Introduction to Python

Jin-Soo Kim
(jinsoo.kim@snu.ac.kr)

Systems Software &
Architecture Lab.

Seoul National University

Dec 16 – 20, 2019

교재

- 데이터 과학을 위한 파이썬 프로그래밍
- 최성철 저
- 한빛 아카데미, 2019.

일정

<table>
<tr><th></th><th></th><th>12/16 (Mon)</th><th>12/17 (Tue)</th><th>12/18 (Wed)</th><th>12/19 (Thu)</th><th>12/20 (Fri)</th></tr>
<tr><td rowspan="4">오전</td><td>8</td><td>Introduction
(Ch. 1)</td><td>What is
Programming?</td><td>Functions
(Ch. 5)</td><td rowspan="2">Dictionary
(Ch. 7.4)</td><td rowspan="2">[실습]</td></tr>
<tr><td>9</td><td>Basic data types
(Ch. 2, 3.1-3.3)</td><td rowspan="2">Lists
(Ch. 3.4-3.5, 7.1-7.2)</td><td>File I/O
(Ch. 12.2)</td></tr>
<tr><td>10</td><td rowspan="8">[실습]</td><td rowspan="2">Strings
(Ch. 6)</td><td rowspan="2">Tuple, Set
(Ch. 7.3)</td><td rowspan="5">Project</td></tr>
<tr><td>11</td><td>Control structures
(Ch. 4)</td></tr>
<tr><td rowspan="4">오후</td><td>1</td><td rowspan="4">[실습]</td><td rowspan="4">[실습]</td><td>List comprehension
(Ch. 8)</td></tr>
<tr><td>2</td><td rowspan="2">Lambda
(Ch. 9.1)</td></tr>
<tr><td>3</td><td rowspan="2">Project 발표
Wrap-up</td></tr>
<tr><td>4</td><td>Map/reduce
(Ch. 9.2)</td></tr>
</table>

About Me

- 김진수 (Jin-Soo Kim)
  - Professor @ Dept. of Computer Science & Engineering, SNU
  - Systems Software & Architecture Laboratory
  - Operating systems, Storage systems, Parallel and Distributed computing, Embedded systems, …
- E-mail: jinsoo.kim@snu.ac.kr
- http://csl.snu.ac.kr
- Tel: 02-880-7302
- Office: SNU Engineering Building #301-520
- https://github.com/snu-csl/pyrisc: Educational RISC-V toolset in Python

