COMP636 – Software Development

Learning Session 3a: Functions

4 ways of interacting with functions

1. Just code

my_function()

<do something>
..some
..code
..here

2. Code + input

in

my_function(input)

<do something
using **input**>
..some
..code
..here

3. Code + output

my_function()

<do something>
..some
..code
..here

return **output**

out

4. Code + input + output

in

my_function(input)
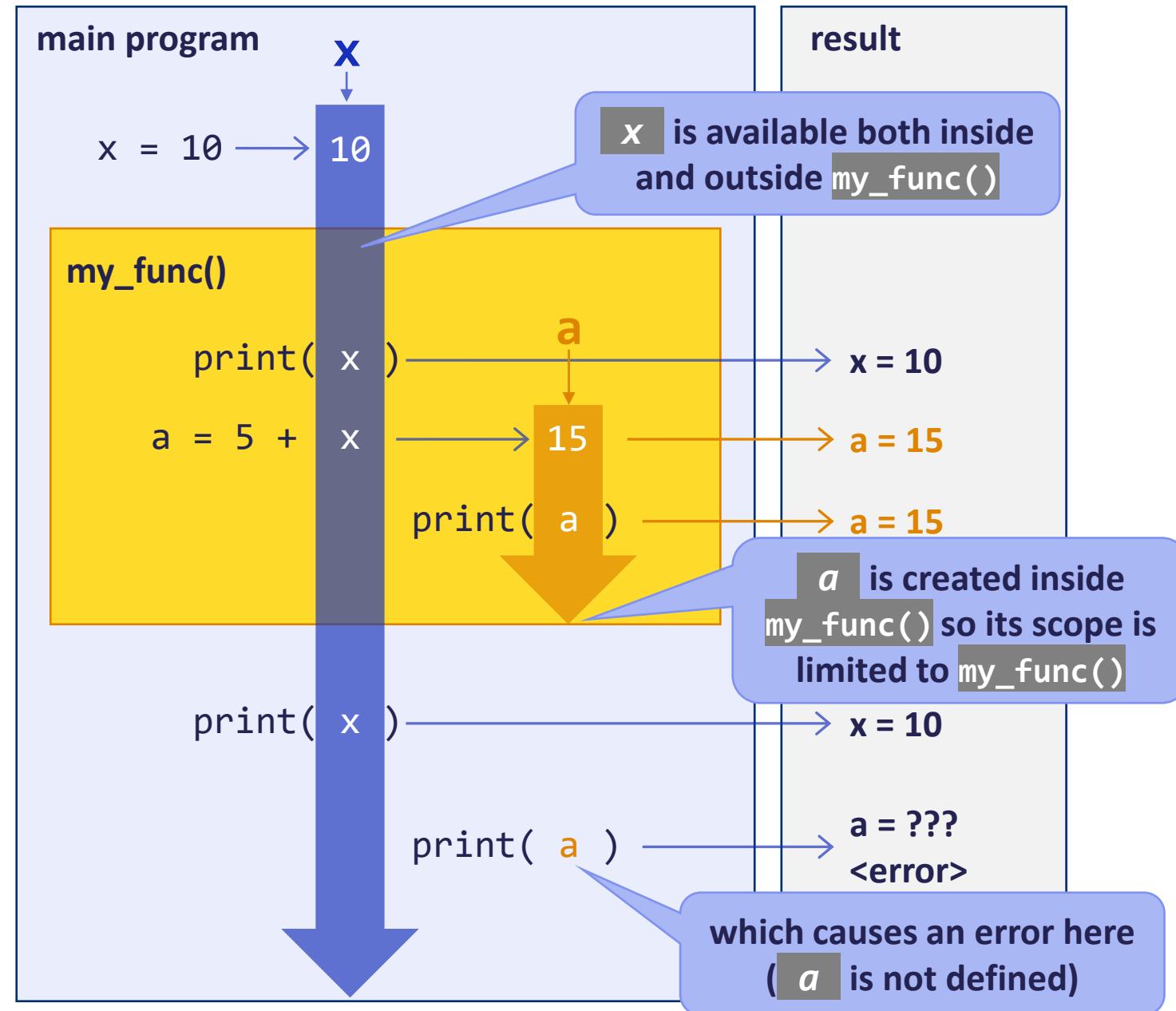
<do something
using **input**>
..some
..code
..here

