Intro to Computer Science

Local Laboratory

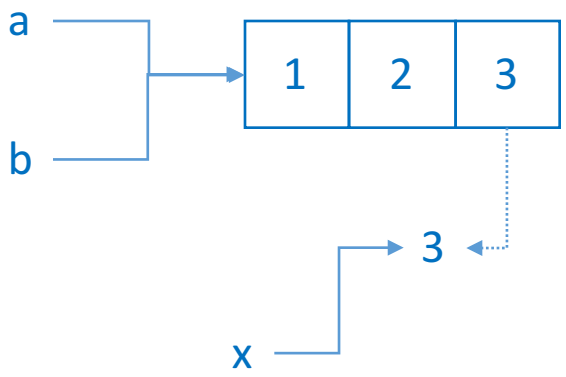
*** Udacity – Intro to Computer Science**

Pop

<list>.pop() -> element

list의 마지막 요소를 지우고(mutate하게) 그 값을 반환(return)함

```
a = [1, 2, 3]
b = a
x = a.pop()
```

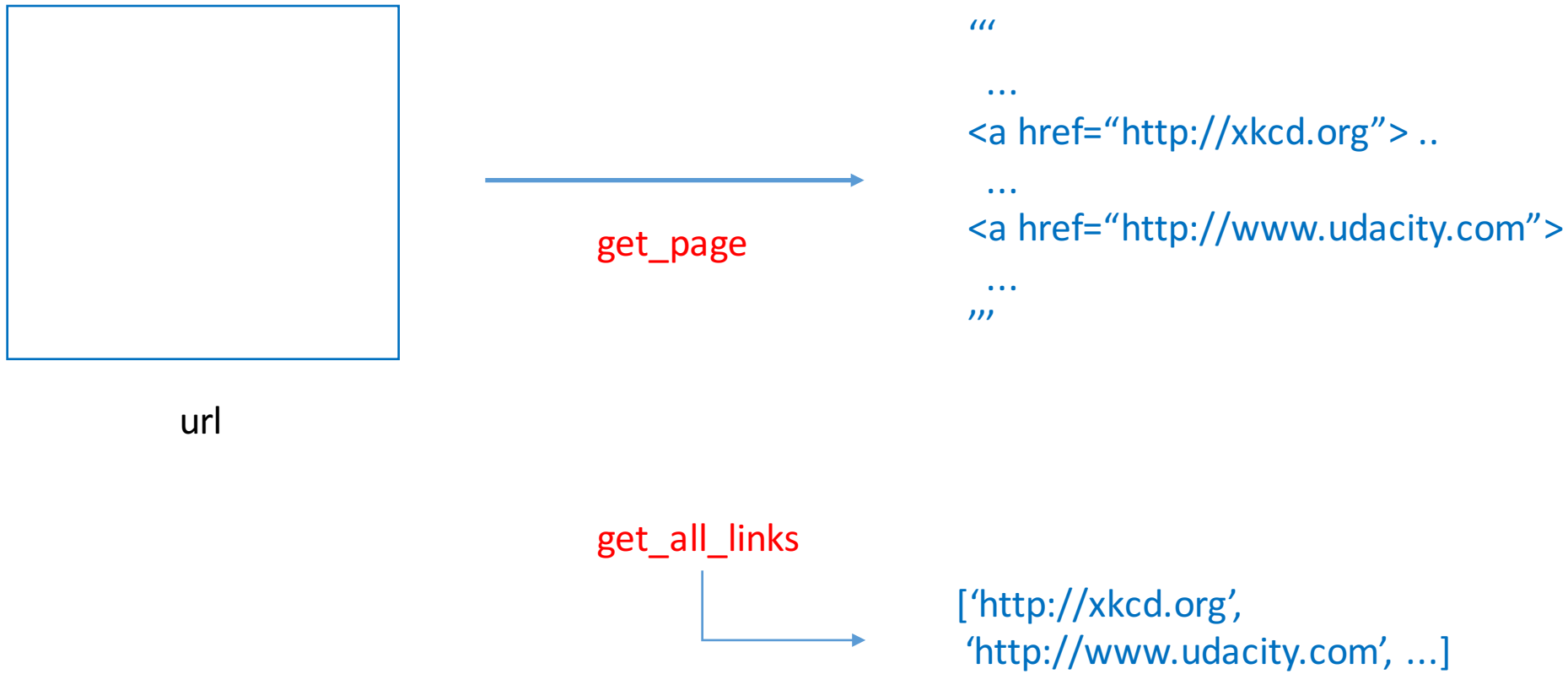


Quiz: Pop Quiz

p 는 적어도 두개의 요소를 가지는 리스트라고 가정한다.
아래의 코드들 중 코드의 마지막에서 p의 값이 변하지 않는 것들을 모두 고르시오.

- ❑ p.append(3)
  p.pop()
- ❑ x = p.pop()
  p.append(x)
- ❑ x = p.pop()
  y = p.pop()
  p.append(x)
  p.append(y)
- ❑ x = p.pop()
  ❑ y = p.pop()
  ❑ p.append(y)
  ❑ p.append(x)

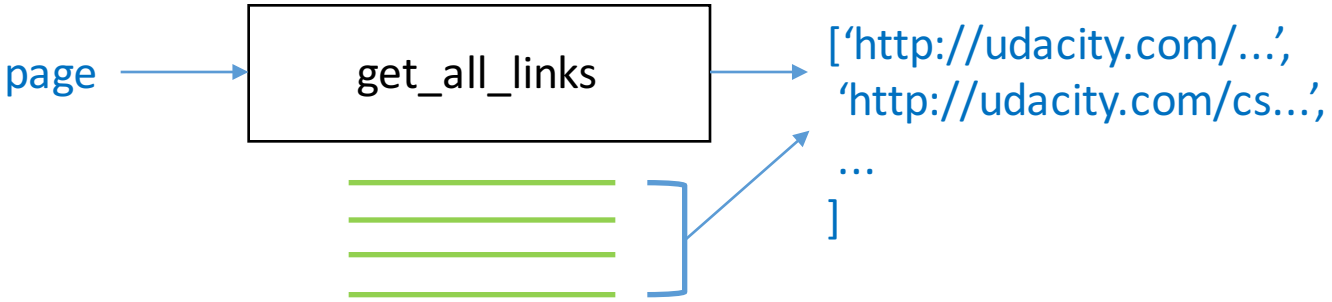
Collecting Links



Get All Links

```
def get_next_target(page):
    start_link = page.find('<a  href=')
    if start_link == -1:
        return None, 0
    start_quote = page.find('"', start_link)
    end_quote  = page.find('"', start_quote + 1)
    url = page[start_quote + 1:end_quote]
    return url, end_quote

def print_all_links(page):
    while True:
        url,endpos  = get_next_target(page)
        if url:
            print(url)
            page = page[endpos:]
        else:
            break
```



Links

code

```
link = get_all_links(get_page('http://www.udacity.com/cs101x/index.html'))
print(link)
print(link[0])
```

../cs101/index.html'

This is a test page for learning to crawl!

It is a good idea to learn to crawl before you try to walk or fly.

result

```
['http://www.udacity.com/cs101x/crawling.html',
'http://www.udacity.com/cs101x/walking.html',
'http://www.udacity.com/cs101x/flying.html']

http://www.udacity.com/cs101x/crawling.html
```

Quiz: Starting Get All Links

```
def get_all_links(page):
    links = [          ]
    while True:
        url,endpos = get_next_target(page)
        if url:
            print(url)
            page = page[endpos:]
        else:
            break
```