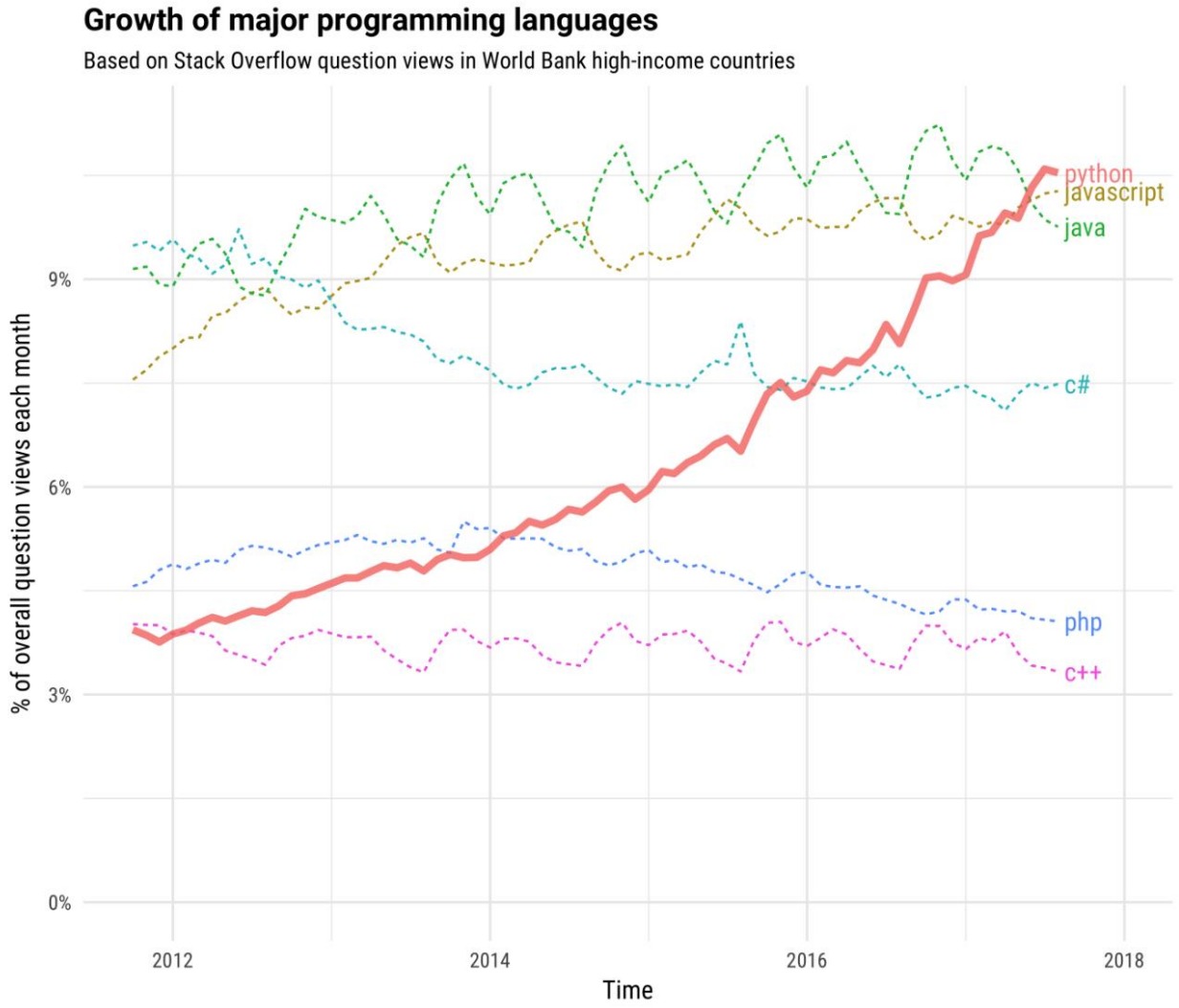
Python

Why Python?

- 1st place among the "Top Programming Languages" (IEEE Spectrum, 2019)
- "Fastest growing major programming language" (stackoverflow.com, 2019)
- The 2nd most popular programming language in GitHub (Nov. 2019)
- "The language of AI"

Rank	Language	Type	Score
1	Python		100.0
2	Java		96.3
3	C		94.4
4	C++		87.5
5	R		81.5
6	JavaScript		79.4
7	C#		74.5
8	Matlab		70.6
9	Swift		69.1
10	Go		68.0

Growth of Major Programming Languages



The Birth of Python

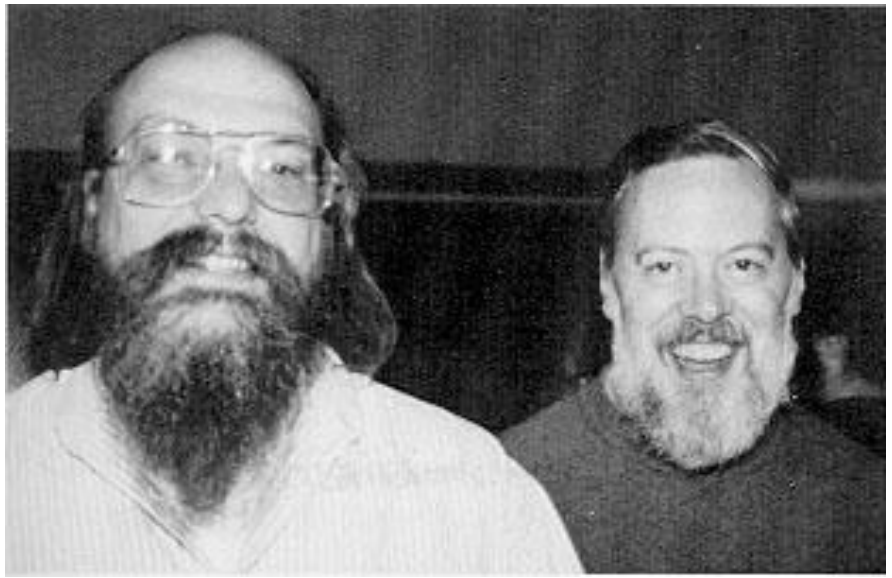
- Developed by Guido van Rossum in 1990

Over six years ago, in December 1989, I was looking for a "hobby" programming project that would keep me occupied during the week around Christmas. My office … would be closed, but I had a home computer, and not much else on my hands. I decided to write an interpreter for the new scripting language I had been thinking about lately: a descendant of ABC that would appeal to Unix/C hackers. I chose Python as a working title for the project, being in a slightly irreverent mood (and a big fan of Monty Python's Flying Circus).

Sounds Familiar?

- Unix developed by Ken Thompson in 1969

… It was the summer of '69. In fact, my wife went on vacation to my family's place in California.... I allocated a week each to the operating system, the shell, the editor, and the assembler, to reproduce itself, and during the month she was gone, it was totally rewritten in a form that looked like an operating system, with tools that were sort of known, you know, assembler, editor, and shell .... Yeh, essentially one person for a month.



