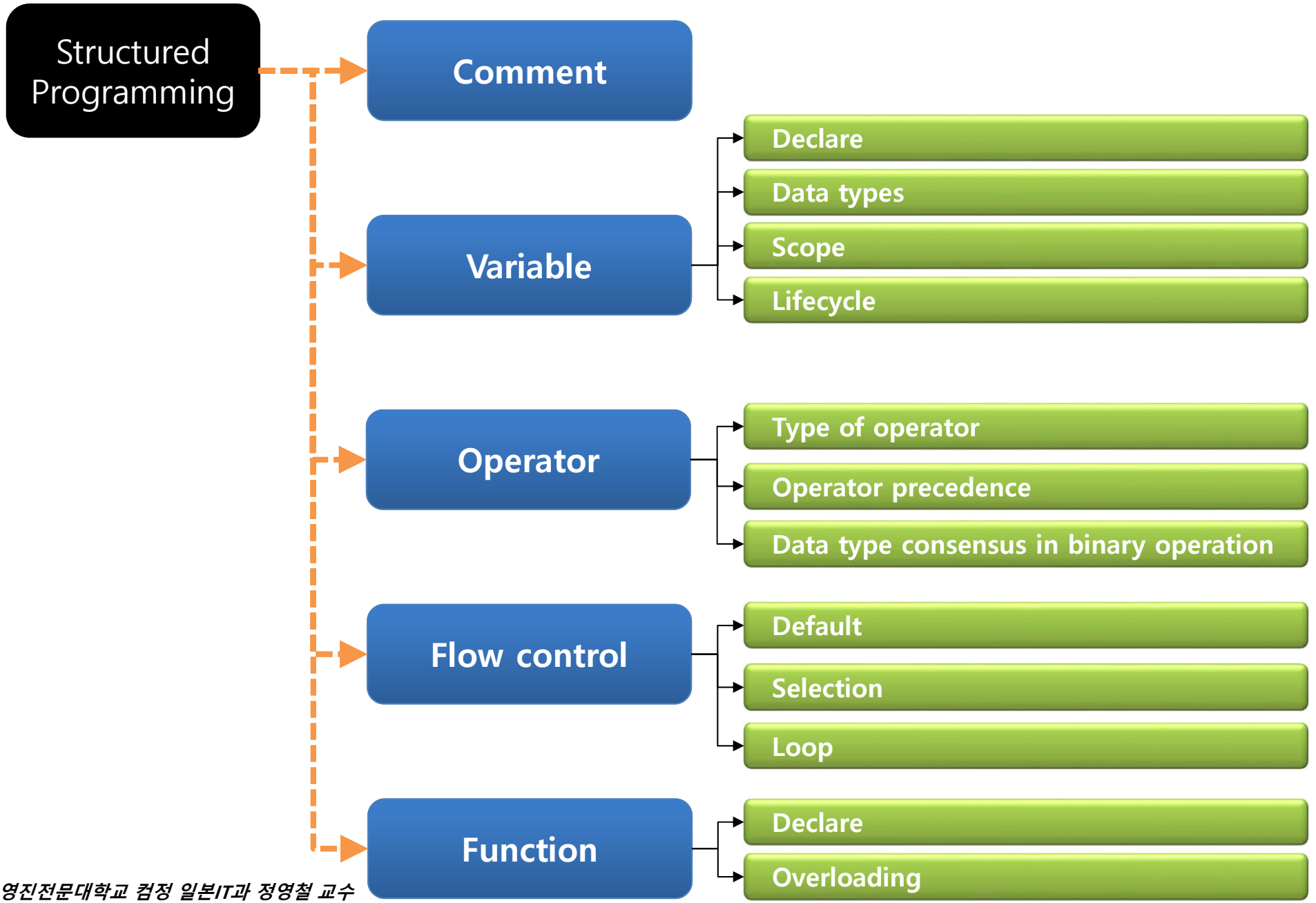
주문식교육의 산실
영진전문대학교

구조적 프로그래밍 구성 요소 (함수)

일본IT과

정영철 교수

구조적 프로그래밍 구성 요소



함수(Function) or Subroutine 란?

- 특정 알고리즘을 "함수"라는 단위로 묶어 반복적으로 호출하여 사용

- 장점

  - 프로그램을 기능별로 나누어 구현 가능
  - 프로그램의 유지 보수의 효율성 증대
  - 프로그램의 확장성 제고

Non-Functional program

```
1  # N개의 정수를 입력 받아 오름 차순으로 정렬
2
3  N = 5   # 입력 정수 갯수
4
5  inputValueList = [] # 입력 값 리스트
6
7  # 키보드로 부터 N개의 정수를 입력 받아 리스트에 저장
8  for index in range(N):
9      inputValue = input(str(index +  1) + "번째 입력 값")
10     inputValueList.append(int(inputValue))
11
12 # 리스트에 저장된 정수 값을 오름 차순으로 정렬
13 for iCount in range(N - 1):
14     for jCount in range(N - iCount - 1):
15         if inputValueList[jCount] > inputValueList[jCount + 1]:
16             temp = inputValueList[jCount + 1]
17             inputValueList[jCount + 1] = inputValueList[jCount]
18             inputValueList[jCount] = temp
19
20 # 리스트 값 출력
21 print(inputValueList)
```

```
1번째 입력 값7
2번째 입력 값0
3번째 입력 값9
4번째 입력 값2
5번째 입력 값4
[0, 2, 4, 7, 9]
```

