CSE 8A: Intro to Programming in Python

Fall 2021

Lecture 14 - memory model exercises

UC San Diego

Stack Frames

Every time a function is invoked (i.e., called), the invocation gets a new "frame" for holding variables

- The parameters also exist in a frame
- When a variable name is used within a function, Python looks for it in the current frame first
- We call these variables local variables

Global frame

- There is always one global frame that all functions can access
- When a variable name is used in a function, Python looks two places:
  1. the function invocation's frame (first)
  2. the global frame (only if not found before)

Stack frames key points

1. Python doesn't evaluate a function until it is called
2. First line to execute in a python code is the first statement that isn't part of any functions
3. Function returns when the last statement of the function is executed or when a return statement is executed
4. Function returns to its caller
5. A function stack frame is created when a function is called, and is destroyed when a function returns to its caller
6. A function can only access variables in its own frame or variables in the global frame
7. Local variables take precedence than the global variables

Global variable

- Variables that are declared outside any functions
- Inside a function, you can use global keyword to tell python that a certain variable is global so it doesn't create a local variable instead

Global variable

```
name = ‘christine’ #a global variable

def foo():
  name = ‘paul’ #a local variable

foo()
print(name) #try to print the global variable
```

vs

```
name = ‘christine’ #a global variable

def foo():
  global name
  name = ‘paul’ #use the global variable

foo()
print(name) #try to print the global variable
```

Exercise: Modifying Global Variables in Functions

What will happen when we run this code?

```
msg = 'hello'
def greeting():
    global msg
    msg = 'welcome!'
    print('greeting: ' + msg)

print('before: ' + msg)
greeting()
print('after: ' + msg)
```

A)
before: hello
greeting: welcome!
after: hello

B)
before: hello
greeting: welcome!
after: welcome!

C) Error: variable msg is not defined

D) I don't know! :(

Exercise: Passing Parameters/Arguments to Functions

What will happen when we run this code?

```
def f(x):
    x = 'B'
    print('inside: ' + x)

val = 'A'
print('before: ' + val)
f(val)
print('after: ' + val)
```

A)
before: A
inside: B
after: B

B)
before: A
inside: B
after: A

C) Error: variable x is not defined

D) I don't know! :(

Exercise: Passing Parameters/Arguments to Functions

What will happen when we run this code?

```
def f(x):
    x = 'B'
    print('inside: ' + x)
x = 'A'
print('before: ' + x)
f(x)
print('after: ' + x)
```

A)
before: A
inside: B
after: B

B)
before: A
inside: B
after: A

C) Error: variable x is not defined

D) I don’t know! :(

Function Chain Calls

What is printed out when the following code executes?

```
def foo():
    print('A')
    fubar()
    print('B')

def fubar():
    print('C')
    bar()
    print('D')

def bar():
    print('E')

foo()
```

A)
A
B
C
D
E

B)
A
B

C)
A
C
E

D)
A
C
E
D
B

E) None of the answers is correct

Function Chain Calls

```
def foo():
    print('A')
    fubar()
    print('B')

def fubar():
    print('C')
    bar()
    print('D')

def bar():
    print('E')

foo()
```

Pass a list to a function

```
def add_fish(names, new_fish):
    names.append(new_fish)

fishes = ['carp', 'dolphin', 'shark']
add_fish(fishes, 'whale')
print(fishes)
```

A. [‘carp’, ‘dolphin’, ‘shark’]
B. [‘carp’, ‘dolphin’, ‘shark’, ‘whale’]
C. [‘whale’]
D. [ ]
E. None of the given choices is correct

```python
def add_fish(names, new_fish):
    names.append(new_fish)

fishes = ['carp', 'dolphin', 'shark']
add_fish(fishes, 'whale')
print(fishes)
```

Pass a list to a function

```
def add_fish(names, new_fish):
    names = ['turtle', 'jelly fish']
    names.append(new_fish)

fishes = ['carp', 'dophine', 'shark']
add_fish(fishes, 'whale')
print(fishes)
```

A. [‘carp’, ‘dolphin’, ‘shark’]
B. [‘carp’, ‘dolphin’, ‘shark’, ‘whale’]
C. [‘turtle’, ‘jelly fish’, ‘whale’]
D. [ ]
E. None of the given choices is correct

```
def add_fish(names, new_fish):
    names = ['turtle', 'jelly fish']
    names.append(new_fish)

fishes = ['carp', 'dophine', 'shark']
add_fish(fishes, 'whale')
print(fishes)
```