Addis Coder Quiz 2

Problem

What is the **printed output** of the following code?

```
numbers = {"asir" : "ten", "hulet" : "two", "amist" : "five", "haya"
: "twenty", "arba" : "forty"}

numbers["arba"] = 40
del numbers["asir"]
numbers["sabat"] = 7

print(numbers)
```

In [ ]:
```
{"hulet": "two", "amist": "five", "haya": "twenty", "arba": 40, "sabat": 7}
```

Problem

What is the **printed output** of the following code?

```
count = 10
if count <= 10:
    for i in range(5):
        print(count)
```

In [ ]:
```
10
10
10
10
10
```

```
for i in range(5):
    if i < 3:
        print(i)
```

In [ ]:
```
0
1
2
```

```
count = 10
if count > 11:
    print(count)
for i in range(5):
    print(i)
```

In [ ]:
```
0
1
```

```
2
3
4
```

Problem

Fill in the function `filterList(lst)` below so that it **returns a new list** containing all the elements of list **smaller than 10**.

Example: `[9, 42, 12, 2, 17, 6, 6, 5] --> [9, 2, 6, 6, 5]`

In [ ]:
```python
def filterList(lst):
    # Write your code here
    new_list = []
    for n in lst:
        if n < 10:
            new_list.append(n)
    return new_list
```

Problem

Write a function `countCharacters(word)` that returns the **number of times each character appears** in the word.

Hint: Try using a dictionary

Example: `countCharacters("Addis")` --> `{"A":1, "d":2, "i":1, "s":1}`

In [ ]:
```python
def countCharacters(word):
    # Write your code here
    dic = {}
    for char in word:
        if char in dic:
            dic[char] += 1
        else:
            dic[char] = 1
    return dic
```

Problem

```python
def square_area(num):
    return num ** 2

def print_square_area(num):
    print(num ** 2)
```

What is the **value** of `hollow_square` in the code below? If there is an **error** explain why.

```python
hollow_square = square_area(10) - square_area(5)
```

In [ ]:
```
# 10**2 - 5**2 = 100 - 25 = 75
```

What is the **value** of `printed_hollow_square` in the code below? If there is an **error** explain why.

```
printed_hollow_square = print_square_area(10) - print_square_area(5)
```

In [ ]:
```
# Error, because you try subtracting None from None, because print_square_area
```

Default Parameters

What is the **printed output** of the following code?

```
def default_params_func(x = 1, y = 2):
    print(x)
    print(y)


default_params_func()
```

In [ ]:
```
1
2
```

```
default_params_func(10)
```

In [ ]:
```
10
2
```

```
default_params_func(y = 15)
```

In [ ]:
```
1
15
```

```
default_params_func(10, 15)
```

In [ ]:
```
10
15
```

Early return

The following function is supposed to reverse the input string, but it **does not work**.

What is the **output** of the following code? Hint: it is **not** `dcba`.

In [ ]:
```
def reverse_string(string):
    new_string = ''
    for letter in string:
        new_string = letter + new_string
        return new_string
```

```
print(reverse_string('abcd'))
```

Fix the code. The problem is **indentation**. Change the **indentation** of the code inside the function so that the output of `print(reverse_string('abcd'))` will be `abcd`

In [ ]:
```
# It will print "a" because the return statement is inside the for loop, so it

def reverse_string(string):
    new_string = ''
    for letter in string:
        new_string = letter + new_string
    return new_string
```

What is the image?

What image is produced by the following code? **Fill in the black pixels** below.

```
from simpleimage import SimpleImage

size = 5
image = SimpleImage.blank(size, size)
for i in range(size):
    image.set_rgb(i, i, 0, 0, 0)  # black pixel
for i in range(size-1):
    image.set_rgb(i, i+1, 0, 0, 0)  # black pixel
image.set_rgb(0, size-1, 0, 0, 0)  # black pixel

image.show()
```

Answer: Diagonal from 0,0 to 4,4 will be filled in, then diagonal from 0,1 to 3,4 will be filled in, then pixel 0,4.

