CS 61λ

Discussion 1

Link: cs61a.org/disc/disc01.pdf

Welcome Survey: cs61a.amks.me/dis1

Song Rec:

Here Comes The Sun

by The Beatles

Announcements

Note On Preparing For Exams

arushisomani@berkeley.edu

amks.me/zoom

cs61a.amks.me

Discussion 1

Control and Environments

Expressions vs Statements

Expressions

Evaluate to values

```
>>> 3 + 4 + 2 + 1
10
>>> “hello”
“hello”
```

Statements

Do not evaluate to values, instead determine some kind of change in control flow.

```
>>> a = 3
>>>
```

Control Flow

```
a = 3
print
(a)
```

Control Flow with IF

Conditional execution of statements.

```
if (3 > 4):
    print (1)
else:
    print (2)
```

Control Flow with AND/OR

AND returns the first False value or last True value.

```
3 and 4 and 0 and 1
```

OR returns the first True value or last False value.

```
0 or False or 1 or 0
```

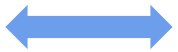
Control Flow with WHILE

While loops create iterative control flow.

```
n = 1

while (n > 3):
   print (n)
   n = n +1

print("End")
```



```
n = 1

print (n)
n = n +1
print (n)
n = n +1

print("End")
```

Question 1.1

```
def wears_jacket_with_if(temp, raining):
    """
    >>> wears_jacket_with_if(90, False)
    False
    >>> wears_jacket_with_if(40, False)
    True
    >>> wears_jacket_with_if(100, True)
    True
    """
```

Question 1.1.2

```
def wears_jacket(temp, raining):
```

Question 1.2

1.2 What is the result of evaluating the following code?

```
def square(x):
    print("here!")
    return x * x

def so_slow(num):
    x = num
    while x > 0:
        x = x + 1
    return x / 0

square(so_slow(5))
```

Question 1.3

```
def is_prime(n):
    """
    >>> is_prime(10)
    False
    >>> is_prime(7)
    True
    """
```

Anatomy of a Function

```
def func(x, y):
    print(“Hi”)
    print(“Hello”)
    return x+y

func(2, 3)
```

Functions are outside of Control Flow

```
def f(x):
    print (“Inside”)
    return 1/0

print(“Outside”)
print(f)
```

How Functions Are Evaluated

```
def f(x):
    print (“Inside”)
    return x + 1


f(3)
```

Function Frames Are Independent

```
def f(x):
    a = 3
    return x + 1

a = 4
f(3)
print(a)
print(x)
```

Function Frames Can Access Parent Frame Bindings

```
def f(x):
    b = a + 3
    return x + 1


a = 4
f(3)
print(a)
print(b)
```

Question 2.2

2.2 Use these rules and the rules for assignment statements to draw a diagram for the code below.

```
def double(x):
    return x * 2

def triple(x):
    return x * 3

hat = double
double = triple
```

Question 2.3

```
def double(x):
    return x * 2

hmmm = double
wow = double(3)
hmmm(wow)
```

Question 2.4

```
def f(x):
        return x

def g(x, y):
        if x(y):
                return not y
        return y

x = 3
x = g(f, x)
f = g(f, 0)
```

Prints In Prints

```
print(a=3)
```

```
print(print(2))
```

Preview Of Next Week

Higher Order Functions

1. Functions accept/return python values
2. Functions are python values
3. Functions can accept/return functions

Higher Order Functions

```
def make_adder(add_value):
    def adder(input_val):
        return add_value + input_val
    return adder


make_adder(1)(3)
```

Higher Order Functions

```
def compose(func1, func2):
    def f(x):
        return func1(func2(x))
    return f


def add_one(x):
   return x+1
def mul_two(x):
   return 2*x


compose(add_one, mul_two)(3)
```

Quiz!

```
x = 3

def p(rint):
        print(rint)

def g(x, y):
        if x:
                print("one")
        elif x:
                print(True, x) # Does x being truth-y affect the printed value?
        if y:
                print(True, y) # Does y being truth-y affect the printed value?
        else:
                print(False, y) # Does y being false-y affect the printed value?
        return print(p(y)) + x
```

Quiz!

Expression	Interactive Output
`print(4, 5) + 1`	`4 5` `Error`
`2 * 2 * 1 + x * x`	
`print(3 * 3 * 1)`	
`print(x + 1 * x + 1)`	
`print(print(x + 1 * x + 1))`	
`print(print(x + 1 * x + 1) + 1)`	
`print(p("rint"))`	
`x, y = 2, x` `g(y, x)`	
`g(y, p("rint"))`	

Quiz!

Quiz!

Quiz!

Feedback!

cs61a.amks.me/feedback