

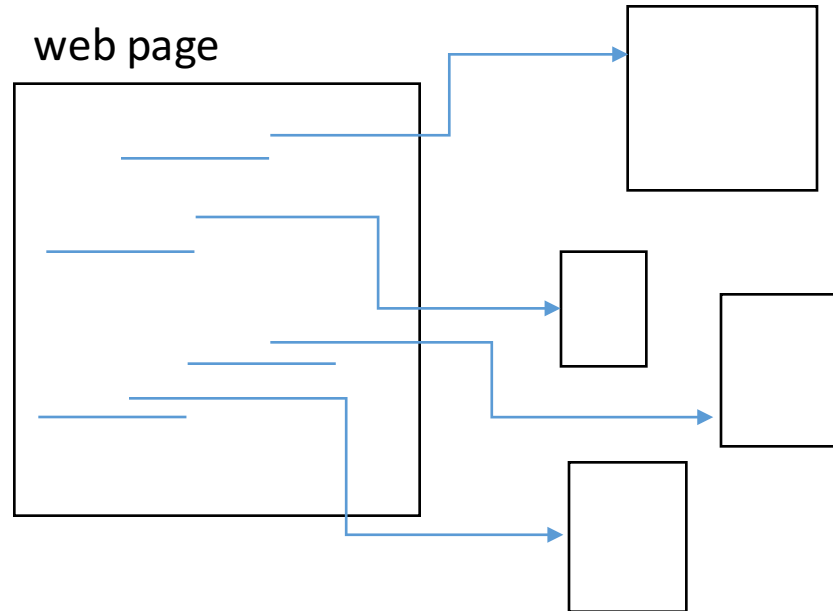
Intro to Computer Science

Local Laboratory

*** Udacity – Intro to Computer Science**

Introduction

Unit 2: How to Repeat **Procedures & Control**



```
page = ... contents of some web page ...
start_link = page.find('<a href=')
start_quote = page.find('"' , start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
start_link = page.find('<a href=')
start_quote = page.find('"' , start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
start_link = page.find('<a href=')
start_quote = page.find('"' , start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
```

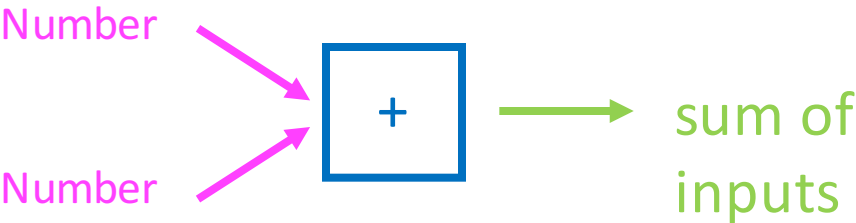
Motivating Procedures

```
page = ... contents of some web page ...
start_link = page.find('<a href=')
start_quote = page.find('"', start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]

start_link = page.find('<a href=')
start_quote = page.find('"', start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]

start_link = page.find('<a href=')
start_quote = page.find('"', start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

Introducing Procedures



```
def <name> ( <parameters> ):  
    <block>
```

Diagrammatic annotations for the code block:
- A magenta arrow labeled "inputs" points to the parameter list `<parameters>`.
- A magenta arrow points to the opening parenthesis of the parameter list.
- A magenta vertical line labeled "tab" is positioned to the left of the block content, indicating indentation.

Procedure Code

procedure



```
page = ... contents of some web page ...
start_link = page.find('<a href=')
start_quote = page.find('"', start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

```
start_link = page.find('<a href=')
start_quote = page.find('"', start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

```
start_link = page.find('<a href=')
start_quote = page.find('"', start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

Quiz: Procedure Code

아래 보기 중 `get_next_target` 프로시저를 위한 input 값은 어떤 것들이 되어야하는가?

- ☐ 링크의 시작 위치에 대한 number 값
- ☐ 마지막 따옴표의 시작 위치에 대한 number 값
- ☐ 웹페이지 코드의 남은 내용에 대한 string 값
- ☐ 링크의 시작 위치에 대한 number 값,
페이지 내용에 대한 string 값

procedure



```
page = ... contents of some web page ...
start_link = page.find('<a href=')
start_quote = page.find('"', start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

```
start_link = page.find('<a href=')
start_quote = page.find('"', start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

```
start_link = page.find('<a href=')
start_quote = page.find('"', start_link)
end_quote = page.find('"', start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

Quiz: Output

아래 보기 중 `get_next_target` 프로시저를 위한 output 값들은 어떤 것들이 되어야하는가?

- ☐ 다음 타켓 url 에 대한 string 값(`url`)
- ☐ `url`, `page`
- ☐ `url`, `end_quote`
- ☐ `url`, `start_link`

procedure