Problem

Fill in the function below to draw a black rectangle with given `width` and `height` onto the input `image`. The top left corner of the rectangle should be (x, y).

In [ ]:
```
from simpleimage import SimpleImage

def draw_rectangle(image, x, y, width, height):
    # Write your code here
    for col in range(x, x + width):
        for row in range(y, y + height):
            image.set_pixel(col, row, 0, 0, 0)



image = SimpleImage.blank(8, 5)
```

```
draw_rectangle(image, 1, 2, 3, 2)
image.show()
```

The code above should give the following rectangle:

Problem

What is the **printed output** of the following code?

```
def y(x):
    result = 0
    for i in x:
        result += i
    return result

def f(x):
    result = []
    for i in x:
        result += [y(i)]
    return result

print(f([[1,2,3],[4,5,6],[7,8,9]]))
```

In [ ]:
```
[6, 15, 24]
```

Function decomposition

The following function computes the sum of digits of a number:

```
# Example: sum_of_digits(427) = 13
def sum_of_digits(n):
    total = 0
    for digit in str(n):
        total += int(digit)
    return total
```

A **Harshad number** is a number that is **divisible by its sum of digits**.

Fill in the function `is_harshad` that takes a number `n` as a parameter and returns `True` **if it is a Harshad number**, and `False` otherwise.

Hint: You can **call** `sum_of_digits` in your code.

In [ ]:
```
def is_harshad(n):
    # Write your code here
    return n % sum_of_digits(n) == 0
```

Write a function `print_harshads` that takes a number `n` and **prints all Harshad numbers** from 1 to `n`.

Hint: You can **call** `is_harshad` in your code.

In [ ]:
```
def print_harshads(n):
    for i in range(1, n+1):
        if is_harshad(i):
            print(i)
```

Accidental recursion

The following code should remove spaces from a string and print the result for the input `"Addis Coder"` and `"Don't worry be happy"`. But this code **does not work**.

```
def remove_spaces(string):
    new_string = ""
    for letter in string:
        if letter != " ":
            new_string += letter
    print(new_string)

    remove_spaces("Addis Coder")

remove_spaces("Don't worry be happy")
```

What happens when you run this code? **Why?** (Hint: Carefully look at the **indentation**.)

In [ ]:
```
# We get an infinite loop, because of remove_spaces being indented.
```

Fix the code. The problem is **indentation**. Change the **indentation** of the code inside the function so that the output of the code will be

```
AddisCoder
Don'tworrybehappy
```

In [ ]:
```
# dedent the first remove_spaces call
```

Recursion

For the following code **circle** the **base case** and the **recursive case**.

```
# TODO: Decide which option we would like to have.

# Simple option.
def simple_recursion(n):
    if n <= 0:
        return
    simple_recusion(n - 1)
```

```
# Slightly more tricky option.
def flatten_list(lst):
    if type(lst) == list:
        for  i in lst:
            flatten_list(i)
    else:
        print(lst)
```

Recursion

What is the **printed output** of the following code?

```
def func(x):
    print(x)
    if x >= 1:
        func(x-1)
    if x >= 2:
        func(x-2)

func(3)
```

In [ ]:
```
3
2
1
0
0
1
0
```

Recursion

Write a function to sum up the numbers from 1 to `n`. **Use recursion!**

NO for/while loops.

In [ ]:
```
def sumN(n):
    if n == 0:
        return 0
    else:
        return n + sumN(n-1)
```

Harder recursion

Consider the number sequence defined via $s_n = n \cdot s_{n-1} + s_{n-2}$ with $s_0 = 1$ and $s_1 = 1$.

Write a function to compute s_n. **Use recursion!**

In [ ]:
```
def mystery(n):
    if n <= 1:
```

```
        return n
    return n * mystery(n-1) + mystery(n-2)
```