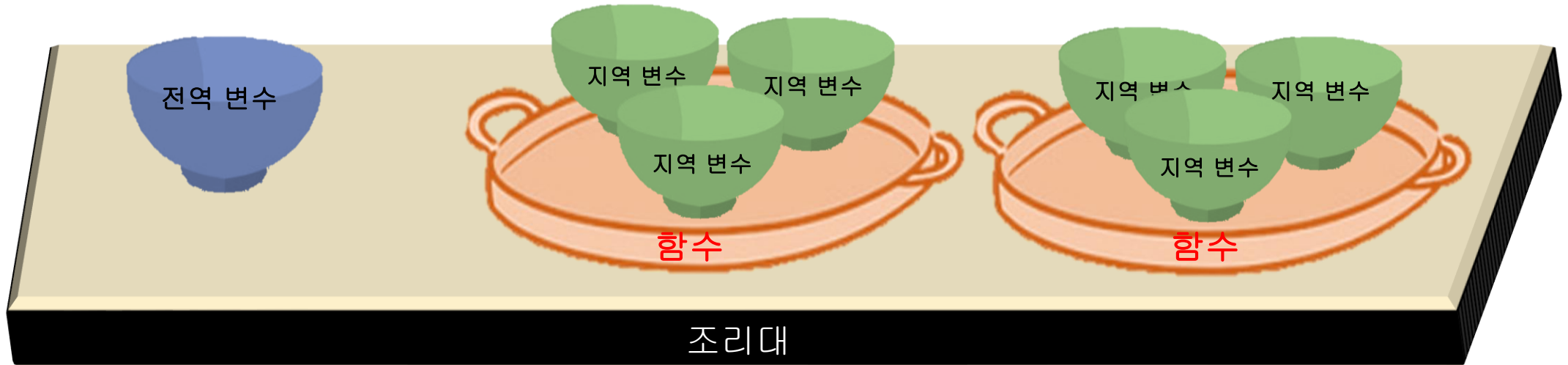


LESSON

변수의 유효 범위(scope) 지역 변수와 전역 변수

- 지역 변수(local variable)
 - **함수 안에 정의된 변수**로 함수 안에서만 사용할 수 있다. 즉, 외부에서는 접근할 수 없다.
 - 지역 변수의 생존 기간은 함수가 호출될 때 생성되고 함수가 종료되면 소멸되어 더 이상 사용할 수 없다.
- 전역 변수(global variable)
 - **함수의 외부에 정의된 변수**로 프로그램 전체에서 사용될 수 있다.

변수의 유효 범위(scope)



조리대에 **쟁반은 함수**. **쟁반위에 있는 그릇은 지역 변수**.
쟁반을 조리대에서 치우면 쟁반 위에 그릇들도 모두 없어진다.
조리대 위에 있는 그릇은 전역 변수.

```
width=10  
height=20
```

함수 외부에
선언되었으므로 전역
변수이다.

```
def area(w,h):  
    result=w*h  
    return result
```

함수 내부에
선언되었으므로 지역
변수이다.

```
print(area(width,height))
```


```
def f():  
    a=10  
    print(f'함수 내부:{a}')
```

f()

함수 내부:10

```
def f():  
    a=10  
    print(f'함수 내부:{a}')
```

f()
print(f'함수 외부:{a}')



함수 내부:10

NameError: name 'a' is not defined

```
def area(w,h):  
    result=w*h  
    return result
```

함수의 매개 변수도
일종의 지역 변수이다.

```
print(area(10,20))  
print(w))      # 오류 발생
```

```
def f():  
    s='바나나'  
    print(f'함수 내부:{s}')
```



```
s='사과'  
f()  
print(f'함수 외부:{s}')
```

함수 내부:바나나
함수 외부:사과



```
a=10
```

```
def f():
```

```
    print(f'함수 내부:{a}')
```

```
f()
```

```
print(f'함수 외부:{a}')
```

함수 내부:10
함수 외부:10



1. 함수 내부를 검색한다. 내부에 a변수가 없다.
2. 함수 외부로 나가서 찾는다.



쟁반(함수) 위에 그릇(변수)이 없으면 조리대에 있는 그릇(변수)을 사용하면 된다.


```
def f():  
    print(f'함수 내부:{s}')
```

```
s='사과'  
f()  
print(f'함수 외부:{s}')
```



내부에 지역 변수 s가
있지만 선언부터 해야
사용할 수 있다.

```
UnboundLocalError: cannot access local variable 's' where it is not associated with a  
value
```

```
def f():  
    global s  
    print(f'함수 내부:{s}')    s='바나나'  
    print(f'함수 내부:{s}')
```

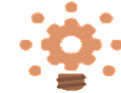
```
s='사과'  
f()  
print(f'함수 외부:{s}')
```

전역 변수 s를 사용하겠다고
파이썬 인터프리터에게
알려줘야 한다.

전역 변수가 변경된다.

```
함수 내부:사과  
함수 내부:바나나  
함수 외부:바나나
```

- 전역변수는 모든 함수에서 접근이 가능하기 때문에, 다른 함수에서의 전역 변수의 변경에 의해 현재 함수의 결과가 변경될 수 있다. 따라서 디버깅이 어렵고 프로그램의 복잡도가 증가된다.



