National University of Singapore
School of Computing
CS1010S: Programming Methodology

Extra Practice 1 Solutions

Question 1

Let's start with a basic code tracing.

```
x = 1

def foo(x):
    return x + 1

print(foo(2))   # 3
print(foo(x))   # 2
print(x)        # 1
```

Question 2

How about this one?

```
x = 1

def add_one(x):
    print(x + 1)     # note that this prints x + 1 and returns None

print(add_one(2))    # 3
                     # None
y = add_one(x)       # 2
                     # y = None
print(add_one(y))    # TypeError
```

Question 3

Give the output of the following function call.

```
x = 10
def ping(x):
    return pong(x + 4)
def pong(x):
    x += 1
    return x ** 2

print(ping(3))   # 64
print(pong(x))   # 121
ping(2)          # Nothing is printed!
```

Question 4

Give the output of the following function call.

```
x, y = 1, 4
x, y = y, x      # This swaps x and y, so x = 4 and y = 1 now

def ding(x):
    if x % 2 == 1:
        print("Alright")
    elif x % 3 == 1:
        print("Okay")
    if x ** 0.5 == y + 1:
        print("Awesome")
    else:
        print("This question sucks")

ding(x)          # Okay
                 # Awesome
ding(y)          # Alright
                 # This question sucks
ding(9)          # Alright
                 # This question sucks
print(ding(3))   # ding(3) will print "Alright" and "This question sucks"
                 # Then it will print None
```

Question 5

Give the output of the following function call.

```
def check(word):
    if len(word) >= 3:
        print("gg")
    if word[0] == word[-1]:
        print("cool")
    elif word[::2] == "cdc":
        print("nice")
    else:
        print("end?")
    if word[3::-1] == "edoc":
        print("not yet")
    else:
        print("end now")
check("codec")       # gg
                     # cool
                     # not yet
check("codecs")      # gg
                     # nice
                     # not yet
check("ar")          # end?
                     # end now
```

Question 6

(a) Define a function named `total_legs` that takes in two inputs - the number of chickens and the number of cows, and returns the total number of legs in total. **Sample Execution:**

```
>>> total_legs(2, 2)
12
>>> total_legs(1, 4)
18
>>> total_legs(0, 1)
4
```

Solution:

```
def total_legs(chickens, cows):
    return 2*chickens + 4*cows
```

(b) A tax is imposed on a farm that charges the farmer $2 for every leg present on the farm. Define a function named `tax_count` that takes in two inputs - the number of chickens and the number of cows, and returns the amount of tax that the farmer needs to pay. Use your previously-defined function(s). :)
Sample Execution:

```
>>> tax_count(2, 2)
24
>>> tax_count(0, 2)
16
```

Solution:

```
def tax_count(chickens, cows):
    return 2*total_legs(chickens, cows)
```

(c) The farmer wants to see if the total number of animals he has on his farm exceeds 10. Otherwise, he needs to pay $5 more as tax. Let's define a function named `too_many` that takes in two inputs - the number of chickens and the number of cows, and returns `True/False` depending on whether the total number of animals exceeds 10.
Sample Execution:

```
>>> too_many(2, 2)
False
>>> too_many(7, 4)
True
```

Solution:

```
def too_many(chickens, cows):
    return chickens + cows > 10
```

(d) Now, we want to find the total amount that the farmer needs to pay in total as tax. Define a function named `total` that takes in the same two inputs and returns the amount he needs to pay. You need to use your previously-defined functions.
Sample Execution:

```
>>> total(2, 2)
24
>>> total(10, 1)
53
```

Solution:

```
def total(chickens, cows):
    if too_many(chickens, cows):
        return tax_count(chickens, cows) + 5
    return tax_count(chickens, cows)

# Alternate solution
def total(chickens, cows):
    return tax_count(chickens, cows) + 5*int(too_many(chickens, cows))
```

Question 7

Sometimes, we may wish to encrypt our password by adding some asterisks at the back of the word. We want to mask the final 4 characters with "*". If the word is shorter than 4 letters, the entire word is masked. This function will be called **`maskify`** that takes in a word and returns the new masked word.
Sample Execution:

```
>>> maskify("password")
'pass****'
>>> maskify("burger")
'bu****'
>>> maskify("cone")
'****'
>>> maskify("cs")
'**'
>>> maskify("cs1010s is fun")
'cs1010s is****'
```

Solution:

```
def maskify(word):
    if len(word) < 4:
        return "*"*len(word)
    return word[:-4] + "****"

# Alternate solution
def maskify(word):
    return word[:-4] + "*"*min(4, len(word))
```

Question 8

Last question! What does this function do?

```
def iterate(x):
    total = 0
    for i in range(x):
        if x % 2 == 1:
            total += i
    return total
```

Solution:
This function will return 0 if x is even and the sum of all positive integers less than x if x is odd.
Note that the if part is checking whether x is odd or not, not i.

Solution compiled by Russell Saerang.