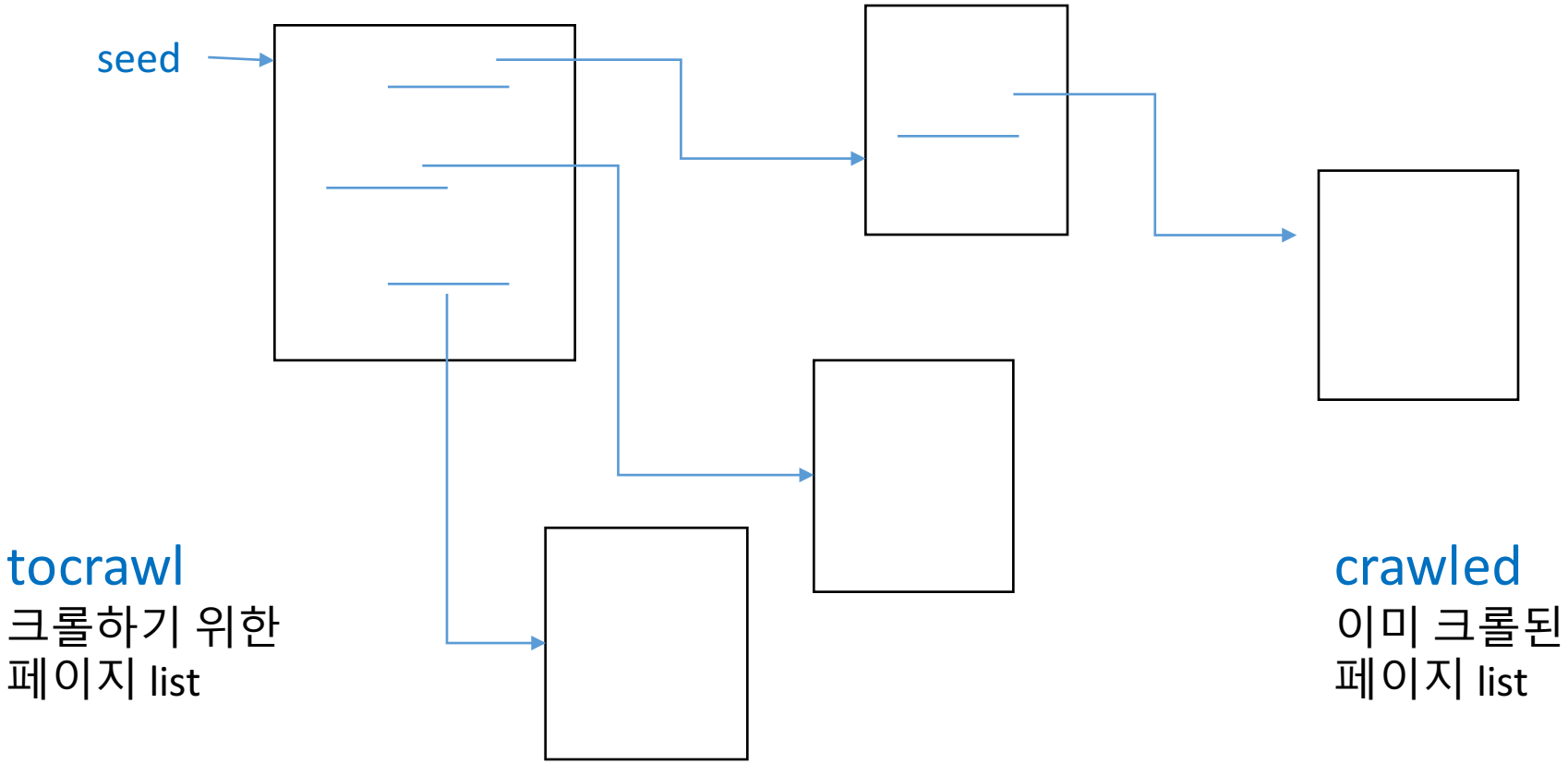
Quiz: Updating Links

```
def get_all_links(page):
    links = []
    while True:
        url,endpos = get_next_target(page)
        if url:
            [                    ]
            page = page[endpos:]
        else:
            break
```

Quiz: Finishing Get All Links

```
def get_all_links(page):
    links = []
    while True:
        url,endpos = get_next_target(page)
        if url:
            links.append(url)
            page = page[endpos:]
        else:
            break
    return links
```