Python Philosophy

- Beautiful is better than ugly
- Explicit is better than implicit
- Simple is better than complex
- Complex is better than complicated
- Readability counts

- "There is more than one way to do it" (Perl)
- "There should be one – and preferably only one – obvious way to do it" (Python)

Python Goals

- "Computer programming for Everybody"
  - DARPA funding proposal

- An easy and intuitive language just as powerful as major competitors
- Open source, so anyone can contribute to its development
- Code that is as understandable as plain English
- Suitability for everyday tasks, allowing for short development times

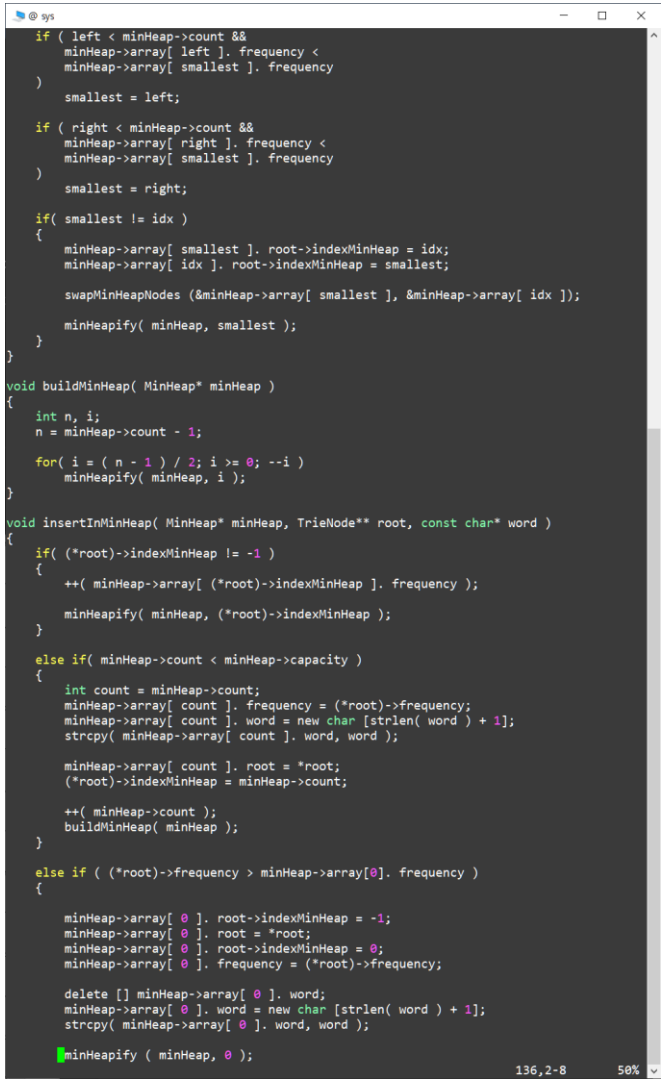
Program like Plain English?

```
name = input('Enter file:')
f = open(name)

counts = dict()
for line in f:
    words = line.split()
    for word in words:
        counts[word] = counts.get(word, 0) + 1

lst = list()
for key, val in counts.items():
    t = (val, key)
    list.append(t)

lst = sorted(lst, reverse=True)

for val, key in lst[:5]:
    print(key, val)
```

Compare with this:

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

# define MAX_CHARS 26
# define MAX_WORD_SIZE 30

struct TrieNode
{
    bool isEnd;
    unsigned frequency;
    int indexMinHeap;
    TrieNode* child[MAX_CHARS];
};

struct MinHeapNode
{
    TrieNode* root;
    unsigned frequency;
    char* word;
};

struct MinHeap
{
    unsigned capacity;
    int count;
    MinHeapNode* array;
};

TrieNode* newTrieNode()
{
    TrieNode* trieNode = new TrieNode;

    trieNode->isEnd = 0;
    trieNode->frequency = 0;
    trieNode->indexMinHeap = -1;
    for( int i = 0; i < MAX_CHARS; ++i )
        trieNode->child[i] = NULL;

    return trieNode;
}

MinHeap* createMinHeap( int capacity )
{
    MinHeap* minHeap = new MinHeap;

    minHeap->capacity = capacity;
    minHeap->count = 0;

    minHeap->array = new MinHeapNode [ minHeap->capacity ];

    return minHeap;
}

void swapMinHeapNodes ( MinHeapNode* a, MinHeapNode* b )
{
    MinHeapNode temp = *a;
    *a = *b;
    *b = temp;
}

void minHeapify( MinHeap* minHeap, int idx )
{
    int left, right, smallest;

    left = 2 * idx + 1;
    right = 2 * idx + 2;
    smallest = idx;
```

68,1-4 Top



```
    if ( left < minHeap->count &&
        minHeap->array[ left ]. frequency <
        minHeap->array[ smallest ]. frequency
    )
        smallest = left;

    if ( right < minHeap->count &&
        minHeap->array[ right ]. frequency <
        minHeap->array[ smallest ]. frequency
    )
        smallest = right;

    if( smallest != idx )
    {
        minHeap->array[ smallest ]. root->indexMinHeap = idx;
        minHeap->array[ idx ]. root->indexMinHeap = smallest;

