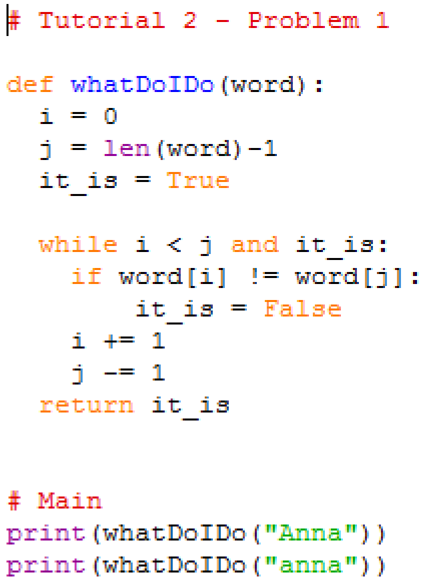
Tutorial 2

Problem 1.

Chapter 9, pages 86 and 87 from our online textbook

What does the following Python program do? Hand trace it in order to answer the question.



```
# Tutorial 2 - Problem 1

def whatDoIDo(word):
  i = 0
  j = len(word)-1
  it_is = True

  while i < j and it_is:
    if word[i] != word[j]:
        it_is = False
    i += 1
    j -= 1
  return it_is


# Main
print(whatDoIDo("Anna"))
print(whatDoIDo("anna"))
```

Problem 2.

Modify the Python program found in Problem 1 above such that it reads the content of the file **some_words.txt** and calls the function `whatDoIDo(<argument>)` passing each word read from this file as an argument.

The content of the file **some_words.txt** is as follows:

```
armour
civic
kayak
mother
politics
racecar
radiator
rotator
zoo
```

For each of the above word (test data), what is the expected result?

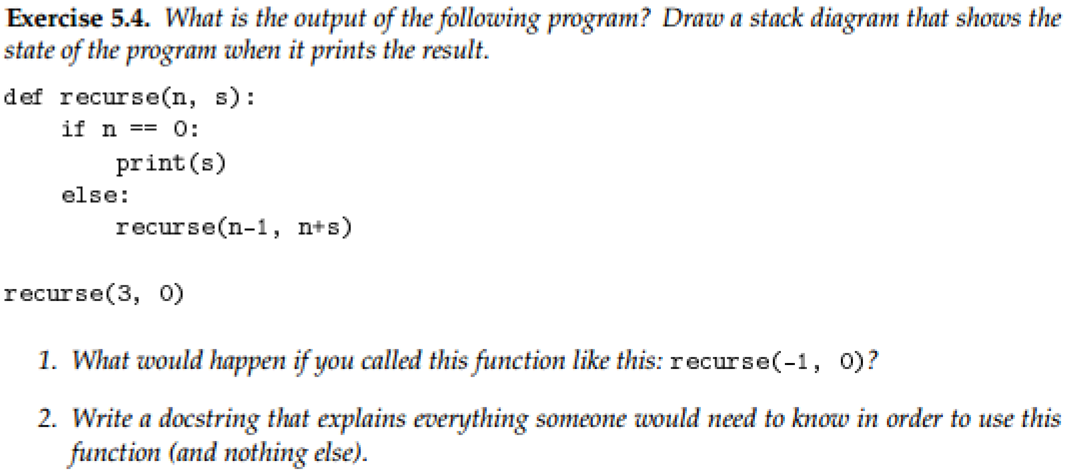
Problem 3.

Question inspired from Chapter 6, page 54 of our online textbook

Describe how we would incrementally develop a function called **hypotenuse** that returns the length of the hypotenuse of a right triangle given the lengths of the other two legs as arguments. Write the test case(s) we would use in order to test each stage of the development process (i.e., each version of our function as we incrementally develop it).

Problem 4.

From our online textbook



Exercise 5.4. *What is the output of the following program? Draw a stack diagram that shows the state of the program when it prints the result.*

```
def recurse(n, s):
    if n == 0:
        print(s)
    else:
        recurse(n-1, n+s)

recurse(3, 0)
```

1. *What would happen if you called this function like this:* `recurse(-1, 0)`*?*
2. *Write a docstring that explains everything someone would need to know in order to use this function (and nothing else).*

A stack diagram is the diagram we produce when we box trace the execution of a function.

Problem 5.

Counting exercise: in a table, write all the binary numbers corresponding to the decimal numbers 0_{10} to 32_{10}.

Problem 6.

Consider the following binary numbers. Convert each of them to its equivalent decimal number. Let us show our work, i.e., show how we get to our answer.

a. 10010011_2

b. 00001100_2

c. 11110011_2

d. 01010010_2

e. 01110110_2

Problem 7.

Consider the following decimal numbers. Convert each of them to its equivalent binary number. Let us show our work, i.e., show how we get to our answer. Pad your answer to nearest multiple of 8-bit binary number.

