flow independent of condition

condition

```
if 조건식1:
    문장1
elif 조건식2:
    문장2
.....
else:
    문장n
```

CodeUp #1069

```
1  a=input()
2
3  if a=="D" :
4      print("slowly~")
5  elif a=="C" :
6      print("run!")
7  elif a=="B" :
8      print("good!!")
9  elif a=="A" :
10     print("best!!!")
11 else :
12     print("what?")
```

pass
- if 문에 아무 코드도 넣지 않으면 에러가 발생하므로, 아무일도 하지 않고 그냥 넘어가는 의미로 사용

```
if x==1:
    pass
```

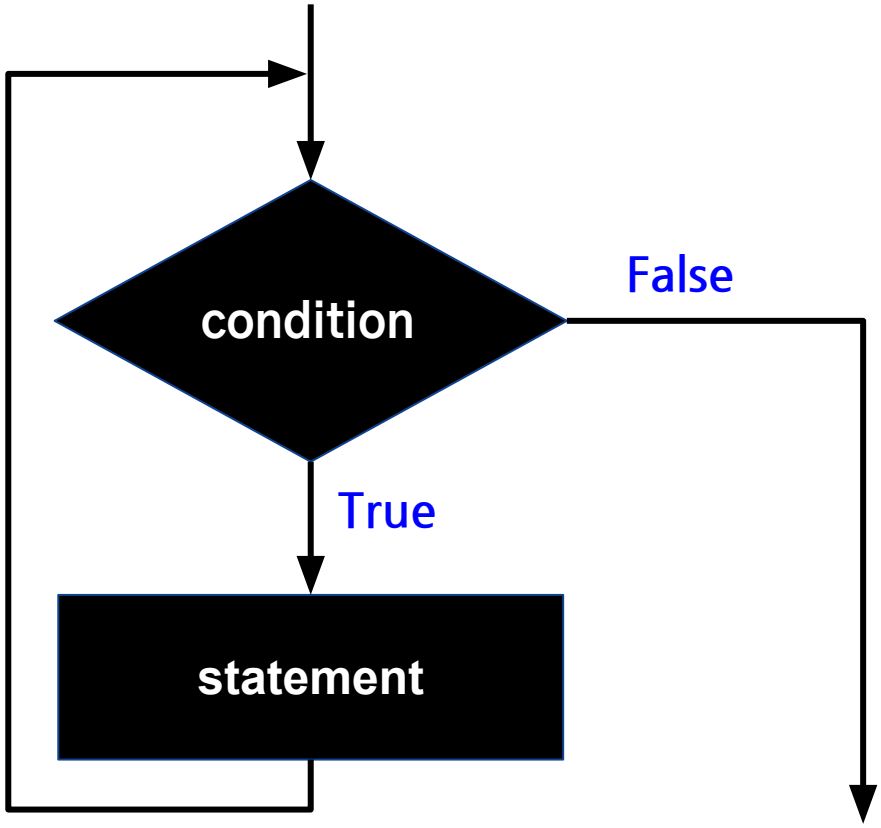
condition

if-else 문 축약 가능함

변수 = 값 if 조건문 else 값

```
n=int(input())
print(“even” if n%2==0 else “odd”)
```

repeat



repeat _ while

초기값

while 조건:

    실행문

    증감식

CodeUp #1074

```
1  n=int(input())
2  while n:
3   print("%d"%n)
4   n-=1;
```

repeat _ for

```
for 변수 in range(횟수):
    실행문
```

- **for 변수 in range(초기값, 최종값, 증감):**
  - **초기값 ~ (최종값-1) 까지 증감 반복**

- **for i in range(n) :**
  - 변수i의 값이 0부터 n-1까지 n번 반복

- **for i in range(1, 3)**
  - 변수i의 값이 1부터 2까지 두 번 반복

- **for i in range(1, 10, 2) :**
  - 변수i의 값이 1부터 9까지 2씩 증가(1,3,5,7,9)하여 5번

repeat _ for

```
>>> for i in range(5):
        print("Hello",i)


Hello 0
Hello 1
Hello 2
Hello 3
Hello 4
```

```
range(5)
= range(0, 5)
= range(0, 5, 1)
```

```
>>> for i in range(5,0,-1):
        print("Hello",i)


Hello 5
Hello 4
Hello 3
Hello 2
Hello 1
```

range(5,0,-1) : -1은 숫자 감소할때

```
>>> for i in range(5,0):
        print("Hello",i)
```

증감 기본값이 +1 증가값이므로, range(5,0)은 동작하지 않음.

repeat _ for

- 파이썬은 입력방식에 따라 소요시간이 달라질 수 있음.