        swapMinHeapNodes (&minHeap->array[ smallest ], &minHeap->array[ idx ]);

        minHeapify( minHeap, smallest );
    }
}

void buildMinHeap( MinHeap* minHeap )
{
    int n, i;
    n = minHeap->count - 1;

    for( i = ( n - 1 ) / 2; i >= 0; --i )
        minHeapify( minHeap, i );
}

void insertInMinHeap( MinHeap* minHeap, TrieNode** root, const char* word )
{
    if( (*root)->indexMinHeap != -1 )
    {
        ++( minHeap->array[ (*root)->indexMinHeap ]. frequency );

        minHeapify( minHeap, (*root)->indexMinHeap );
    }

    else if( minHeap->count < minHeap->capacity )
    {
        int count = minHeap->count;
        minHeap->array[ count ]. frequency = (*root)->frequency;
        minHeap->array[ count ]. word = new char [strlen( word ) + 1];
        strcpy( minHeap->array[ count ]. word, word );

        minHeap->array[ count ]. root = *root;
        (*root)->indexMinHeap = minHeap->count;

        ++( minHeap->count );
        buildMinHeap( minHeap );
    }

    else if ( (*root)->frequency > minHeap->array[0]. frequency )
    {

        minHeap->array[ 0 ]. root->indexMinHeap = -1;
        minHeap->array[ 0 ]. root = *root;
        minHeap->array[ 0 ]. root->indexMinHeap = 0;
        minHeap->array[ 0 ]. frequency = (*root)->frequency;

        delete [] minHeap->array[ 0 ]. word;
        minHeap->array[ 0 ]. word = new char [strlen( word ) + 1];
        strcpy( minHeap->array[ 0 ]. word, word );

        minHeapify ( minHeap, 0 );
```

136,2-8 50%

```
    }
}

void insertUtil ( TrieNode** root, MinHeap* minHeap,
                  const char* word, const char* dupWord )
{
    if ( *root == NULL )
        *root = newTrieNode();

    if ( *word != '\0' )
        insertUtil ( &((*root)->child[ tolower( *word ) - 97 ]),
                     minHeap, word + 1, dupWord );
    else
    {
        if ( (*root)->isEnd )
            ++( (*root)->frequency );
        else
        {
            (*root)->isEnd = 1;
            (*root)->frequency = 1;
        }

        insertInMinHeap( minHeap, root, dupWord );
    }
}


void insertTrieAndHeap(const char *word, TrieNode** root, MinHeap* minHeap)
{
    insertUtil( root, minHeap, word, word );
}

void displayMinHeap( MinHeap* minHeap )
{
    int i;

    for( i = 0; i < minHeap->count; ++i )
    {
        printf( "%s : %d\n", minHeap->array[i].word,
                             minHeap->array[i].frequency );
    }
}

void printKMostFreq( FILE* fp, int k )
{
    MinHeap* minHeap = createMinHeap( k );

    TrieNode* root = NULL;

    char buffer[MAX_WORD_SIZE];

    while( fscanf( fp, "%s", buffer ) != EOF )
        insertTrieAndHeap(buffer, &root, minHeap);

