NYU, Tandon School of Engineering
CS-UY 1114 Introduction to Programming and Problem Solving — Fall 2017

Homework #6
Due by Friday 11/3, 11:55pm

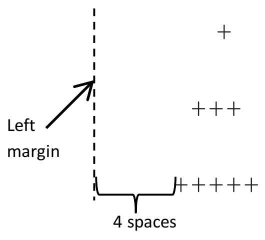
Submission instructions:

1. You should submit your homework in the NYU Classes system.
2. **Create one '.py' file per question. Each '.py' file may contain multiple functions.**
3. For this assignment, you should turn in 3 '.py' files. Name your files 'YourNetID_hw6_q1.py', 'YourNetID_hw6_q2.py' and 'YourNetID_hw6_q3.py'.

Question 1: Pine Tree

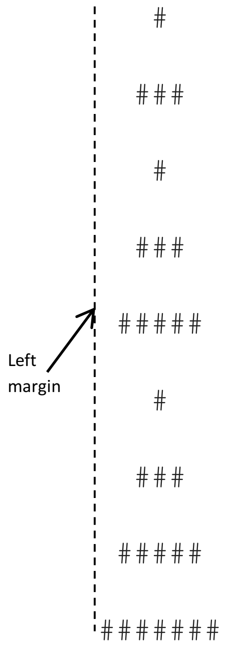
Write a program that, prints a ‘pine tree’ consisting of triangles of increasing sizes, filled with a character (‘*’ or ’+’ or ‘$’ etc). Your program should consist of three functions:

1. A function `print_shifted_triangle(n, m, symbol)`. It prints an `n`-line triangle, filled with `symbol` characters, shifted `m` spaces from the left margin.
   For example, if we call `print_shifted_triangle(3, 4, `+`)`, the expected output is:



```
      +
     +++
    +++++
```

2. A function `print_pine_tree(n, symbol)`. It prints a sequence of `n` triangles of increasing sizes (the smallest triangle is a 2-line triangle), which form the shape of a pine tree. The triangles are filled with the `symbol` character.
   For example, if we call `print_pine_tree(3, `#`)`, the expected output is:



```
   #
  ###
   #
  ###
 #####
   #
  ###
 #####
#######
```

3. A `main()` function that interacts with the user to read the number of triangles in the tree and the character filling the tree.

Question 2: Calendar

a. Implement a function:
`print_month_calender(num_of_days, starting_day)`
This function is given two parameters:
- `num_of_days` - The number of days in the month
- `starting_day` – a number 1-7 that represents the day in the week of the first day in that month (1 for Monday, 2 for Tuesday, 3 for Wednesday, etc.).

The function should:
- Print a formatted monthly calendar of that month
- Return a number 1-7 that represents the day in the week of the **last day** in that month.

Formatting Notes:
- The output should include a header line with the days' names.
- Columns should be spaced by a Tab.

Example: when calling `print_month_calender(31, 4)` it should return 6, and should print:

```
Mon	Tue	Wed	Thr	Fri	Sat	Sun
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31
```

b. Textbook P. 279, Q12.
A method for determining if a year is a leap year in the Gregorian calendar system is to check if it is divisible by 4 but not by 100, unless it is also divisible by 400.
For example, 1896, 1904, and 2000 were leap years but 1900 was not.
Write a function that takes in a year as input and return true if the year is a leap year, return false otherwise.
Note: background on leap year https://en.wikipedia.org/wiki/Leap_year

c. Implement a function:
`print_year_calender(year, starting_day)`
This function is given two parameters:
- `year` – an integer that represents a year (e.g. 2016)
- `starting_day` – a number 1-7 that represents the day in the week of 1/1 in that year (1 for Monday, 2 for Tuesday, 3 for Wednesday, etc.).

The function should use the functions from sections (a) and (b) in order to print a formatted yearly calendar of that year.

Formatting Note: As the header for each month you should print the months' name followed by the year (e.g. March 2016).

Example: Appendix A shows the expected output of the call `print_year_calender(2016, 5).`

d. Write a driver program, ie. `main()` function, that interacts with the user and your function in (c).

<u>Question 3</u>: <u>Reversing words in a phrase</u>

a. Write a function that is given a phrase and returns the first word in that phrase.
   For example, given ‘the quick brown fox’, your function should return ‘the’.

b. Write a function that is given a phrase and returns the phrase we get if we take out the first word from the input phrase.
   For example, given ‘the quick brown fox’, your function should return ‘quick brown fox’

c. Use the functions from a and b to implement a function that given a phrase, it reverses the words in that phrase (and returns it)
   For example, given ‘the quick brown fox jumps over a lazy dog”, your function should return ‘dog lazy a over jumps fox brown quick the’.

d. Write a driver program, ie. `main()` function, that interacts with the user and uses function you wrote for c.

<u>Note</u>: For all sections, assume:
1. The phrase is a sequence of words separated with a single space.
2. All letters in the phrase are lowercase.
3. The phrase does not contains any special symbols or punctuation marks

Appendix A.

The expected output of the call `print_year_calender(2016, 5)` is:

```
January 2016
Mon   Tue   Wed   Thr   Fri   Sat   Sun
                        1     2     3
4     5     6     7     8     9     10
11    12    13    14    15    16    17
18    19    20    21    22    23    24
25    26    27    28    29    30    31

February 2016
Mon   Tue   Wed   Thr   Fri   Sat   Sun
1     2     3     4     5     6     7
8     9     10    11    12    13    14
15    16    17    18    19    20    21
22    23    24    25    26    27    28
29

March 2016
Mon   Tue   Wed   Thr   Fri   Sat   Sun
      1     2     3     4     5     6
7     8     9     10    11    12    13
14    15    16    17    18    19    20
21    22    23    24    25    26    27
28    29    30    31

April 2016
Mon   Tue   Wed   Thr   Fri   Sat   Sun
                        1     2     3
4     5     6     7     8     9     10
11    12    13    14    15    16    17
18    19    20    21    22    23    24
25    26    27    28    29    30

May 2016
Mon   Tue   Wed   Thr   Fri   Sat   Sun
                                    1
2     3     4     5     6     7     8
9     10    11    12    13    14    15
16    17    18    19    20    21    22
23    24    25    26    27    28    29
30    31

June 2016
Mon   Tue   Wed   Thr   Fri   Sat   Sun
            1     2     3     4     5
6     7     8     9     10    11    12
13    14    15    16    17    18    19
20    21    22    23    24    25    26
27    28    29    30
July 2016
```

Mon	Tue	Wed	Thr	Fri	Sat	Sun
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

August 2016

Mon	Tue	Wed	Thr	Fri	Sat	Sun
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

September 2016

Mon	Tue	Wed	Thr	Fri	Sat	Sun
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

October 2016

Mon	Tue	Wed	Thr	Fri	Sat	Sun
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

November 2016

Mon	Tue	Wed	Thr	Fri	Sat	Sun
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

December 2016

Mon	Tue	Wed	Thr	Fri	Sat	Sun
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	