Finishing the Web Crawler



Finishing the Web Crawler

seed = ‘http://www.udacity.com/cs101x/index.html’

tocrawl	crawled
[~~‘.../index.html’~~]	[]
	[‘.../index.html’]
[~~‘.../crawling.html’~~, ‘.../walking.html’, ~~‘.../flying.html’~~]	
	[‘.../index.html’, ‘.../flying.html’]
+ ‘.../kicking.html’	+ ‘.../crawling.html’

../cs101/index.html’

This is a test page for learning to crawl!

It is a good idea to learn to crawl before you try to walk or fly.

../cs101/crawling.html’

I have not learned to crawl yet, but I am quite good at kicking.

../cs101/flying.html’

The magic words are Squeamish Ossifrage!

../cs101/kicking.html’

Kick! Kick! Kick!

Quiz: Crawling Process

pseudo code

```
start with tocrawl = [seed]
crawled = []
while there are more pages tocrawl:
    pick a page from tocrawl
    add that page to crawled
    add all the link targets on this page to tocrawl
return crawled
```

Quiz: Crawling Process

아래의 pseudo code 프로세스를 다음의 seed 페이지에서 시작한다면 어떤 일이 일어 나겠는가?

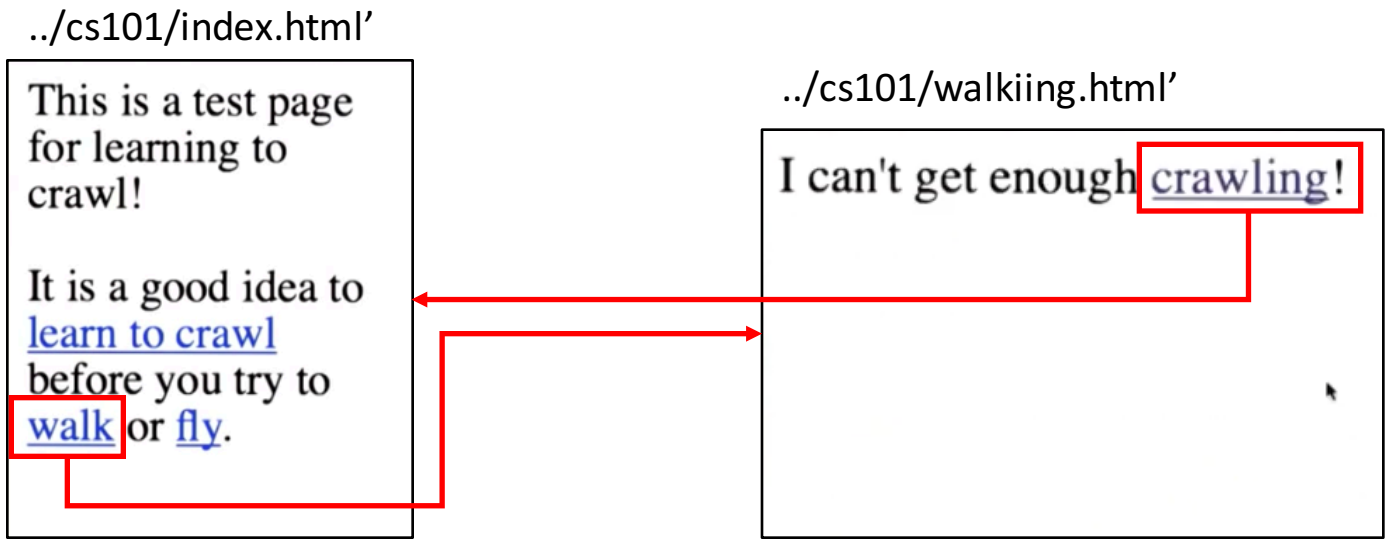
http://www.udacity.com/cs101x/index.html

pseudo code

```
start with tocrawl = [seed]
crawled = []
while there are more pages tocrawl:
    pick a page from tocrawl
    add that page to crawled
    add all the link targets on this page to tocrawl
return crawled
```

- ❑ It will return a list of <u>all</u> the urls reachable from the seed page.
- ❑ It will return a list of <u>some</u> of the urls reachable from the seed page.
- ❑ It will never return.

