# T7: Mad Libs

* Paired assignment T7 should be completed with a partner.
* To begin, go to “File” and Select “Make a Copy...”
* Move your copy of the document to the folder for your section.
* Rename the file to **T7: Mad Libs - usernames** (replace usernames with your usernames). To do this, click the label in the top left corner of your browser.

## Objectives

* Practice more breaking a larger problem down into smaller "pieces" using functions
* Gain practice using strings

## Palindromes

A **palindrome** is a word, line, verse, number, or a sentence that reads the same backward as forward when you ignore punctuation and spacing. Examples include "*Madam, I'm Adam*" or in "*Poor Dan is in a droop*". Some people love making palindromes, and [Palindrome List](http://www.palindromelist.net/) presents more examples than you might imagine of what people can do as a hobby.

The [t7-palindrome.py](https://drive.google.com/open?id=0B0J8Yj0B6KRSZXJxbHl3bkliQ3c) code is an example of a program in Python that uses several of the features of the string class to check for palindromes. It might be useful as a starting point for this assignment.

## Mail Merge

In this teamwork we will write a program to generate Mad Libs using a process called mail-merging. This process could actually be useful when you are sending out your resume to find a job!

## Mad Libs

A Mad Lib is a game in which words are substituted for blanks in a story. Each blank is specified by a generic category like "noun", "verb", "adjective", "place", etc. The words are chosen by the category and substituted into blanks in the story. This process often produces funny results. You can try some Mad Libs at <http://www.madglibs.com/>.

Your primary challenge in this assignment is a design problem. Before coding, you must figure out how to design a program which will generate a Mad Lib story when run by the user. As you may have learned from A4, without a solid plan, coding can become unruly and messy.

Create a program to accomplish the following tasks:

1. Create a story using a triple-quoted string, which will allow it to span multiple lines.
2. This story string will contain the unchanging parts of the story (the **template**) as well as the blanks that will be replaced by words from categories input by the user.
3. Your story string may say whatever you want it to say, but it must have at least 5 merge-fields which will be delineated by place holders ({1}, {2}, ... {n}). At least one of these fields must appear in the story more than once, which works just as you might hope using the Python string format method.
4. For example, your story string might look like something like:

*'''Be kind to your {0}-footed {1},*

*For a {2} may be somebody's mother.*

*Be kind to your {1} in the {3},*

*Where the weather is always {4}.''’*

1. When the program runs, it should ask the user to input various words for the categories that will replace the blanks in the template. The program can prompt for that information with questions or commands like:

*Enter your choice of noun:*

*Enter your choice of noun:*

*Enter your choice of noun:*

*Enter your choice of place:*

*Enter your choice of adjective:*

1. After the user has entered all of the words, the program should then display the completed story on the screen. For example, suppose the user entered:

*Enter your choice of noun:* ***dog***

*Enter your choice of noun:* ***table***

*Enter your choice of noun:* ***lamp***

*Enter your choice of place:* ***phone booth***

*Enter your choice of adjective:* ***tall***

1. The completed story becomes:  
    *Be kind to your* ***dog****-footed* ***table****,  
    For a* ***lamp*** *may be somebody's mother.  
    Be kind to your* ***table*** *in the* ***phone booth****,  
    Where the weather is always* ***tall****.*
2. **Hint:** The various words that are input from the user for each category might best be stored in a **list** of **strings**, so you can have access to them. For example, once the data is input by the user, the list in the example above will look like:

["dog", "table" "lamp", "phone booth", "tall"]

1. Think about how one could get this list constructed from data entered by the user.
2. You may design your code however makes sense to you, but realize that there are much easier ways and much harder ways to design this program, so please design before you implement! A healthy discussion between partners should prove fruitful. Get it?

## Be sure to also note the following:

* Look carefully at [t7-palindrome.py](https://drive.google.com/open?id=0B0J8Yj0B6KRSZXJxbHl3bkliQ3c) to see some string functions being used.
* Include comments for any parts of the code that is non-intuitive to let the reader know what that portion does.
* Make sure that each function has parameters that make sense. If a function does not need a parameter, do not include one.
* Use meaningful variable names.
* Include a descriptive header as a comment at the top of your source code.
* Include a main() function.
* The highest level of your program (i.e., no indenting) should **only** contain the following:
  + the header
  + any import statements
  + function definitions
  + a call to the main() function
* Use functions for encapsulation with triple-quoted docstring for each of these functions. This docstring must include a description of the main purpose of the function, and descriptions of all input parameters as well as what is returned by the function.

## Submission Instructions

1. (Submitter) Save your codes as **T7\_madlibs\_*usernames*.py**. Replace *usernames* with your Berea usernames. For example, the TA Bianca Marrero and I would name ours **T7\_madlibs\_heggens\_marrerob.py.   
   NOTE:** Incorrect filenames will automatically reduce your grade by 1 point. Fortunately, the format is always the same no matter what the assignment.
2. (Submitter) Upload the Python file to Moodle by the due date listed on the course website: <https://trello.com/b/w7bIrLoV/>.
3. (All Other Partners) Open up Wordpad. Create a new text document (.txt) and include all members names in it.
4. (All Other Partners) Save the document as **T7\_madlibs\_*usernames*.txt**. Replace *usernames* with your Berea usernames. For example, the TA Bianca Marrero and my document would be named **T7\_madlibs\_heggens\_marrerob.txt.**
5. (All Other Partners) Upload the document to Moodle by the due date listed on the course website: <https://trello.com/b/w7bIrLoV/>.