MTH 3300
E. Fink

3300 Problems, Section 3: Introduction to Functions, Lists and `for` Loops

1. Write a Python function which takes two `float`s x and y as input, and returns $\sqrt{x^2 + y^2}$ as output.
2. Write a Python function representing $f(x) = |x|$, without using `abs` or exponents.
3. Write a Python function called `sum_length`, which takes two `str`s as input, and returns the sum of their lengths.
4. Write a Python function called `same_length`, which takes two `str`s as input, and returns whether or not the two `str`s have the same length. (So, what are the possible values it can return?)
5. Two planes are a dangerous distance from one another if the difference between their heights is less than 2000 feet. Given that, write a function `too_close` that makes the following code work:

```
height1 = float(input("Enter height of first plane"))
height2 = float(input("Enter height of second plane"))
if too_close(height1, height2):
   print("Too close!")
else
   print("Ok")
```

6. You have a list of 25 `int`s named `number_list`. Write code that will assign the last 15 of those numbers to a new list called `short_list`.
7. I have a list defined by the following line of code:

```
x = ["a", "b", "c", "d", "e"]
```

   Write code that changes the value of `x` to be `["a", "b", "u", "e", "f", "g", "h"]`, *without using* `=` *anywhere.* (Instead, feel free to use `.append()`, `.insert()`, and `del`.)
8. What will the following code print out?

```
a = ["p", "q", "r", "s"]
b = a
c = a + ["t"]
a.append("u")
print(len(a), len(b), len(c))
```

9. Write code that allows the user to enter a long sentence, and then prints out the third word of that sentence. Assume that the sentence is written with the usual spacing, no punctuation, and at least three words. For example, if the sentence entered is

```
The quick brown fox jumped
```

   then the program should print `brown`.
10. You have a list of 25 `int`s named `number_list`. Using a `for` loop, write code that will display the sum of these numbers.
11. Suppose that `abc` is a list of `float`s that already has been created.
    a. Write code that prints the product of the entries.
    b. Write code that prints the average of the entries.
12. Suppose that I have a list of `int`s named `x`, defined by

```
x = [14, 18, 31, 16, 23, 28, 10, 17]
```

    Write code that prints out the LARGEST EVEN number in `x` (in this case, it would be `28`). Your code should still work if `x` were replaced with *any list of positive integers that contains at least one even number.*
13. Suppose that I have a list of `int`s named `y`, defined by

```
y = [3, 4, 7, 2, 1]
```

Write code that prints out the SUM of all the ODD numbers in `y` (in this case, it would be `11`. Your code should still work if `y` were replaced with *any list of positive integers*.

14. Suppose that I have a list of `str`s named `words`, defined by

```
words = ["apple", "banana", "pear"]
```

Write code that will print `p-word` if there is at least string that begins with the character `p` in the list, and prints `No p-word` otherwise. Your code should still work if `words` were replaced with *any list of strings*.

15. What prints out from each of the two code snippets?

```
x = ["a", "b", "a", "b", "a"]
count = 1
for i in x:
    if i == "b":
        print(count)
        count = count + 1
print(i)
####################
x = ["a", "b", "a", "b", "a"]
count = 1
for i in x:
    if i == "b":
        print(count)
        count = count + 1
    print(i)
```

16. When the following code fragment is finished executing, what values are held in the array `my_list`?

```
my_list = [1, 5, 3, 2]
for i in range(4):
    my_list[i] += 1
```

17. When the following code fragment is finished executing, what values are held in the array `my_list`?

```
my_list = [1, 5, 3, 2]
for i in range(1, 4):
    my_list[i] += my_list[i-1]
```

18. Write the code necessary to create a list that contains 100 numbers, all entered by the user.
19. Write code that creates a list of length 100, with each entry 0.
20. Write the code necessary to make a list of 40 entries named `surveys`, with each entry being a computer-generated random integer between 1 and 10.
21. Write the code necessary to make a list of 50 entries named `examScores`, with each entry being a computer-generated random `int` between 70 and 100.
22. Write code that will print out the result of the sum $1^2 + 2^2 + 3^2 + 4^2 + \ldots + 100^2$, without using the `sum()` function.
23. Write code that will print out the result of the sum $1 + 4 + 7 + 10 + \ldots + 97 + 100$, without using the `sum()` function.
24. Create a loop which will output the first 50 EVEN integers, each on a different line.
25. Write a loop that prints out all multiples of 173 less than 10000.
26. Suppose that the user has *already* entered a positive integer `n`. Write the code necessary to print out the product of all the EVEN integers LESS than `n`. (For example, if the user had entered `7`, the program should print out `48`, because $2 \times 4 \times 6 = 48$.)
27. Suppose that the user has *already* entered a positive integer `n`. Write the code necessary to print out the sum of all the ODD integers LESS than `n`. (For example, if the user had entered `8`, the program should print out `16`, because $1 + 3 + 5 + 7 = 16$.)
28. Write code necessary to allow the user to input 100 names, and then print them out in reverse order.
29. Write code which takes every value in a list of `float`s named `num_list`, and doubles every value.