    displayMinHeap( minHeap );
}

int main()
{
    int k = 5;
    FILE *fp = fopen ("test.txt", "r");
    if (fp == NULL)
        printf ("File doesn't exist ");
    else
        printKMostFreq (fp, k);
    return 0;
}
```

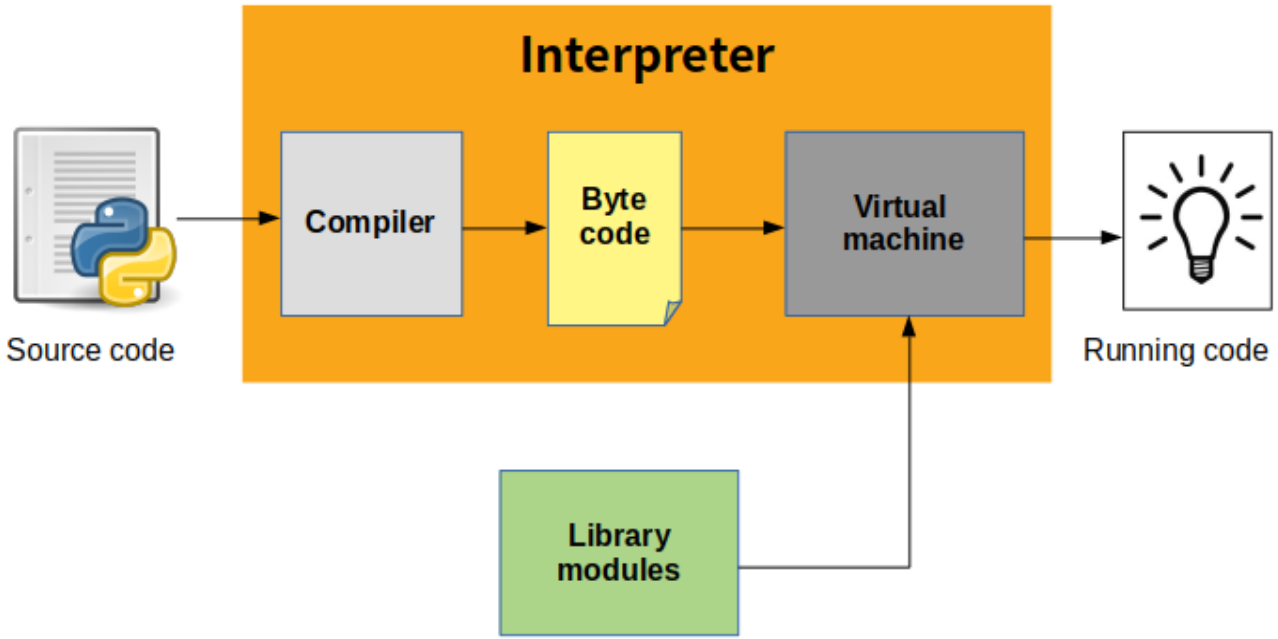
204,0-1 Bot

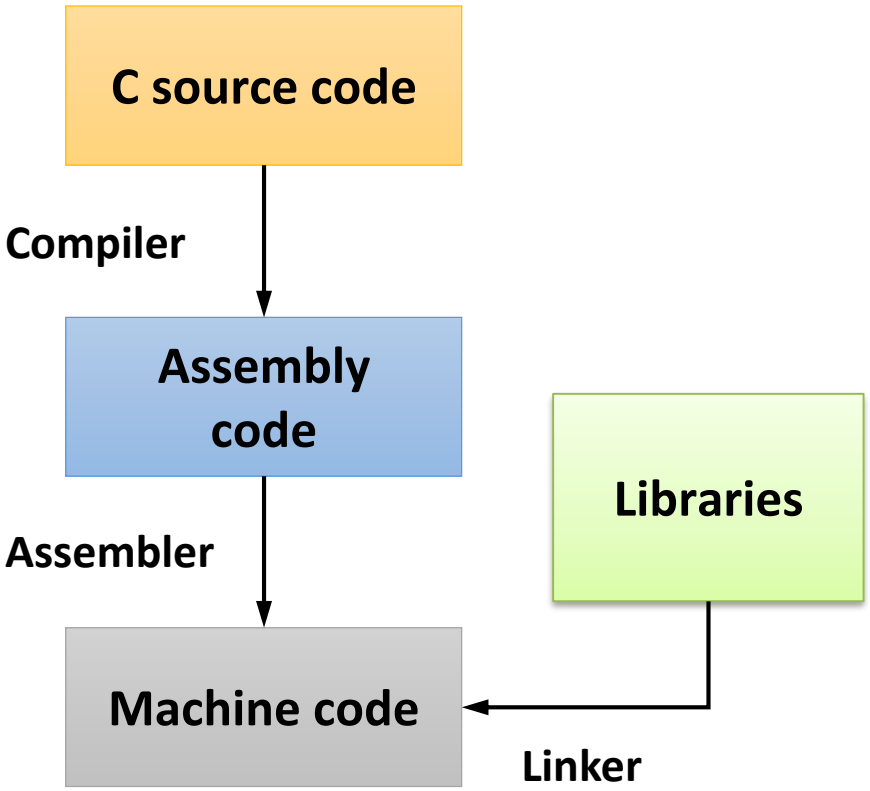
Source: https://www.geeksforgeeks.org/find-the-k-most-frequent-words-from-a-file/