- 모든 자료를 입력받아 리스트에 저장후 사용하면 3초안에 해결할 수 있지만, 리스트를 만들지 않고 하나씩 입력받아 사용하면 주어진 시간안에 문제를

시간안에 해결	시간초과

for i in range(m):
  a[i] = int(input())
 	
for i in range(m):
  print(“%d”%f(a[i])
	for i in range(m):
	  k=int(input())
	  print(“%d”%f(k))

repeat _ for

1 ~ n(입력정수)까지의 합 구하기

```
n=int(input())
total=0
for

print(total)
```

```
 100
5050
```

repeat _ for

숫자 삼각형 출력하기

```
n=int(input())
for i in range(      ):
    for j in range(      ):
        print(j, end=" ")
    print()
```

```
 5
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

python programming-3

1. list (리스트)
2. function (함수)
3. module (모듈)
4. recursive function (재귀함수)

list

- 여러 요소들을 갖는 집합(Collection)
- 새로운 요소를 추가, 갱신, 삭제하는 일이 가능함.
- 파이썬의 리스트는 **동적배열(Dynamic Array)**로서 자유롭게 확장할 수 있는 구조임.
- 모든 자료형(문자열, 정수, 실수, 불 등)을 저장할 수 있으며, 자료형을 섞어서 저장해도 됨.
- 대괄호([ ])로 감싸고, 안에 들어갈 값들을 쉼표로 구분해서 만든다.

- 1차원 리스트 a=[1,2,3] x=[]
- 2차원 리스트 b=[[1,2,3],[4,5,6]]

list

```
>>> a=list(range(10))
>>> a
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> b=list(range(10,0,-1))
>>> b
[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
>>> c=list(range(1,10,3))
>>> c
[1, 4, 7]
>>> d=[]
>>> d
[]
>>> e=list()
>>> e
[]
```

list

1차원 리스트 생성 및 초기화

```
>>> b=[0]*10
>>> b
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

>>> b=[0 for i in range(10)]
```

list

2차원 리스트 초기화

```
>>> a=[]
>>> for i in range(3):
        a.append([])
        for j in range(2):
            a[i].append(0)

>>> print(a)
[[0, 0], [0, 0], [0, 0]]
```

list

```
>>> a=[[0]*2 for i in range(3)]
>>> print(a)
[[0, 0], [0, 0], [0, 0]]
```

- 리스트 * 정수
  - 특정횟수(정수값)만큼 리스트의 요소를 반복함
  - [0]*2 => [0,0]

list

- **리스트명.append(x)**
  - 리스트 맨 마지막에 x 요소 삽입
- **리스트명.insert(x,y)**
  - x번째 위치에 y를 삽입
- **리스트명.remove(x)**
  - 리스트에서 첫번째 나오는 x 데이터 삭제
- **리스트명.pop()**
  - 리스트의 맨 마지막 요소를 꺼내고, 마지막 요소는 삭제

list

- **리스트명.count(x)**
  - 리스트안에 있는 특정 값 개수 세기
- **리스트명.sort()**
  - 리스트 데이터 정렬(오름차순)
- **리스트명.reverse()**
  - 리스트 항목의 순서를 반대로 바꾸기

list