Crawling Process



Crawling Process

```
start with tocrawl = [seed]
crawled = []
while there are more pages tocrawl:
    pick a page from tocrawl
    add that page to crawled
    add all the link targets on this page to tocrawl
return crawled
```

이미 크롤링한 페이지인지 체크

Quiz: Crawl Web

seed 페이지 주소를 입력으로하여, seed 페이지로부터 시작하여 닿을 수 있는 모든 페이지의 url을 요소로 하는 리스트를 리턴하는 crawl_web이라는 프로시저를 정의하시오.

```
def crawl_web(seed):
    tocrawl = [seed]
    crawled = []
```

Quiz: Crawl Web Loop

```
def crawl_web(seed):
    tocrawl = [seed]
    crawled =  []
    while tocrawl:
        page = tocrawl.pop()
```

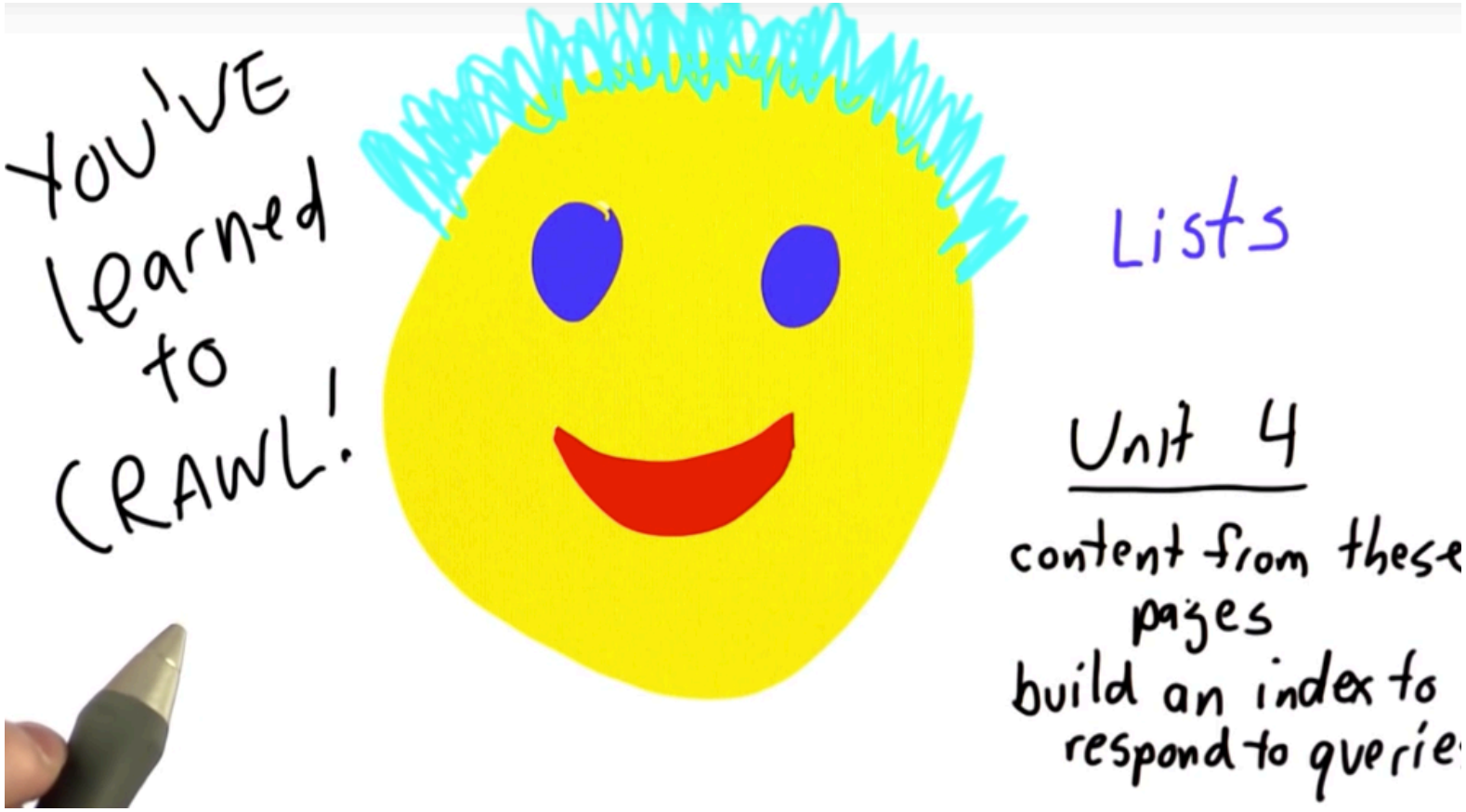
Quiz: Crawl If

```
def crawl_web(seed):
    tocrawl = [seed]
    crawled =  []
    while tocrawl:
        page = tocrawl.pop()
        if [page not in crawled] :
            crawl this page
```

