

National University of Singapore
School of Computing
CS1010S: Programming Methodology
Semester I, 2018/2019

Sidequest 8.2
Cheryl's Birthday

Release date: 24 September 2018

Due: 09 October 2018, 23:59

Required Files

- sidequest08.2-template.py

Background

Albert and Bernard just became friends with Cheryl, and they want to know when her birthday is. Cheryl gives them a list of 10 possible dates. Cheryl then tells Albert and Bernard separately the month and the day of her birthday respectively.

May 15 May 16 ~~May 19~~
June 17 ~~June 18~~
July 14 July 16
Aug 14 Aug 15 Aug 17

Passing by, you overhear the conversation between Albert and Bernard.

Albert I don't know Cheryl's birthday, but I know that Bernard doesn't know too.

Bernard At first I didn't know when Cheryl's birthday is, but I know now.

Albert Then I also know when Cheryl's birthday is.

Using the information given by Albert and Bernard as constraints, you realize that you can create a program that can filter out the invalid cases, and identify Cheryl's birthday.

You begin by arranging all possible birthdays into a neat table, and you can observe that some dates may contain a unique day or a unique month, or both, among all other possible dates. If Albert is given such a month where only one of the possible birthdays is in that month, he will know Cheryl's birthday immediately. Similarly, if Bernard is given such a unique day, he will also know Cheryl's birthday immediately.

	14	15	16	17	18	19
May		×	×			×
June				×	×	
July	×		×			
Aug	×	×		×		

Administrivia

For this sidequest, birthday refers to a tuple containing two strings: the month, followed by the day. The term month refers to the month of the birthday while day refers to the day of the birthday.

For example, for the birthday ("May", "15"), month is represented by the string "May" and date is represented by the string "15".

All the possible birthdays are stored in a bigger tuple, named possible_birthdays. The possible_birthdays provided by Cheryl in the beginning has been given in the template file.

This mission consists of **three** tasks.

Task 1: Unique dates and months (3 Marks)

You would like to find out if a given day or month appears only once among all possibilities of birthdays.

- The function unique_day takes in a string day and a tuple of possible_birthdays and returns True if the day is unique for the particular set of possible birthdays. Otherwise, it returns False. Implement unique_day. (1 Mark)
- The function unique_month takes in a string month and a tuple of possible_birthdays and returns True if the month is unique for the particular set of possible birthdays. Otherwise, it returns False. Implement unique_month. (1 Mark)
- The function contains_unique_day takes in a string month and a tuple of possible_birthdays and returns True if there is a unique day in the month for the particular set of possible birthdays. Otherwise, it returns False.

Implement contains_unique_day. (1 Mark)

For example if the possible_birthdays are (("May", "16"), ("May", "17"), ("June", "16")). Then,

- 17 is a **unique day** because 17 appears only once across all the months.
- June is a **unique month** because it only has one day whereas May has two.
- May contains a **unique day** ("May", "17").
- June does not contain a unique because in the set of possible_birthdays, the month June only has day 16 which is **not a unique day**.

Task 2: Setting up the constraints (6 marks)

- I don't know Cheryl's birthday, but I know that Bernard doesn't know too.*
Analyze the first statement from the conversation, and you will realize that Albert is essentially saying that his given month is **not** unique. Also, for his given month, all the days are **not unique**, so Bernard cannot know the birthday either.

The function `statement1` takes in a tuple `birthday` and a set of `possible_birthdays`. Then, it will return `True` if the month of the `birthday` is not unique, and that particular month does not contain a unique day either in the given set of possible birthdays. Otherwise it will return `False`. Implement `statement1`. (2 marks)

- (b) *At first I didn't know when Cheryl's birthday was, but I know now.*

Analyze the second statement from the conversation, you will realize that the day that Cheryl told Bernard is a unique day from the remaining possible birthdays. The function `statement2` takes in a tuple `birthday` and a set of `possible_birthdays`, and will return `True` if the day of the given `birthday` is unique in the set of `possible_birthdays`. Otherwise, it returns `False`. Implement `statement2`. (2 marks)

- (c) *Then I also know when Cheryl's birthday is*

Analyze the third statement from the conversation, and you will realize that the month Albert has heard is a unique month from the remaining possible birthdays. The function `statement3` takes in a tuple `birthday` and a set of `possible_birthdays`. It will return `True` if the month of the `birthday` is unique for the particular set of possible birthdays. Otherwise it will return `False`. Implement `statement3`. (2 marks)

Task 3: And now I know her birthday, too! (1 mark)

Implement the function `get_birthday` that will take in a tuple, `possible_birthdays` and will return a tuple containing the birthdays that are still valid after being filtered through the constraints. If done correctly, the results should be a single element tuple consisting of Cheryl's birthday only.

(Hint: Make use of `filter` to shortlist a subset of the previous list of `possible_birthdays` using one constraint each time.)