2023 N.E.T Python

Today

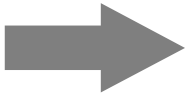
1. Variables
2. Data Type 1
    - Int
    - Float
    - Complex
    - Boolean
    - String
3. Operators
    - Arithmetic
    - Assignment
    - Comparison
    - Logical
    - Identity
    - Membership
4. Conditional Statement
5. Loop
    - For
    - While

Variables

Variable(변수) : 값을 저장하는 장소

변수 선언 방법

```
a = 1
print(a)

a, b, c = 1, 2, 3
print(a, b, c)

a, b = b, a
print(a, b)
```



```
1
1 2 3
2 1
```

Variables

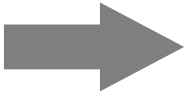
이름 지정 규칙

X		O	
0variable	<- 숫자가 맨 앞	Variable	대소문자 구별 가능 => 변수 이름으로 지정 가능
Print	<- 키워드	variable	
1234	<- 숫자로만 이루어져 있음	max_number	
He llo	<- 띄어쓰기	_hello	
He-llo	<- 하이픈(-)	World_	

Data Type 1

- **int (정수)**
- **float (소수)**
- **complex (복소수)**

```
var1 = 10
print(type(var1))

var2 = 3.14
print(type(var2))

var3 = 1+2j
print(type(var3))
```

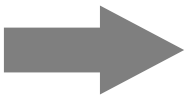


```
<class 'int'>
<class 'float'>
<class 'complex'>
```

Data Type 1

- **bool (T / F)**
- **str (문자열)**

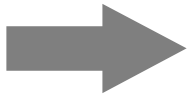
```
var4 = True
print(type(var4))

var5 = 'Hello'
print(type(var5))
```



```
<class 'bool'>
<class 'str'>
```

String

```
str = "Hello World!"
print(str)
print(str[0])
print(str[2:5])
print(str[2])
print(str * 2)
print(str + "TEST")
```



```
Hello World!
H
llo
l
Hello World!Hello World!
Hello World!TEST
```

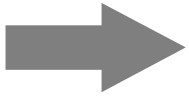
Arithmetic Operator

+	a = 2 result = a + 3 print(result)	2
-	a = 10 result = a - 3 print(result)	7
*	a = 3 result = a * 9 print(result)	27
/	a = 7 result = a / 2 print(result)	3.5

%	a = 7 result = a % 2 print(result)	1
**	a = 2 result = a ** 3 print(result)	8
//	a = 7 result = a // 2 print(result)	3

Assignment Operator

```
number = 20
result = number
print(result)

str1 = 'hello world'
str2 = str1
print(str2)

bool1 = True
bool2 = bool1
print(bool2)
```



```
20
hello world
True
```

Assignment Operator

=	a = 2 print(a)	2
+=	a = 2 a += 3 print(a)	5
-=	a = 10 a -= 4 print(a)	6
*=	a = 4 a *= 7 print(a)	28

/=	a = 72 a /= 8 print(a)	9
//=	a = 7 a //= 2 print(a)	3
%=	a = 7 a %= 2 print(a)	1
**=	a = 4 a **= 2 print(a)	16

Comparison Operator

==	5 == 3	False
!=	5 != 3	True
>	4 > 3	True
<	3 < 3	False
>=	6 >= 6	True
<=	7 <= 8	True

Logical Operator

and	모두 참일 경우에만 True 리턴	2 < 5 and 3 < 10	True
or	하나라도 참이면 True 리턴	2 < 4 or 7 > 5	True
not	결과가 False이면 True 리턴	not (3 < 4 and 5 < 2)	True

Identity Operator

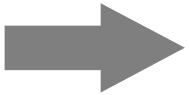
is	두 변수가 같은 객체메모리면 True 리턴	x = [“apple”, “banana”] y = [“apple”, “banana”] z = x print(x is z)	True
is not	두 변수가 같은 객체메모리가 아니면 True 리턴	x = [“apple”, “banana”] y = [“apple”, “banana”] print(x is not y)	True

Membership Operator

in	객체 내에 해당 값이 포함된다면 True 리턴	x = [“apple”, “banana”] print(“banana” in x)	True
not in	객체 내에 해당 값이 포함되지 않는다면 True 리턴	x = [“apple”, “banana”] print((“pineapple” not in x)	True

input & print

```
a = input()
print(a)

a = int(input())
print('a :', a)

a, b = input().split()
print("a + b =", a+b)

a, b = map(int, input().split())
print("a + b =", a+b)

abc = input("""input : """)
print(abc)
```



```
hi
hi
8
a : 8
1 2
a + b = 12
1 2
a + b = 3
input : string
string
```

Conditional Statement

a	b	a and b	a or b
True	True	True	True
True	False	False	True
False	True	False	True
False	False	False	False