30. Write code that will print out the numbers between 1 and 100, except that: instead of the number, it will print `Bop` if the number is a multiple of 2, `Slap` if the number is a multiple of 3, and `BopSlap` if the number is both. So, the printout should start like:

    ```
    1

    Bop

    Slap

    Bop

    5

    BopSlap

    7
    ```

31. Write the code necessary to do the following. Print out the first 1000 numbers, one on each line, EXCEPT: instead of printing the number, print "Fizz" if the number is a multiple of 3, "Buzz" if the number is a multiple of 5, and "FizzBuzz" if the number is a multiple of both 3 and 5.

32. I write code to check whether the following array is sorted:

    ```
    my_list = [4, 6, 19, 2, 7]
    ```

    I use the following code, expecting `Not sorted` to appear on the console.

    ```
    for i in range(4):
       if my_list[i] < my_list[i+1]:
          print("Sorted")
       else:
          print("Not sorted")
    ```

    It doesn't work like I planned it. What prints out *instead*? How could I fix this code?

33. Suppose that I have a list defined as follows:

    ```
    myList = [1, 8, 3, 7, 2]
    ```

    I *try* to write code to search the list as follows:

    ```
    searchEntry = int(input("Search for: "))
    found = False
    for elt in myList:
        if searchEntry == elt:
            found = True
            print("Found")
            break
        else:
            found = False
            print("Not Found")
    ```

    To my chagrin, when the user enters `3` for `searchEntry`, this does NOT work the way that I intend it to. What DOES print out? How could I fix this code?

34. Suppose that I have an array defined as follows:

    ```
    myList = [1, 8, 3, 7, 2]
    ```

    I *try* to find the largest element by using the following code:

    ```
    largest = myList[0]
    for i in range(1,5):
       if myList[i] > myList[i-1]:
          largest = list[i]

    print(largest)
    ```

To my chagrin, this does NOT work. What prints out INSTEAD? How could I fix this code?

35. Suppose that a list containing `int`s named `vec` is currently defined. Write code to read the first element of `vec`, erase it from the beginning, and put it on the end. For example, if `vec` contains `[1,10,24,80]` in the beginning, it should contain `[10,24,80,1]` after.

36. Suppose that a list containing `str`s named `words` currently has several entries, including *exactly one* that is the `str` `"xo"`. Write code that will remove the entry `"xo"` from the list. (Hint: you may want to stop looking after you find the `"xo"`.)

37. Suppose that a list containing `int`s named `aaa` is currently **sorted**, and an extra `int` variable named `y` has already been defined. Write the code necessary to put the value of `y` into `aaa` in the right position so that it remains sorted. In other words, if `aaa` currently contains `[1, 10, 24, 80]`, and `y` contains the value `35`, then after this code `aaa` will contain `[1, 10, 24, 35, 80]`. (Hint: think about when you know you have found the right location, and you may want to use a `break`.)

38. Write a program that simulates a 2D drunkard's walk with 100 steps. That is, write a program that starts at $(0, 0)$, and at each step of the program, changes that position by either $+1$ in the x-direction, -1 in the x-direction, $+1$ in the y-direction, or -1 in the y-direction, each possibility happening with probability $1/4$.

    For example, the first few steps might be $(0, 0) \to (0, 1) \to (-1, 1) \to (-2, 1) \to (-2, 0) \to (-1, 0) \to (-2, 0)$

    Have the program print out the location (that is, the x-coordinate and y-coordinate) after 100 steps.

39. Write a program that asks for a word (all lowercase, with no spaces – assume that the user complies), and then counts the number of *character doubles* – that is, two of the same character in a row. For example, if the user enters `bookkeeper`, the program would output 3, because of "oo", "kk", and "ee". If the user enter `butterball`, the program would output 2, because of "tt" and "ll". You may assume that the user will never enter a word with three or more of the same characters in a row.

40. Write a program that first asks the user to enter a "special" letter. The program should then ask the user to enter a single line sentence containing all lowercase letters and no punctuation, except for a period at the end. The program will then output the number of times that this letter appears in the sentence.

    Example:

    ```
    Enter a special letter:  t

    Enter a sentence:  here is my little sentence.

    Your letter appears 3 times.
    ```

    (`t` and `here is my little sentence.` are user inputs, the rest is output by the program.)

41. Write a program that first asks the user to enter a single line sentence containing all lowercase letters and no punctuation, except for a period at the end. The program will then output the number of letters in the sentence, not including spaces or the period.

    Example:

    ```
    Enter a sentence:  hey there.

    Your sentence has 8 letters.
    ```

    (`hey there.` is user inputs, the rest is output by the program.) You may assume that all spaces are the space character (as opposed to, e.g., newlines or tabs).

42. Suppose `x` is a list that has been created in a program. Write code that will take the list and reverse its contents, without using the built-in `reverse` function. So, for example, if `x` starts out as `[1, 3, 5, 10]`, it should hold the values `[10, 5, 3, 1]` afterwards.

43. Suppose `x` is a list that has been created in a program. Write code that will take the list and shift the contents of each entry one place to the right, except for the contents of the last entry, which should be moved into the first entry. So, for example, if `x` starts out as `[1, 3, 5, 10]`, it should hold the values `[10, 1, 3, 5]` afterwards.