```
>>> a=[10,30,20,40]
>>> a.append(100)
>>> a
[10, 30, 20, 40, 100]
>>> a.insert(3,30)
>>> a
[10, 30, 20, 30, 40, 100]
>>> a.count(30)
2
>>> a.sort()
>>> a
[10, 20, 30, 30, 40, 100]
>>> a.append(20)
>>> a
[10, 20, 30, 30, 40, 100, 20]
>>> a.reverse()
>>> a
[20, 100, 40, 30, 30, 20, 10]
>>>
```

- **리스트명.append(x)**
  - 리스트 맨 마지막에 x 요소 삽입
- **리스트명.insert(x,y)**
  - x번째 위치에 y를 삽입
- **리스트명.count(x)**
  - 리스트안에 있는 특정 값 개수 세기
- **리스트명.sort()**
  - 리스트 데이터 정렬(오름차순)
- **리스트명.reverse()**
  - 리스트 항목의 순서를 반대로

```
>>> a=[20,100,40,30,30,20,10]
>>> a.pop()
10
>>> a.pop()
20
>>> a
[20, 100, 40, 30, 30]
>>> a.index(30)
3
>>> a.count(30)
2
>>> len(a)
5
>>> a[len(a):]=[90]
>>> a
[20, 100, 40, 30, 30, 90]
>>> del a[1]
>>> a
[20, 40, 30, 30, 90]
>>> del a[2:4]
>>> a
[20, 40, 90]
>>>
```

- **리스트명.pop()**
  - 리스트의 맨 마지막 요소를 꺼내고, 마지막 요소는 삭제
- **리스트명.index(x)**
  - 리스트에서 특정 값의 인덱스를 구함
- **리스트명.count(x)**
  - 리스트안에 있는 특정 값 개수 세기
- **len(리스트명)**
  - 리스트안의 값 개수 구하기
- **리스트명[len(리스트명):]**
  - 시작인덱스를 len(a)인 리스트의 맨 마지막부터 시작하겠다는 의미
  - a[len(a):]=[90]
    a.append(90)과 같은 의미
- **del 리스트명[시작인덱스:마지막인덱스]**
  - 리스트의 (시작인덱스~ 마지막인덱스-1) 까지의 값을 삭제함

list _ list 연산

- 리스트 + 리스트
  - 리스트를 서로 연결

```
>>> a=[10,20,30]
>>> b=[40,50,60]
>>> a+b
[10, 20, 30, 40, 50, 60]
>>> a*3
[10, 20, 30, 10, 20, 30, 10, 20, 30]
```

- 리스트 * 정수
  - 정수값만큼 리스트의 요소를 반복함
  - a=[0]*3
    - a=[0,0,0]

```
>>> c=[0]*3
>>> for i in range(3):
        c[i]=a[i]+b[i]


>>> c
[50, 70, 90]
```

list _ repeat

len(리스트명) 이용

```
1  a=[10,20,30,50]
2  i=0
3  while i<len(a):
4      print(a[i])
5      i+=1
```

```
10
20
30
50
```

function

- built-in function
  - max(), min(), abs(), input(), print(), int(), float(), len()..
- user-defined function
  - def 키워드로 함수 정의함
  - return-type이 없음

def 함수명(): 명령문	함수명();

def 함수명(매개변수):
    명령문
    return 반환값
	변수 = 함수명(매개변수)

function

함수 실행 결과를 호출한 곳에서 받지 않음

(return 없음)

```
def f(a,b):
    print(a+b)

f(3,4)
```

```
7
```

function

함수 실행 결과를 호출한 곳에서 받지 않음

(return 없음)

```
def f(a,b):
    print(a+b)

f(3,4)
```

```
7
```

함수 실행 결과를 호출한 곳에서 받음

(return 있음)

```
def f(a,b):
    return a+b
print(f(3,4))
```

```
7
```

```
def f(a,b):
    return a+b
x=f(3,4)
print(x)
```

```
7
```

module

- 서로 연관된 작업을 하는 코드들의 모임
- 하나의 파일로 되어 있음

- **표준모듈** : 파이썬 패키지 안에 포함
- **사용자 생성 모듈** : 사용자가 만든 모듈
- **Third Party 모듈** : 협력업체나 개인이 만들어서 제공하는 모듈