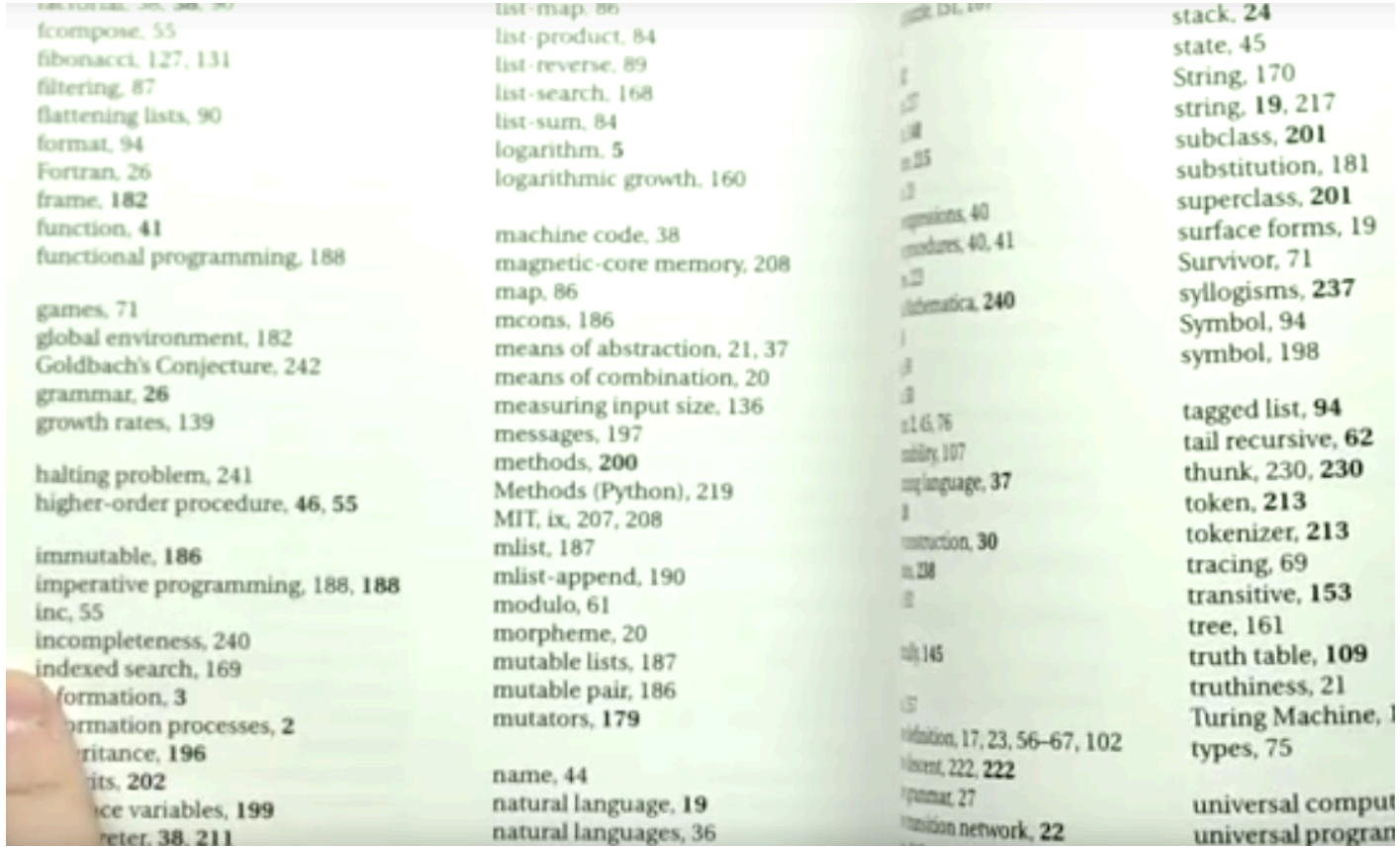
Quiz: Finishing Crawl Web

```
def crawl_web(seed):
    tocrawl = [seed]
    crawled =  []
    while tocrawl:
        page = tocrawl.pop()
        if page not in crawled:
            union(tocrawl, get_all_links(get_page(page)))
            crawled.append(page)
    return crawled
```