```
page = ... contents of some web page ...
start_link = page.find('<a href=')
start_quote = page.find('"' , start_link)
end_quote = page.find('"' , start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

```
start_link = page.find('<a href=')
start_quote = page.find('"' , start_link)
end_quote = page.find('"' , start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

```
start_link = page.find('<a href=')
start_quote = page.find('"' , start_link)
end_quote = page.find('"' , start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

Return Statement

```
page = ... contents of some web page ...
start_link = page.find('<a href=')
start_quote = page.find('"' , start_link)
end_quote = page.find('"' , start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

```
start_link = page.find('<a href=')
start_quote = page.find('"' , start_link)
end_quote = page.find('"' , start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```

```
start_link = page.find('<a href=')
start_quote = page.find('"' , start_link)
end_quote = page.find('"' , start_quote + 1)
url = page[start_quote + 1 : end_quote]
print(url)
page = page[end_quote:]
```



procedure

def get_next_target(s):

```
start_link = s.find('<a href=')
start_quote = s.find('"' , start_link)
end_quote = s.find('"' , start_quote + 1)
url = page[start_quote + 1 : end_quote]
```

return <expression>, <expression>



Quiz: Return Statement

빈칸을 채우시오.

```
def get_next_target(page):
```

```
    start_link = page.find('<a href=')
    start_quote = page.find('"', start_link)
    end_quote = page.find('"', start_quote + 1)
    url = page[start_quote + 1 : end_quote]
```

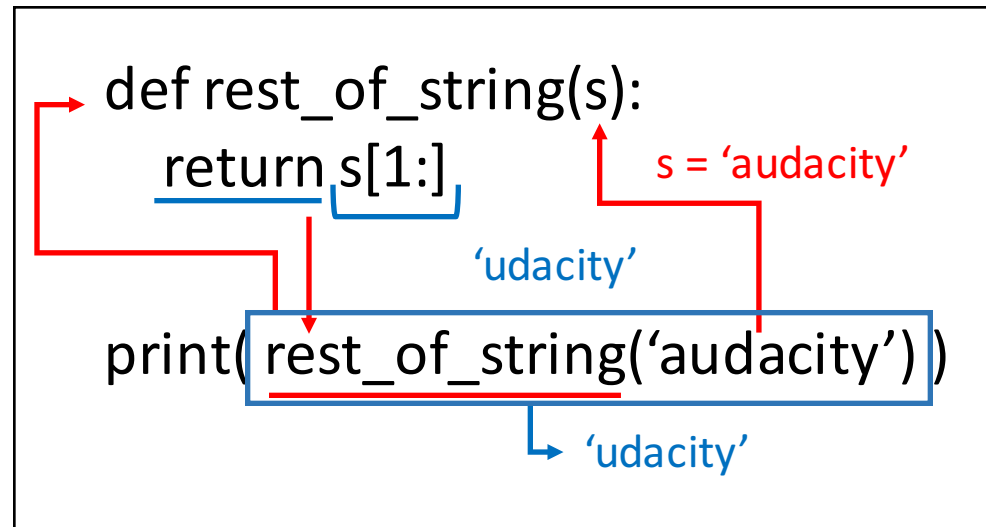
```
    return url , end_quote
```

Using Procedures

<procedure> (<input>, <input>, ...)

operands

arguments



Quiz: Inc Procedure

아래에 정의된 `inc` 프로시저는 보기 중 어떤 작업을 수행하는가?

```
def inc(n):  
    return n + 1
```

- ☐ Nothing!
- ☐ number 값을 입력으로 받아, 그 number 값에 1을 더한 값을 출력한다.
- ☐ number 값을 입력으로 받아, 동일한 number 값을 출력한다.
- ☐ 두개의 number 값을 입력으로 받아, 두 값의 합을 출력한다.

Quiz: Sum Procedure

아래에 정의된 `sum` 프로시저는 보기 중 어떤 작업들을 수행하는가?

```
def sum(a, b):  
    a = a + b
```

- ☐ Nothing
- ☐ 두개의 number 값을 입력으로 받아, 그 값들의 합을 출력한다.
- ☐ 두 string 값을 입력으로 받아, 두 string 값의 결합 값을 출력한다.
- ☐ 두개의 number 값을 입력으로 받아, 첫번째 입력 값을 두 number 값의 합으로 교체한다.