Python Features

- Multi-paradigm programming language
  - Structured
  - Object-oriented
  - Functional
  - …
- Highly extensible
  - Modules can be written in other languages such as C, C++, …
- Interpreted
- "Pythonic"

Interpreted vs. Compiled





Python Versions

- Python 1.0 (1990)
- Python 2.0 (2000)
- Python 3.0 (2008) – Not backward compatible to 2.0

- The latest version: 3.8.0

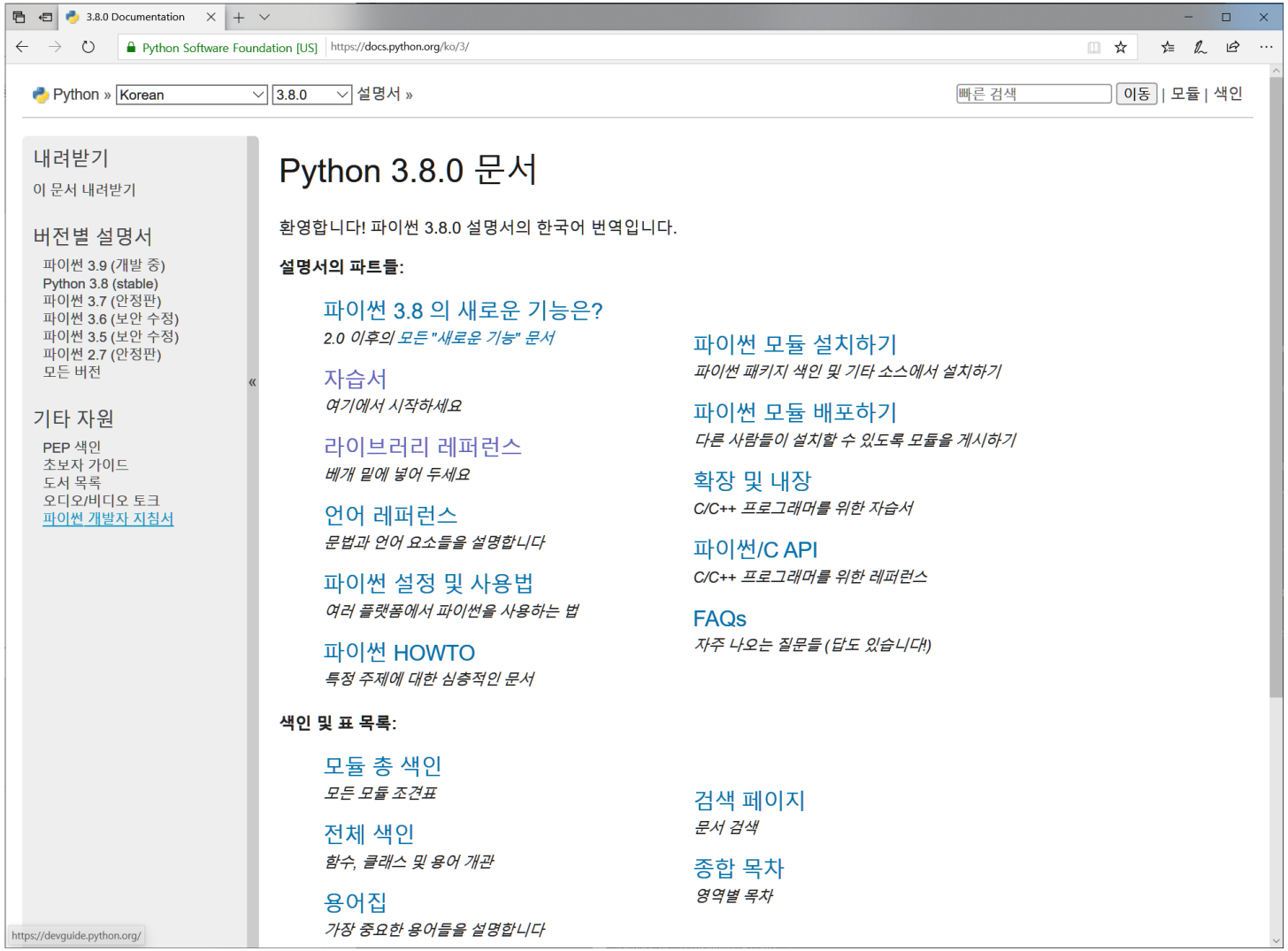
- Official homepage: https://www.python.org
- Tutorial: https://docs.python.org/ko/3/tutorial

Python Applications

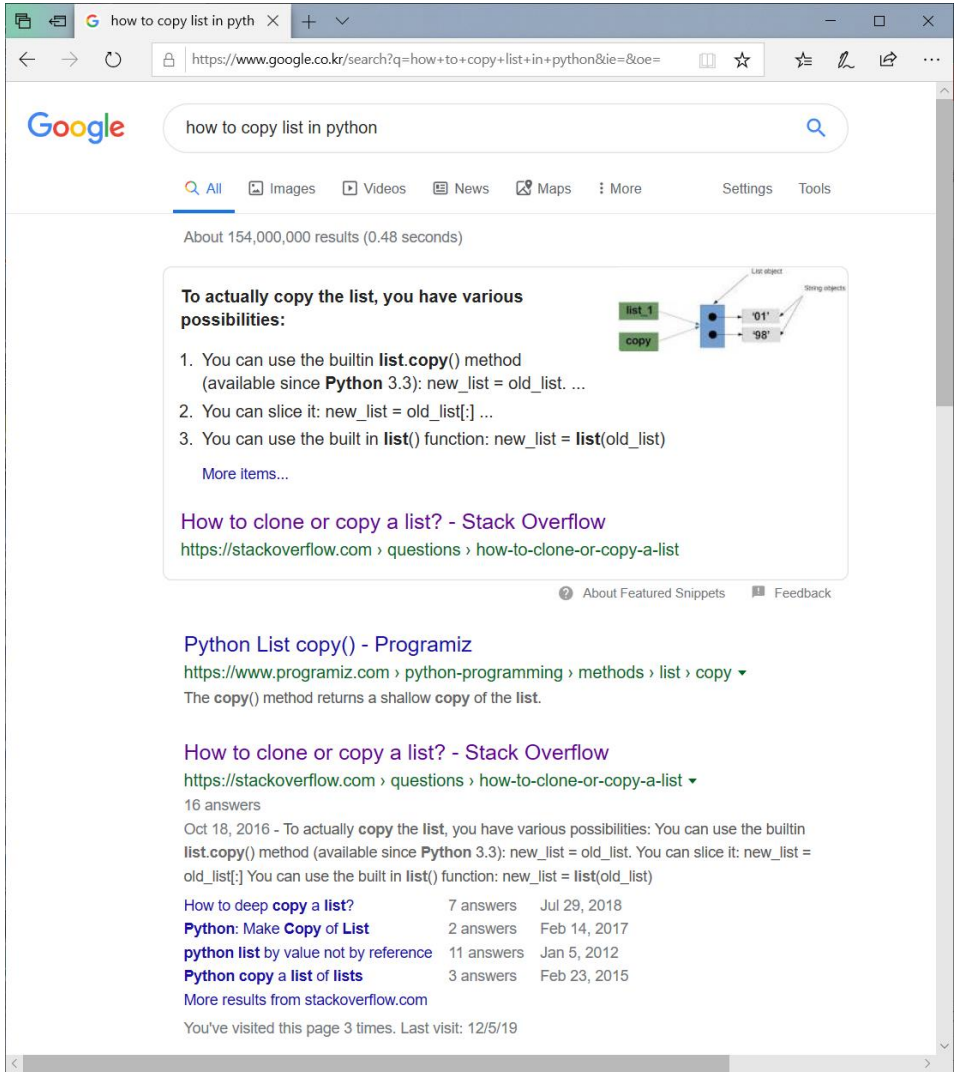

- Machine Learning (TensorFlow, PyTorch, etc.)
- GUI Applications (Kivy, Tkinter, PyQt, etc.)
- Web frameworks (Django used by YouTube, Instagram, Dropbox)
- Image processing (OpenCV, Pillow, etc.)