Conclusion


YOU'VE learned to CRAWL!
Lists
Unit 4
content from these pages
build an index to respond to queries

Unit 4: Responding to Queries



Quiz: Data Structures

아래 중 어떤 자료구조가 우리의 검색엔진을 위한 인덱스를 표현하기에 가장 좋은 방법일지 선택하시오.

- ❑ [<keyword_1>, <url_1_1>, <url_1_2>, <keyword_2>, ...]
- ❑ [[<keyword_1>, <url_1_1>, <url_1_2>],
  [<keyword_2>, <url_2_1>], ...
  ]
- ❑ [[<url1_1>, [<keyword_1>, <keyword_23>, ...]],
  [<url2_1>, [<keyword_2>, ...]],
  ...]
- ❑ [[<keyword_1>, [<url_1_1>, <url_1_2>]],
  [<keyword_2>, [<url_2_1>]], ...]

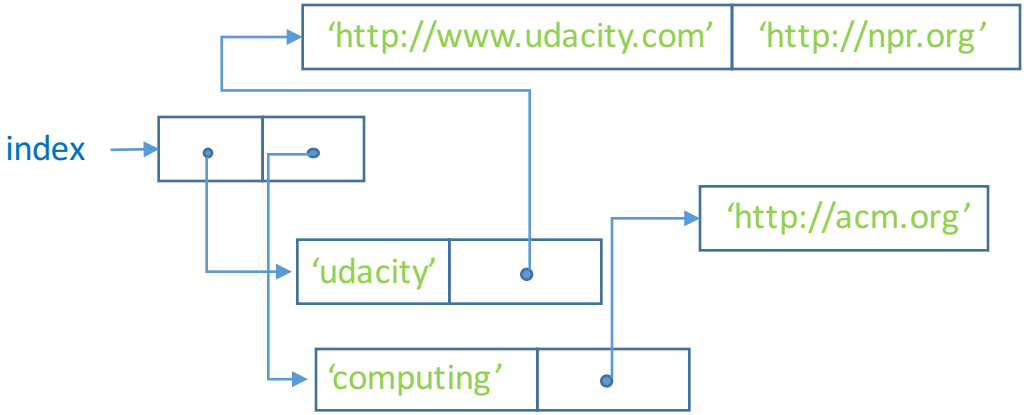
Quiz: Add to Index

아래 3개의 입력을 받는
add_to_index라는 프로시져를 정의하시오.