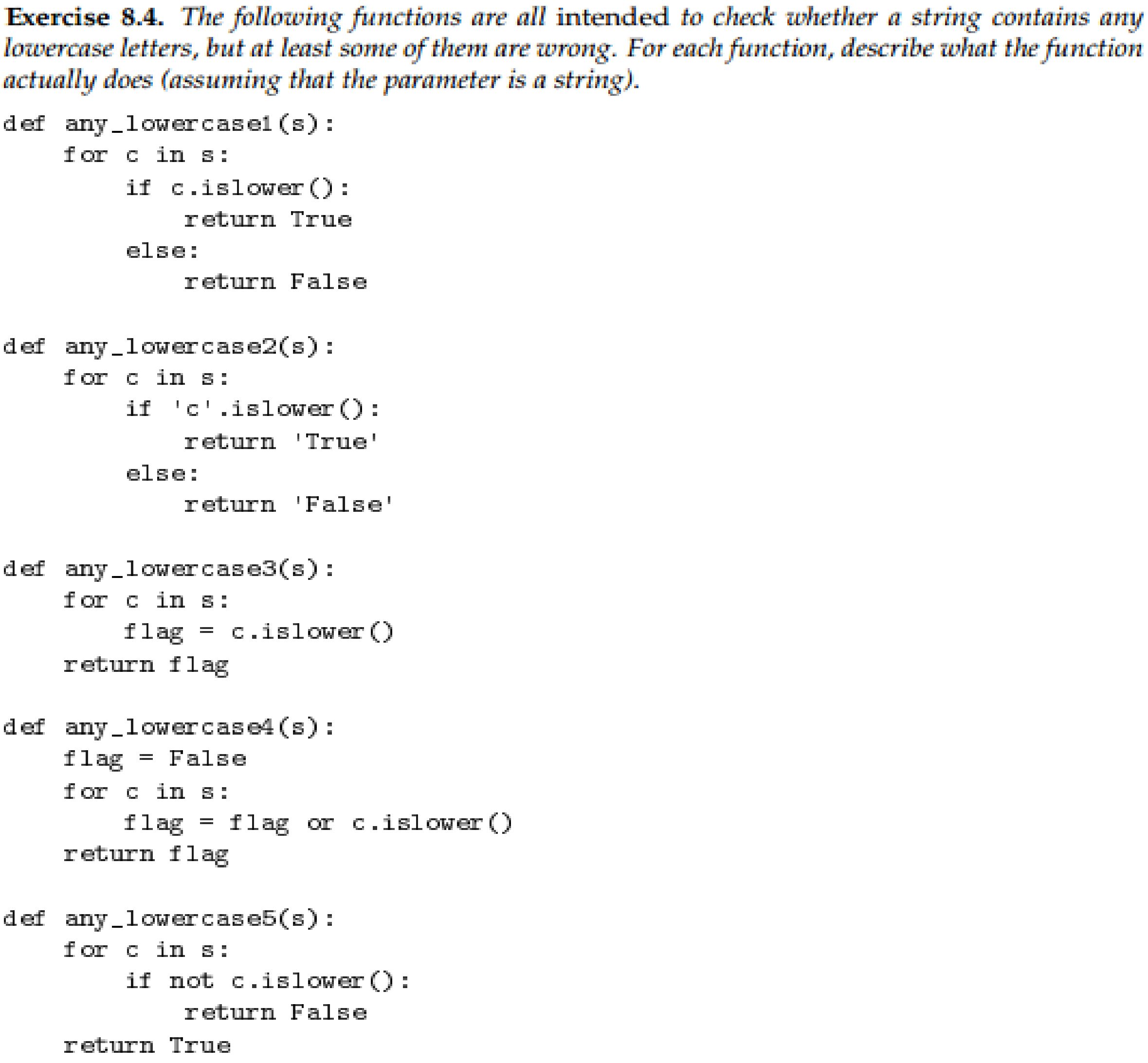
a. 28_{10}

b. 498_{10}

c. 2498_{10}

Problem 8.

From our online textbook



Exercise 8.4. *The following functions are all* intended *to check whether a string contains any lowercase letters, but at least some of them are wrong. For each function, describe what the function actually does (assuming that the parameter is a string).*

```
def any_lowercase1(s):
    for c in s:
        if c.islower():
            return True
        else:
            return False

def any_lowercase2(s):
    for c in s:
        if 'c'.islower():
            return 'True'
        else:
            return 'False'

def any_lowercase3(s):
    for c in s:
        flag = c.islower()
    return flag

def any_lowercase4(s):
    flag = False
    for c in s:
        flag = flag or c.islower()
    return flag

def any_lowercase5(s):
    for c in s:
        if not c.islower():
            return False
    return True
```