Conditional Statement

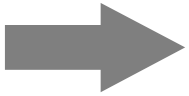
ASCII : 미국에서 표준화가 추진된 정보교환용 7비트 부호

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

A ~ Z	65 ~ 90
a ~ z	97 ~ 122

Conditional Statement

```
print('A' < 'a')
print('A' > 'z')
print('abc' < 'abcd')
print('a' in 'abc')
```



```
True
False
True
True
```

Conditional Statement

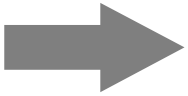
- **If문**

if <condition> :
    <block>

elif <condition> :
    <block>

else :
    <block>

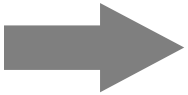
```
a = 1
if (a == 0) :
    print('a =', 0)
elif(a == 1) :
    print('a =', 1)
else:
    print('Nothing')
```



a = 1

Conditional Statement

```
1   flag = True
2
3   if flag:
4       print("참")
5   else:
6       print("거짓")
```

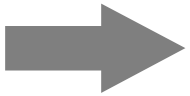


Loop

- **for문**

for <variable> in <list> :
    <block>

```
marks = [90, 25, 67, 45, 80]

number = 0
for mark in marks:
    number = number +1
    if mark >= 60:
        print("%d번 학생은 합격입니다." % number)
    else:
        print("%d번 학생은 불합격입니다." % number)
```

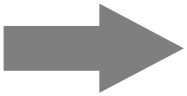


```
1번 학생은 합격입니다.
2번 학생은 불합격입니다.
3번 학생은 합격입니다.
4번 학생은 불합격입니다.
5번 학생은 합격입니다.
```

Loop

for <variable> in range (start, end) :
    <block>

```
1   for i in range (2,10):
2       for j in range(1,10):
3           print(i*j,end=" ")
4       print('')
```



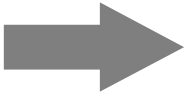
```
2 4 6 8 10 12 14 16 18
3 6 9 12 15 18 21 24 27
4 8 12 16 20 24 28 32 36
5 10 15 20 25 30 35 40 45
6 12 18 24 30 36 42 48 54
7 14 21 28 35 42 49 56 63
8 16 24 32 40 48 56 64 72
9 18 27 36 45 54 63 72 81
```

Loop

- **while문**

while <expression> :
    <block>

```
cnt = 0

while(cnt < 10):
    print(cnt)
    cnt = cnt + 1
```



```
0
1
2
3
4
5
6
7
8
9
```

Loop

- **break**

```
s = "C2H6O"
index = 1
for i in range(len(s)):
    if s[i].isdigit():
        index = i
        print(index)
```

1
3

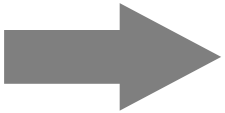
```
s = "C2H6O"
index = 1
for i in range(len(s)):
    if s[i].isdigit():
        index = i
        print(index)
        break
```

1

Loop

- **continue**

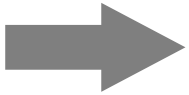
```
for i in range(11):
    if i % 2 == 0:
        continue
    print(i)
```



```
1
3
5
7
9
```

List

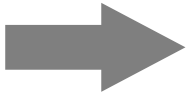
```
list = [ 'abcd', 786 , 2.23, 'john', 70.2 ]
tinylist = [123, 'john']

print(list)
print(list[0] )
print(list[1:3])
print(list[2:])
print(tinylist * 2)
print(list + tinylist)
```



```
['abcd', 786, 2.23, 'john', 70.2]
abcd
[786, 2.23]
[2.23, 'john', 70.2]
[123, 'john', 123, 'john']
['abcd', 786, 2.23, 'john', 70.2, 123, 'john']
```

Tuple

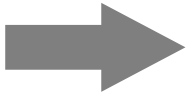
```
tuple = ( 'abcd', 786 , 2.23, 'john', 70.2  )
tinytuple = (123, 'john')

print(tuple)
print(tuple[0])
print(tuple[1:3])
print(tuple[2:])
print(tinytuple * 2)
print(tuple + tinytuple)
```



```
('abcd', 786, 2.23, 'john', 70.2)
abcd
(786, 2.23)
(2.23, 'john', 70.2)
(123, 'john', 123, 'john')
('abcd', 786, 2.23, 'john', 70.2, 123, 'john')
```

Dictionary

```
list = [ 'abcd', 786 , 2.23, 'john', 70.2 ]
tinylist = [123, 'john']

print(list)
print(list[0] )
print(list[1:3])
print(list[2:])
print(tinylist * 2)
print(list + tinylist)
```



```
['abcd', 786, 2.23, 'john', 70.2]
abcd
[786, 2.23]
[2.23, 'john', 70.2]
[123, 'john', 123, 'john']
['abcd', 786, 2.23, 'john', 70.2, 123, 'john']
```