- an index [[<keyword>, [<url>, ...]], ...]
- a keyword string
- a url string

만약 keyword가 이미 index에 있다면, keyword와 연관된 url list에 url을 추가하시오.
만약 keyword가 index에 없다면 index에
다음과 같이 entry를 추가하시오: [keyword, [url]]

```
index = []
add_to_index(index, 'udacity', 'http://udacity.com')
add_to_index(index, 'computing', 'http://acm.org')
add_to_index('udacity', 'http://npr.org')
```



Add to Index

```
def add_to_index(index, keyword, url):
    for entry in index:
        if entry[0] == keyword:
            entry[1].append(url)
            return
    index.append([keyword, [url]])
```

Quiz: Lookup

다음 2개의 입력을 가지는
lookup 이라는 프로시져를 정의하시오.

- an index
- keyword

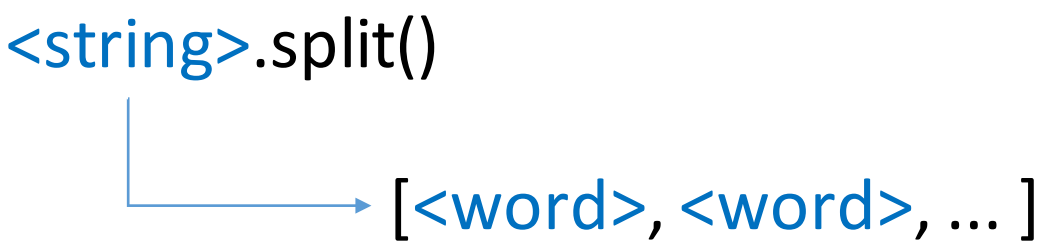
출력(리턴값)은 keyword와 연관된 url들의 list여야 한다.
만약 keyword가 index에 존재하지 않는다면, 출력값은 빈 list여야 한다.

lookup(‘udacity’) -> [‘http://udacity.com’,
‘http://npr.org]

Lookup

```
def lookup (index, keyword):
    for entry in index:
        if entry[0] == keyword:
            return entry[1]
    return []
```

Building the Web Index



code

```
quote = "in Washington, it's dog eat dog. In academia, it's exactly the opposite.  --- Robert Reich"
print(quote.split())
```

result

```
['In', 'Washington,', "it's", 'dog', 'eat', 'dog.', 'In', 'academia,',
"it's", 'exactly', 'the', 'opposite.', '---', 'Robert', 'Reich']
```

Quiz: Add Page to Index

다음 3개의 입력을 받는
add_page_to_index 라는 프로시저를 정의하시오.

- index
- url (string)
- content (string)

이 프로시저는 단어와 관련 있는 url list에 url을 추가하여,
page에서 찾은 모든 단어에 대한 index를 update한다.

code

```
index = []
add_page_to_index(index, 'www.fake.com', 'This is a test')
print(index)
print(index[0])
```

result

```
[['This', ['www.fake.com']], ['is', ['www.fake.com']], ['a', ['www.fake.com']], ['test', ['www.fake.com']]]
['This',['www.fake.com']]
```

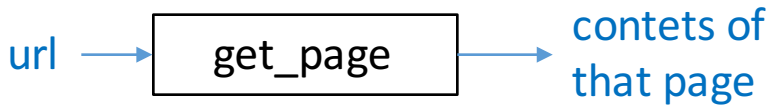
Quiz: Add Page to Index

```
def add_page_to_index(index, url, content):
    words = content.split()
    for word in words:
        add_to_index(index, word, url)
```

Quiz: Finishing the Web Cralwer

```
def crawl_web(seed):
    tocrawl = [seed]
    crawled = []
    index = []
    while tocrawl:
        page = tocrawl.pop()
        if page not in crawled:
            content = get_page(page)
            add_page_to_index(index, page, content)
            union(tocrawl, get_all_links(content))
            crawled.append(page)
    return index
```

get_page




```
def get_page(url):
    try:
        import urllib.request
        return urllib.request.urlopen(url).read().decode("utf8")
    except:
        return ""

content = get_page('http://www.xkcd.com/353')
```