**CS 161 Python Final Study Exercises Part 2**

1. Complete the following code by filling in the blanks to get  
   **Email me at omlida@wou.edu or call at 503 838 9272**as output.

email = 'Email: omlida@wou.edu'  
phone = 'Ph: 503-838-9272'  
my\_str = 'Email me at {} or call at {}'

**print(‘Email me at {} or call at {}’.format(email.split(' ')[1], ' '.join((phone.split(' ' )[1]).split('-'))))**

1. What is the output?

id\_list = [125, 674, 809, 490]  
for id in id\_list:  
    if id != 809:  
        print(id, end=' ')  
print('Done')

**125 674 490 Done**

1. What is the base 10 representation of the binary number 000100101?

**37**

1. What is the binary representation of 17?

**10001**

1. Consider a package **Animals**that contains the modules **cat.py**,**dog.py**, and **bird.py**.

Write a statement to import **dog.py**?

**from Animals import dog**

1. What is the output?

my\_string = 'The Monmouth zip code is 97361’  
print(my\_string[-5:].isdigit())  
print(my\_string[:4].isupper())  
print(my\_string[:3].islower())

**True**

**False**

**False**

1. What is the output?

num\_list = [ 9, 1, 5, 3, 4, 8, 6 ]  
for index, value in enumerate(num\_list):  
    if index == value:  
        print('\*', end='')  
    print(value, end=' ')

**9 \*1 5 \*3 \*4 8 \*6**

1. Which of the standard libraries is used to access a file stored on a website? (Hint: Look in the modules)

**urllib**

1. Write a statement that pauses code execution for 7 seconds? (Hint: use the time module)

**time.sleep(7)**

1. Write a for loop to output the first 10 even numbers.

**for i in range(0:11:2)**

**print(i)**

1. Write a while loop to output odd numbers up to and including 13.

**i = 1**

**while i <= 13:**

**print(i)**

**i += 2**

1. Using the following code, what is the output or expression?

my\_string =”Computer Science Rocks!"

my\_list = my\_string.split()

|  |  |
| --- | --- |
| **Instruction** | **Output/Expression** |
| print(my\_string) | **Computer Science Rocks** |
| print(my\_list) | **[‘Computer’,’Science’,’Rocks’]** |
| Using **my\_string**, write an expression using slicing to print Rocks! | **print(my\_string[-6:]** |
| Using **my\_list**, write an expression to print Rocks! | **print(my\_list[2])** |

1. Create a module named **Square.py**. In the module, include the following functions:
   1. An **area** function that when given the length of a side, calculates and returns the area of a square.
   2. A **perimeter** function that when given the length of a side, calculates and returns the perimeter of a square.
   3. A **print dimensions** function that when given the length of a side, prints out the details of the square.

For example, if given the side’s length of 7, the output would look similar:



**def area(length):**

**return length\*length**

**def perimeter(length):**

**return length\*4**

**def print\_dimensions(length):**

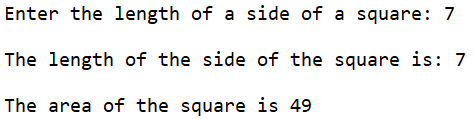
**print('The length of the side of the square is: ', length)**

1. Create a file named **Shapes.py**. In the file, do the following:
   1. Ask the user to enter the length of the side of a square.

Enter the length of a side of a square:

* 1. Using the side length, call the print dimensions function that you wrote in the above module.
  2. Using the side length, call the area function you wrote in the above module.
  3. You will need in import **only** what you plan to use in this file.

Sample output when you run Shapes.py:



**from Square import area, perimeter, print\_dimensions**

**length = int(input('Enter the length of a side of a square: '))**

**print\_dimensions(length)**

**print('The are of the square is', area(length))**

1. Practice with Dictionaries. What is the output of Lines 1-4?

def foo(my\_dict,k,v):

if k in my\_dict:

my\_dict[k] = v

a\_dict = {'buffalo':10, 'raccoon':20, 'fish':100, 'turtle':200}

print('Line 1 ',a\_dict) **# Line 1 {'buffalo':10, 'raccoon':20, 'fish':100, 'turtle':200}**

temp = 0

for value in a\_dict.values():

temp = temp + value

print('Line 2 ',temp) **# Line 2 330**

temp = ''

for key in a\_dict:

if temp < key:

temp = key

print('Line 3 ',temp) **# Line 3 turtle**

foo(a\_dict, 'buffalo', 300)

print('Line 4 ',a\_dict) **# Line 4 {'buffalo':300, 'raccoon':20, 'fish':100, 'turtle':200}**