return **output**
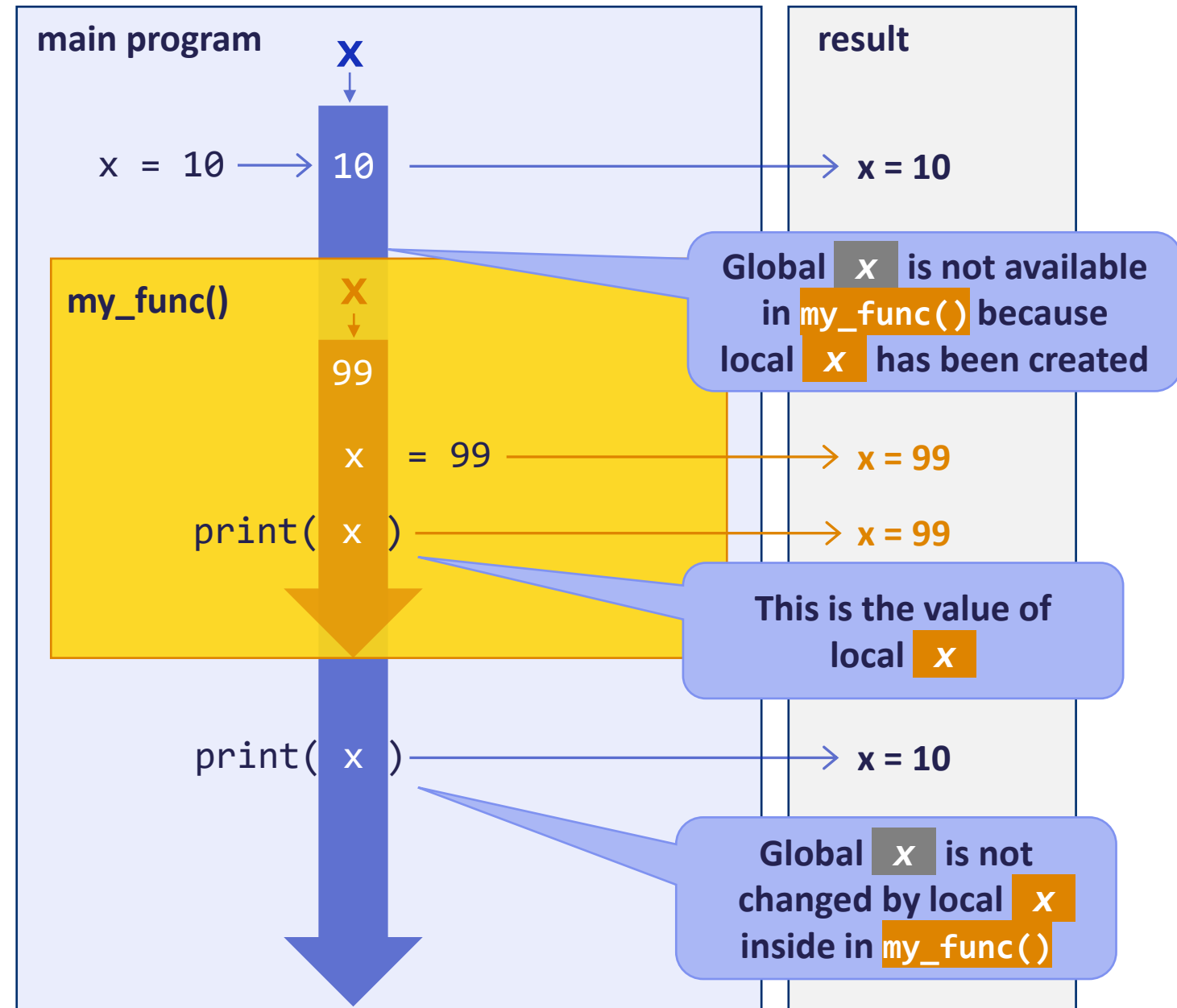
out

Variable 'scope'

```
def my_func():
    print(x)  #  x = 10
    a = 5 + x
    print(a)  #  a = 15

x = 10

my_func()      # call function

print(x)       #  x = 10

print(a)       # error a not known
```

main program

x

x = 10 ⟶ 10

my_func()

print(x)

a

a = 5 + x ⟶ 15

print(a)

print(x)

print(a)

result

x = 10

a = 15

a = 15

x = 10

a = ???
<error>

x is available both inside and outside `my_func()`

a is created inside `my_func()` so its scope is limited to `my_func()`

which causes an error here (*a* is not defined)

Variable 'scope'

```python
def my_func():
    x = 99
    print(x)  #  a = 99


x = 10

my_func()      # call function

print(x)       #  x = 10
```


main program

x

x = 10 ⟶ 10 ⟶ x = 10

result

my_func()

x

99

x = 99 ⟶ x = 99

print(x) ⟶ x = 99

Global **x** is not available in **my_func()** because local **x** has been created

This is the value of local **x**

print(x) ⟶ x = 10

Global **x** is not changed by local **x** inside in **my_func()**

Variable 'scope'

Rules:
- A variable assigned inside a function is not available outside that function
  - Even if it has the same name as a global variable
  - *Unless* `global <variable_name>` precedes the assignment within the function
- Global variables with a value before the function call can be *read* inside the function, but not altered (without the `global` keyword, as above)