함수는 독립된 객체로 존재하는 것이 좋다. 외부 변수에 종속적인 함수는 바람직하지 않다.

```
PI=3.141592656358979
def main():
    radius=float(input('원의 반지름 입력:'))
    print('원의 면적:',area(radius))
    print('원의 둘레:',circumference(radius))

def area(radius):
    return PI**radius

def circumference(radius):
    return 2*PI*radius

main()
```

단, 상수를 정의할 때는 전역으로 선언하면 공간의 낭비를 줄일 수 있다.

- 네임스페이스(name space)란?
 - 프로그래밍시 변수나 함수의 이름을 중복되지 않게 해야 하는데 규모가 있는 프로그래밍에서 사실상 어려운 문제가 된다. 이를 해결하기 위해 도입된 개념으로 같은 이름이라도 다른 객체를 가리키는 게 가능하게 만들어 준다.
 - 파이썬에서 모든 객체는 서로 구분되는 공간들 중 하나에 속하도록 설계되어 있다.
- 파이썬의 네임스페이스
 - local name space: 함수 별로 존재하며, 함수 내의 지역 변수들의 이름들이 소속된다.
 - global name space: 사용중인 파일에 사용되는 변수, 함수, 클래스, import된 객체들의 이름들이 소속된다.
 - built-in name space: 파이썬이 구동과 함께 끝날 때까지 사용할 수 있는 것들로 구성된다. 파이썬 내장함수와 예외들이 소속된다.

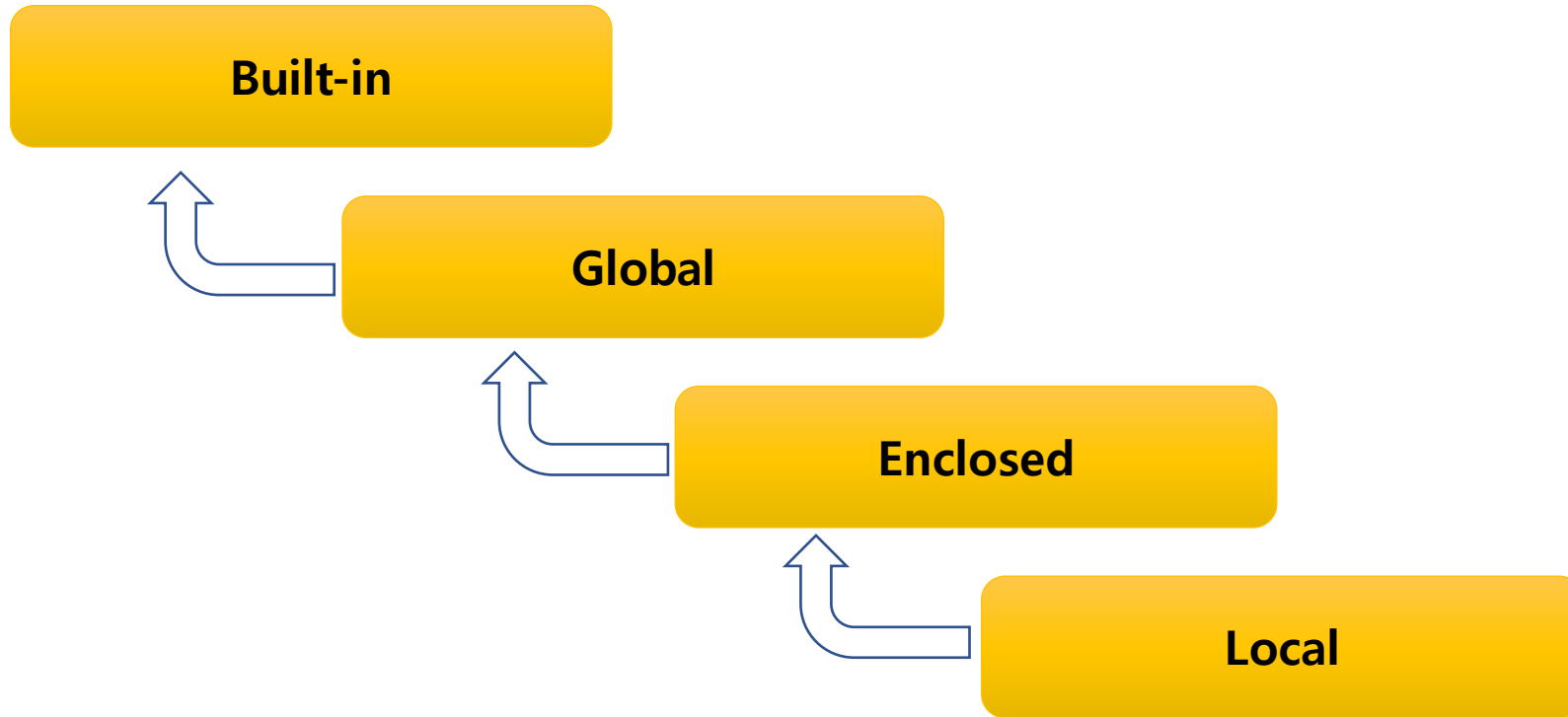


네임스페이스 확인

- `locals()`
- `globals()`
- `dir(__builtins__)`

변수의 유효범위(variable scope)

- 변수의 유효 범위를 계산할 때 네임스페이스를 기반으로 한다.



LEGB : 변수를 확인할 때 순서를 나열한 것.
Local->Enclosed->Global->Built-in 순으로 검색함.

```
def calc():  
    base=10  
  
    def square(x):  
        print('여기는 calc안에 square:',base)  
        return base**x  
  
    return square  
  
f=calc()  
print(f(2))
```

여기는 calc안에 square: 100
100