Functional Program

```python
# 키보드로 부터 N개의 정수를 입력 받아 리스트에 저장
def getInputValue(argNumOfInputValue):
    tempInputValueList = []
    for index in range(argNumOfInputValue):
        inputValue = input(str(index +  1) + "번째 입력 값")
        tempInputValueList.append(int(inputValue))

    return tempInputValueList

# 리스트에 저장된 정수 값을 오름 차순으로 정렬
def myBubbleSort(argList, asscend = True):
    for iCount in range(len(argList) - 1):
        for jCount in range(len(argList) - iCount - 1):
            comparisonResult = \
                argList[jCount] > argList[jCount + 1] if asscend else argList[jCount] < argList[jCount + 1]

            if comparisonResult:
                temp = argList[jCount + 1]
                argList[jCount + 1] = argList[jCount]
                argList[jCount] = temp

# N개의 정수를 입력 받아 오름 차순으로 정렬

# 5개의 정수를 키보드로부터 입력 후 리스트 저장
inputValueList = getInputValue(5)

# 리스트 값 정렬
myBubbleSort(inputValueList)

# 리스트 값 출력
print(inputValueList)
```

함수 정의 및 사용 방법 : 함수 정의 방법

함수 정의

함수 호출

```
1  # 함수 정의 방법
2  def 함수이름 (매개변수) :
3      # 함수 호출 시 실행 될 알고리즘 구문
4      # 함수 호출 시 실행 될 알고리즘 구문
5      # 반환 값 - 반환 값은 옵션 (반환 값 생략 가능)
6
7
8  # 함수 호출
9  #  - 함수를 호출하기 위해서는
10 #    호출 구문이 선언부 보다 앞서 위치해야 됨
11 함수이름(인자값)
```

✓ **매개변수 (Parameter)** : 함수 정의 시 선언된 변수를 지칭
Ex) 3번 라인 sum 함수의 argA, argB 변수

✓ **인자 값 (Argument)** : 함수 호출 시 매개변수에 전달되는 값
Ex) 9번 라인 sum 함수 호출 시 인자 값 2, 3

```
1  # sum 함수 정의
2  # 두 개의 수를 입력 받아 더한 값을 반환
3  def sum(argA, argB):
4      result = argA + argB
5
6      return result
7
8  # sum 함수 호출 - 인자 값 2, 3
9  print(sum(2, 3)) # 결과 값 5
```

함수 정의 및 사용 방법 : 매개변수와 반환 값 (1)

```
1  # 매개 변수와 반환 값은 필요에 따라 생략 가능
2
3  # 매개 변수와 매개 변수가 없는 함수 예제
4  def printHello() :
5      print("hello")
6
7  printHello() # hello
```

함수 정의 및 사용 방법 : 매개변수와 반환 값 (2)

```
1  # 함수 반환 값 예제
2
3  # 평균을 구하는 함수
4  def average(argList):
5      sum = 0
6      for value in argList:
7          sum += value
8
9      return sum/len(argList)
10
11
12 myList = [7, 8, 6, 8]
13
14 # average 함수 호출 후 반환 값을 avg 변수에 저장
15 avg = average(myList)
16
17 print(avg)
```

함수 정의 및 사용 방법 : 매개변수와 반환 값 (3)

```python
# Python의 경우 특이하게 함수의 반환 값이
# 두 개 이상일 경우 Tuple 형으로 반환

def sumAndMultiply(argA, argB):
    resultSum = argA + argB
    resultMultiply = argA * argB
    # 반환 값이 두 개 이상일 경우 Tuple형으로 반환
    # Tuple은 List와 같음, 단 원소 내용은 변경 불가
    return resultSum, resultMultiply

# sumAndMultiply 함수 호출 후 Tuple형으로 반환 값 획득
result = sumAndMultiply(2, 3)
print(result) # (5, 6)


sum, multiply = sumAndMultiply(4, 5)
print(sum, multiply) # 9 20
```

Call-by-value vs Call-by-reference

```
1  def foo(a):
2      a = a + 2
3
4  value = 1
5
6  # Call by value
7  # Primitive variables -> int, float, string, boolean
8  foo(value)
```

Call-by-value vs Call-by-reference

```
12  def bar(a):
13      a.append(1)
14
15  value = [2, 3]
16
17  # Call by reference
18  # Reference variables -> Object -> List, Tuple, Dicitionary
19  bar(value)
20
21  print(value) # 2, 3, 1
```

Arguments type of function in python (1)

```
1  # positional argument
2  def foo (a, b, c):
3      print(a, b, c)
4
5  foo(1, 2, 3) # 1, 2, 3
6
7  # keyword argument
8  def bar (a, b, c):
9      print(a, b, c)
10
11 bar(b=2, a=1, c=3) # 1, 2, 3
12
13 # default argument
14 def han (a, b=2, c=3):
15     print(a, b, c)
16
17 han(1) # 1, 2, 3
```

Arguments type of function in python (2)

```
19  # arbitrary positional arguments
20  def pos(*a):
21      print(type(a))
22      for value in a:
23          print(value)
24
25  pos(1, 2, 3)
26
27  # arbitrary keyword arguments
28  def fox(**a):
29      print(type(a))
30      for k, v in a.items():
31          print(k, v)
32
33  fox(name="youngchul", gender="male")
```

Q/A

감사합니다

주문식교육의 산실

영진전문대학교