Sum Procedure

code

```
def sum(a, b):  
    a = a + b  
  
print( sum(1, 1) )
```

result

None

code

```
def sum(a, b):  
    print( "enter sum!" )  
    print( "a is ", a )  
    a = a + b  
    print("a is ", a)  
  
print( sum(2, 123) )
```

result

enter sum!
a is 2
a is 125
None

code

```
def sum(a, b):  
    a = a + b  
    return a  
  
print( sum(2, 123) )
```

result

125

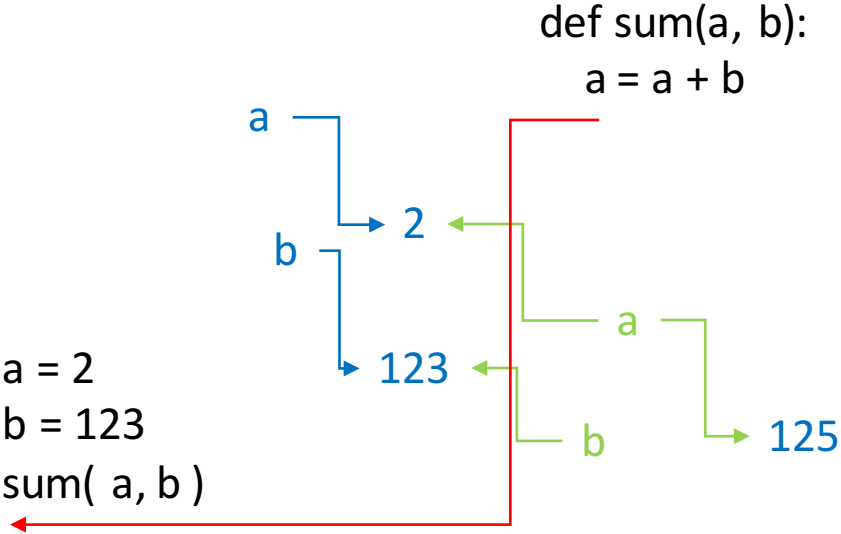
Sum Procedure

code

```
def sum(a, b):  
    a = a + b  
  
a = 2  
b = 123  
print( sum(a, b) )  
print( a )
```

result

```
None  
2
```



Quiz: Sum Procedure with a Return Statement

아래에 정의된 `sum` 프로시저는 보기 중 어떤 작업들을 수행하는가?

```
def sum(a, b):  
    a = a + b  
    return a
```

☒ ~~Nothing~~

- ☐ 두개의 number 값을 입력으로 받아, 그 값들의 합을 출력한다.
- ☐ 두 string 값을 입력으로 받아, 두 string 값의 결합 값을 출력한다.
- ☐ 두개의 number 값을 입력으로 받아, 첫번째 입력 값을 두 number 값의 합으로 교체한다.

Sum Procedure with a Return Statement

code

```
def sum(a, b):  
    a = a + b  
    return a  
  
s = 'Hello '  
t = 'Dave!'  
print( sum(s, t) )  
print( s )
```

result

```
Hello Dave!  
Hello
```


Quiz: Square

`square` 라는 이름의 프로시저를 정의하여, 하나의 숫자 입력을 받아서, 그 숫자의 제곱을 출력하게 하시오.

```
print( square(5) ) -> 25
```

Square

code

```
def square(n):  
    return n * n  
  
x = 37  
print( square(x) )
```

result

1369

Square

code

```
def square(n):  
    return n * n  
  
x = 37  
y = square(x)  
print( square(y) )
```

result

1874161

Sum Procedure

code

```
def square(n):  
    return n * n  
  
x = 37  
print( square(square(x)) )  
  
    procedure composition
```

result

1874161

Quiz: Sum of Three

`sum3` 라는 이름의 프로시저를 정의하여, 세개의 입력을 받아서, 세개의 숫자들의 합을 출력하게 하시오.

```
print( sum3(1,2,3) ) -> 6
```

Quiz: Abbaize

`abbaize` 라는 이름의 프로시저를 정의하여, 두개의 문자열을 입력으로 받고, 첫번째 입력값, 두번째 입력값의 두번 반복, 다시 첫번째 입력값 순서대로 합친 문자열을 출력하도록 하시오.

`abbaize('a', 'b') -> 'abba'`

`abbaize('dog', 'cat') -> 'dogcatcatdog'`

Abbaize

code

```
def abbaize(a, b):  
    return a + b + b + a  
  
print( abbaize('a', 'b')) )  
print( abbaize('dog', 'cat')) )
```

result

```
abba  
dogcatcatdog
```

Quiz: Find Second

`find_second` 라는 이름의 프로시저를 정의하여,
두개의 문자열(검색 대상 문자열, 검색할 문자열)을 입력으로 받으시오.
검색할 문자열이 검색 대상 문자열에서 두번째로 나타나는 위치를 출력하시오.

code

```
danton = "De l'audace, encore de l'audace, toujours de l'audace"  
  
print( find_second(danton, 'audace' ) )
```

result

25

Find Second

code

```
def find_second(search, target):  
    first = search.find(target)  
    second = search.find(target, first + 1)  
    return second  
  
danton = "De l'audace, encore de l'audace, toujours de l'audace"  
  
print( find_second(danton, 'audace' ) )
```

result

25