- Web scraping (Scrapy, BeautifulSoup, etc.)
- Text processing (NLTK, KoNLPy, Word2vec, etc.)
- Test frameworks
- Multimedia
- Scientific computing and many more …

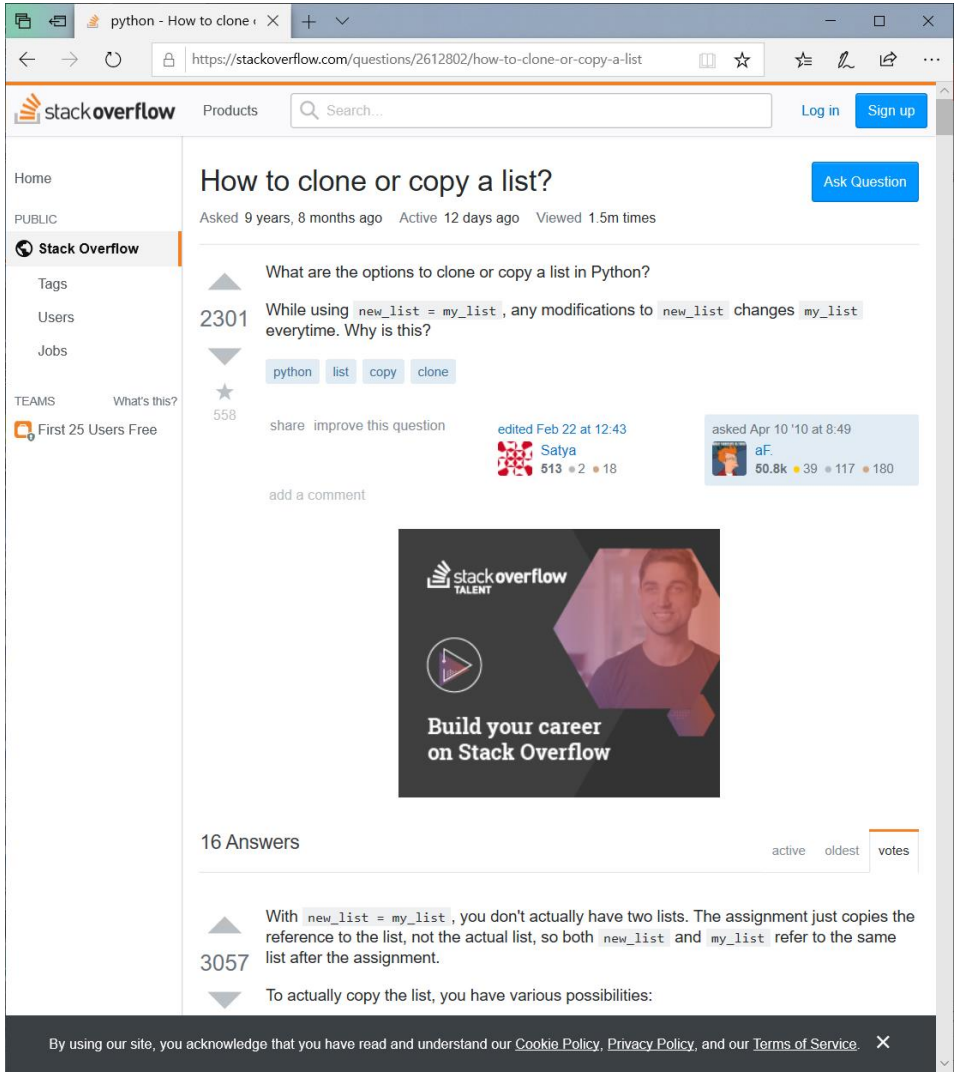


https://docs.python.org

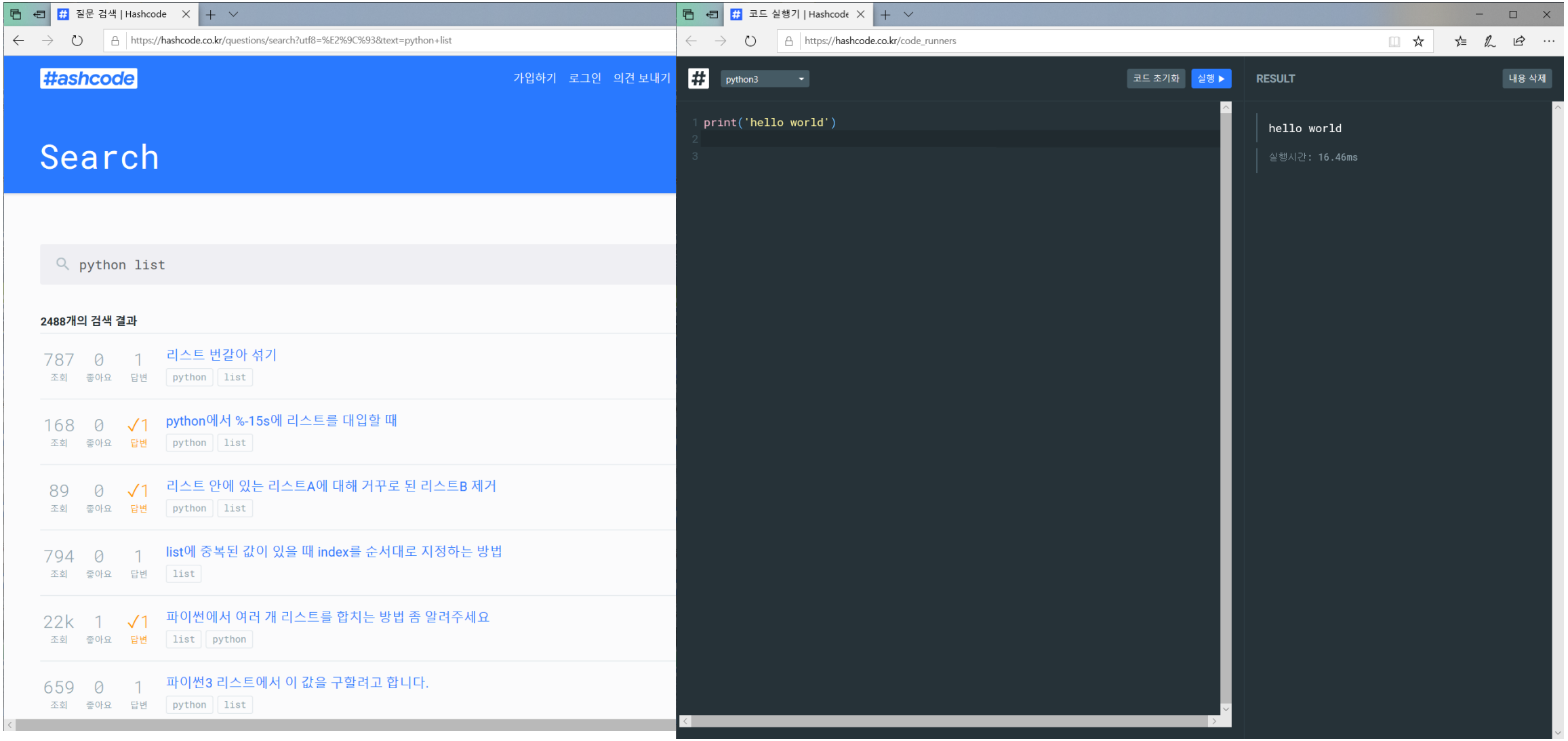


https://stackoverflow.com





https://hashcode.co.kr



Welcome to the World of Spam!

Source: https://www.youtube.com/watch?v=Gxtsa-OvQLA