```
import 모듈명    # 모듈 불러오기
모듈명.함수()    #모듈안에 저장된 함수 호출
모듈명.변수     #모듈안에 저장된 변수 호출
```

module - 표준 모듈

- math 모듈이름만 갖고왔으므로, math.pi 형식으로 사용해야 함

- from 모듈명 import 함수,변수
  - 모듈의 일부만 가져오기
  - math 모듈에 있는 특정 이름들만 현재 공간으로 가져옴.
- from 모듈명 import *
  - 모듈에 정의된 모든 이름을 현재 공간으로 가져옴.
- import 모듈명 as 새이름
  - 모듈이름을 다른 이름으로 사용하고자 할 때 사용함.

```
>>> import math
>>> pi
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    pi
NameError: name 'pi' is not defined
>>>
>>> math.pi
3.141592653589793
>>>
>>> from math import sin,pi
>>> pi
3.141592653589793
>>>
>>> from math import *
>>> pi
3.141592653589793
>>> sin(4)
-0.7568024953079282
>>>
>>>
>>> import math as m
>>> m.pi
3.141592653589793
>>> math.pi
3.141592653589793
>>>
```

module - 표준 모듈

```
import random
print(random.randint(1,5))  #1~5 정수값
print(random.random())  #0~1미만 실수값
random.shuffle(리스트명)  # 리스트값 무작위 섞기
```

module example

- 1 ~ 45까지 무작위로 섞인 숫자 중 6개의 숫자를 출력하는 프로그램을 작성하시오.

```
before shuffle =  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
after shuffle =  [19, 7, 45, 44, 23, 9, 37, 27, 35, 42, 39, 10, 12, 40, 3, 33, 16, 13,
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
<< output >>
20
18
41
17
2
14
```

recursive function

입력

입력으로 자연수 n이 입력된다. ($1 <= n <= 10,000$)

출력

1부터 n까지의 합을 출력한다.

입력 예시 예시 복사

```
100
```

출력 예시

```
5050
```

CodeUp 1905

```
def f(n):
    if n==1:
        return 1
    else:
        return f(n-1)+n

n=int(input())
print("%d"%f(n))
```

```
import sys
sys.setrecursionlimit(10000)   # recursion function 주의!
```

recursive function

python에서는 재귀 호출 빈도를 제한하고 있음.

sys 모듈을 import해서 재귀 호출 빈도수를 지정해줄 수 있음

```
import sys
sys.setrecursionlimit(100000)
```

```
1000
**********************************************************************************************
  File "/Users/jonghyekim/Documents/koi python/star1.py", line 10, in <module>
    f(n)
  File "/Users/jonghyekim/Documents/koi python/star1.py", line 7, in f
    f(n-1)
  File "/Users/jonghyekim/Documents/koi python/star1.py", line 7, in f
    f(n-1)
  File "/Users/jonghyekim/Documents/koi python/star1.py", line 7, in f
    f(n-1)
  [Previous line repeated 994 more times]
  File "/Users/jonghyekim/Documents/koi python/star1.py", line 4, in f
    if n==0:
RecursionError: maximum recursion depth exceeded in comparison

Process finished with exit code 1
```

실습

CodeUp 회원가입 : https://codeup.kr/

<문제 : 14>

=> 필수문제 : 빨간색 문제

1025, 1027, 1071, 1072, 1083, 1092, 1093

1094, 1095, 2081, 2082, 2621, 4037

실습 hint

```
a,b,c=input().split(".")
print("%s-%s-%s"%(a,b,c))



x,y,z=map(int,input().split())
alist=input().split()
n=len(alist)
x=alist[0]

# and, or, not 사용


# 2차원 리스트 (1096)
n=[ [0]*20 for i in range(20)]
print(n)

## 함수 내 변수 호출
n=10  # 전역변수

def f():
    global n
    n+=1
    print(n)

f()
```

```
# 2차원 리스트 : 1097
a=[]
for i in range(19):
    a.append(input().split())

# 2차원 리스트 출력
for i in range(5):
    for j in range(5):
        print(a[i][j], end=" ")
    print()
```

감사합니다.

경기과학고 김종혜