Python

Jin Hyun Kim

2019

This Class

- “A Byte of Python”- Swaroop C H

In this class

- Why “python” in this course?
- Install compiler and IDE (PyCham)
- Lab 00 - “Hello World”

Why

- Simple
- Easy to Learn
- Free and Open Source
- High-level Language
- Portable (이식성)
- Object Oriented Language
- Extensible
- Embeddable
- Extensive Libraries

Lab 1

- print (Hello World)
  - Install python
  - Install “PyCham”
  - Create and configure the project
  - Run “python hello.py”

Basic

Basic

- Comments

```
print 'hello world' # Note that print is a statement
```

or:

```
# Note that print is a statement
print 'hello world'
```

Basic

- Constants
  - Numbers - **5**, **1.23**, **52.3E-4**
  - String - “This is a string”, ‘It’s a string’
  - Multiple string - ‘ ‘ ‘ … ’ ’ ’, “ “ “ … ” ” ”

Format

```
age = 20
name = 'Swaroop'

print '{0} was {1} years old when he wrote this book'.format(name, age)
print 'Why is {0} playing with that python?'.format(name)
```

- A number in “{}” is optional

```
name + ' is ' + str(age) + ' years old'
```

Escape Sequences

- What if a string includes “‘”? `'What's your name?'`

'What\'s your name?'

- ‘\’ -> ‘\\’
- “\n” (newline)

```
'This is the first line\nThis is the second line'
```

- Ignore escape sequence

```
r"Newlines are indicated by \n"
```

Variable Identifiers

- The first character of the identifier must be a letter of the alphabet (uppercase ASCII or lowercase ASCII or Unicode character) or an underscore (_).

- The rest of the identifier name can consist of letters (uppercase ASCII or lowercase ASCII or Unicode character), underscores (_) or digits (0-9).

- Identifier names are case-sensitive. For example, ***myname*** and ***myName*** are not the same. Note the lowercase n in the former and the uppercase in the latter.

Basic

- Variables
- Objects

Lines

Logical vs Physical Lines

```
i = 5
print i
```

is effectively same as

```
i = 5;
print i;
```

which is also same as

```
i = 5; print i;
```

and same as

```
i = 5; print i
```

Lines

- A line can be broken by "\" in code

```
s = 'This is a string. \
This continues the string.'
print s
```

Output:

```
This is a string. This continues the string.
```

Indentation

- Whitespace at the beginning of the line, called **indentation**, is important

```
i = 5
# Error below! Notice a single space at the start of the line
 print 'Value is ', i
print 'I repeat, the value is ', i
```

- Statements should go together and have the same indentation

```
if True:
    print 'Yes, it is true'
```

Operators

- +, -, x, /
- ** (power): ex) 3 ** 4 = 3 * 3 * 3 * 3
- % (modulo): ex)3 % 2 = 1
- << (bit, shift left): ex) 2 << 2 = 8
- >> (bit, shift right): ex) 11 >> 1 = 5
- & (bit, And): ex) 5 & 3 = 1 (0101 & 0011 = 0001)
- | (bit, Or): ex) 5 | 3 = 7 (0101 & 0011 = 0111)

Operators

- ^ (bit, Xor): ex) 5 ^ 3 = 6 (101 ^ 011 = 110)
- ~ (bit, flip, Return -(x+1)) ex) ~5 = -6
- > (Logical “greater than”)
- < (Logical “less than”)
- <= (Logical “greater than or equal to”)
- >= (Logical “less than or equal to”)
- == (Logical “equal to”)

Operators

- != (Logical “not equal to”)
- not (Logical negation): ex) x = True ; not x
- and (Boolean logical “AND”)
- or (Boolean logical “OR”)

Assignments

- Shortened expression

```
a = 2
a *= 3
```

- Precedence (연산 우선 순위)
  - 2 + 3 * 4

Control Flow

If

```
number = 23
guess = int(raw_input('Enter an integer : '))

if guess == number:
    # New block starts here
    print 'Congratulations, you guessed it.'
    print '(but you do not win any prizes!)'
    # New block ends here
elif guess < number:
    # Another block
    print 'No, it is a little higher than that'
    # You can do whatever you want in a block ...
else:
    print 'No, it is a little lower than that'
    # you must have guessed > number to reach here

print 'Done'
# This last statement is always executed,
# after the if statement is executed.
```

While

```
number = 23
running = True

while running:
    guess = int(raw_input('Enter an integer : '))

    if guess == number:
        print 'Congratulations, you guessed it.'
        # this causes the while loop to stop
        running = False
    elif guess < number:
        print 'No, it is a little higher than that.'
    else:
        print 'No, it is a little lower than that.'
else:
    print 'The while loop is over.'
    # Do anything else you want to do here

print 'Done'
```

for

```
for i in range(1, 5):
    print i
else:
    print 'The for loop is over'
```

break

```
while True:
    s = raw_input('Enter something : ')
    if s == 'quit':
        break
    print 'Length of the string is', len(s)
print 'Done'
```

continue

```
while True:
    s = raw_input('Enter something : ')
    if s == 'quit':
        break
    if len(s) < 3:
        print 'Too small'
        continue
    print 'Input is of sufficient length'
    # Do other kinds of processing here...
```