**Week 13 In-Class Exercises (Extra)**

**Q1: HTML [ \*\*\* ]**

Webpages are in HTML format. In an HTML file, texts surrounded by tag pairs <b> and </b> will be shown in bold font on a Web browser.

For example, given the following text in HTML format:

|  |
| --- |
| This is an <b>example</b> of some <b>text</b> in HTML. <b>Some of the segments</b> will be <b>shown in bold</b>. |

The text segments 'example', 'text', 'Some of the segments' and 'shown in bold' will be bold.

Define a function called extract\_bold\_texts(). The function takes in a single string called html\_text. It returns a list of strings that are segments inside html\_text which are surrounded by <b> and </b>.

You can assume that html\_text is a well formatted HTML text without any unpaired tags.

E.g.,

* extract\_bold\_texts('<b>ABC</b> <i>abc</i> <a>This is a link</a> <b>def 123 </b><b></b>0000') returns ['ABC', 'def 123 ', '']. (Note that the last string is an empty string.)
* extract\_bold\_texts('A piece of text without tags.') returns [].

Hint: You can use html\_text.find('<b>') to find the index of the first <b>. Then use html\_text.find('</b>') to find the index of the corresponding </b>. With the indices, you can extract the first segment. You can then use string slicing to change html\_text into the remaining string after discarding everything until the first </b>. Then you can repeat the process above until no more <b> can be found.

## Q2: To-Do List [ \*\*\* ]

**Part (A)**

Write a program that allows a user to enter his/her tasks. The program asks for a task as well as the due date (which contains a month and a date). If the user enters an invalid month or invalid date, the program has to re-prompt the user until the information is valid. (A valid month is an integer between 1 and 12. A valid date is an integer between 1 and the maximum number of days of that month. E.g., if month is 10, then the date has to be between 1 and 31. If month is 4, then the date has to be between 1 and 30. For February, the date has to be between 1 and 29.) The program keeps prompting the user for more tasks and their due dates until the user chooses to stop. The program then displays the list of all tasks, grouped by their due dates.

You can assume that all the tasks are due in the same year. Therefore, for all due dates, the year is omitted.

A sample run of the program is shown below. Texts shown in bold are user inputs.

|  |
| --- |
| Do you want to enter a task? [Y|N]:**Y**  Enter task description:**IS111 Lab2** Enter the month it is due:**13** Sorry, invalid month!  Enter the month it is due:**abc** Sorry, invalid month!  Enter the month it is due:**11**  Enter the date it is due:**31** Sorry, invalid date!  Enter the date it is due:**$$$** Sorry, invalid date!  Enter the date it is due:**3**  IS111 Lab 2 due on 3/11 has been recorded.  Do you want to continue? [Y|N]:**Y**  Enter task description:**Top up EZ-link card** Enter the month it is due:**10** Enter the date it is due:**10**  Top up EZ-link card due on 10/10 has been recorded.  Do you want to continue? [Y|N]:**Y**  Enter task description:**ECON101 HW#3** Enter the month it is due:**11** Enter the date it is due:**3**  ECON101 HW#3 due on 3/11 has been recorded.  Do you want to continue? [Y|N]:**N**  Here are the tasks you've entered:  Due on 3/11:  IS111 Lab 2  ECON101 HW#3  Due on 10/10:  Top up EZ-link card |

**Part (B)**

Improve the program above such that when listing all the tasks entered, they’re ordered by the due dates in chronological order, i.e., the tasks that are due earlier are listed earlier.

For example, given the tasks shown in the sample run above, the program should display the tasks as follows in the end:

|  |
| --- |
| Here are the tasks you've entered:  Due on 10/10:  Top up EZ-link card  Due on 3/11:  IS111 Lab 2  ECON101 HW#3 |

Hint: A brute force solution is to enumerate all the dates of a year and check if there are any tasks due on each date. I.e., you can go through the sequence of dates 1/1, 2/1, 3/1, …, 1/2, 2/2, 3/2, …, 1/12, 2/12, …, 31/12, and for each date, if there are tasks due on that date, display the tasks.

**Part (C)** (Out of Scope)

Further improve the program such that if a due date is in the past, it is not accepted. For example, assuming that today’s date is November 10. The code below shows the expected error messages:

|  |
| --- |
| ...  Enter task description:**IS111 Lab2** Enter the month it is due:**9** Sorry, the month has passed.  Enter the month it is due:**11** Enter the date it is due:**9** Sorry, the date has passed.  Enter the date it is due:**12**  ... |

To get today’s date, you can use the following code:

|  |
| --- |
| import datetime  date\_of\_today = str(datetime.date.today()) |

The code above gives you the date in 'YYYY-MM-DD' format.