CSE 160 Section 5 Problems

1. After the following lines of code are executed, what values are stored in the set `output_set`?

```
input_list = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5, 9]
output_set = set()
for i in input_list:
    output_set.add(i)
```

2. In one line of code, print the set of all letters that are in both the sets. (i.e. - their intersection)

```
set_one = {'a', 'b', 'c', 'd', 'e', 'f'}
set_two = {'a', 'c', 'd','g'}
```

3. Write a function called all_unique_words that takes in a string file_name and returns the number of unique words in the file.

You may use the split() function for this problem, which takes in a string and returns a list of the words in the string.

Example:
colors.txt -> “red green blue green”
print(all_unique_words(“colors.txt”)) -> 3

```
def all_unique_words(file_name):
```

4. Write a function called report_long_lines that takes a string file_name and an integer max_length as arguments. It should return the number of lines that were longer than the given length.

You may assume the file name provided describes a file that exists.

Example:
numbers.txt:
one
two
three
four
five

print(report_long_lines(“numbers.txt”, 3)) -> 3

```
def report_long_lines(file_name, max_length):
```

5. Write a function called write_prime_numbers that takes in a string file_name and an integer end_number as arguments. The function should write all the prime numbers between 1 and end_number (exclusive), with each prime number on a separate line.

Example:
write_prime_numbers(“example.txt”, 10)

example.txt:
2
3
5
7

```
def write_prime_numbers(file_name, end_number):
```

(Optional) Review of Concepts from Before Midterm

1. Write a function that reverses a list, without using the built-in reverse function. Your function should return the reversed list, and not modify the list passed as a parameter. For example: reverse_list([1, 2, 3]) returns [3, 2, 1].

2. Consider the following Python program:

```
def pos_dif(y, x):
    """ Returns the positive difference of two numbers. """
    # Location B
    return abs(x - y)
def percent_error(actual, expected):
    """Returns the percent error of an experimental result. """
    # Location A
    x = pos_dif(actual, expected)
    y = expected # Location C return x / y
    a = 15.0
    b = 10.0 print(percent_error(a, b))
```

For each of the locations indicated above, draw the environment frame(s) at that moment during execution.