CS 1301

# Homework 3 – Iterations, Conditionals, and Control Flow

## Due: September 19th by 11:55pm

**File to submit to TSquare: HW3.py**

You should work individually on this assignment. You may collaborate with other students in this class. Collaboration means talking through problems, assisting with debugging, explaining a concept, etc. Students may only collaborate with fellow students currently taking CS 1301, the TA's and the lecturer. You should not exchange code or write code for others. For individual assignments, each student must turn in a unique program. Your submission must not be substantially similar to another student's submission. **Collaboration at a reasonable level will not result in substantially similar code.**

For Help:
- TA Helpdesk – Schedule posted on class website.
- Email TA's or use Piazza Forums

Notes:

- **Do not forget to include the required collaboration statement** (outlined on the syllabus).
- **Do not wait until last minute** to do this assignment in case you run into problems.
- **Read the entire specifications document before starting this assignment.**

## Function Name: **mixtapeFire**

Description:
Write a function that determines how fire is your mixtape. The user will enter the number of times the mixtape has been played and a rating that people have given it from 0-10. If the number of times played is **10000 or greater AND the rating is 6 or greater,** the function should **return** "Your mix tape is fire!" If either of these categories are not met, the function should return "You should quit the rap game." Finally, if the rating is greater than 10, the function should return "Invalid input. Try again." You may assume that the user will only enter positive integers.

Parameters:

- timesPlayed – an integer representing the number of times people have listened to your mixtape.
- rating – an integer value representing a user rating between 0 and 10.

Return Value:

"Your mix tape is fire!"

or "You should quit the rap game."

or "Invalid input. Play again."

Test Cases:

```
>>>mixtapeFire(10000, 7)
"Your mix tape is fire!"
>>>mixtapeFire (25000, 11)
"Invalid input. Try again."
>>>mixtapeFire (600, 2)
"You should quit the rap game."
```

## Function Name: **myFavSong**

Description:

Write a function that will have a song as a parameter and will ask the user to guess that song. The second parameter should be the artist who wrote the song, and will be used if the user needs a hint. The user will have a **maximum of 5 chances to guess** (after the 5th try, if it's wrong let the user know he has exceeded the number of tries). The user may ask for a hint by writing (hint, Hint or HINT) instead of a song. When the user asks for a hint, use the print function to write "The artist of this song is (whatever artist you wrote for the second parameter)".

If the user guesses the song correctly, say "Great Job!" and how many tries AND hints it took him/her to guess it. Capitalization should not matter when the user guesses the song (i.e. "One Night" is the same as "one night"). No matter what happens, at the end of the game thank the user for playing. Be sure to thank your player for playing the game! **You must use a while loop.**

**Hint:** Built-in functions like .lower() might be helpful.

Parameters:

- song – a string representing the song to be guessed
- artist – a string representing the artist who wrote the song

Return Value:

None

Test Cases:

```
>>> myFavSong("Needed Me", "Rihanna")

Guess my favorite song: Bad Blood

Try again. Guess my favorite song: needed me

Great Job! It took you 2 tries and 0 hints to guess my
favorite song.

Thank you for playing!
```

>>> myFavSong(“Minnesota”, “Lil Yachty”)

Guess my favorite song: **No problem**

Try again. Guess my favorite song: Hint

The artist of this song is: Lil Yachty

Try again. Guess my favorite song: **Minnesota**

Great Job! It took you 2 tries and 1 hints to guess my favorite song.

Thank you for playing!

>>> myFavSong (“System Blower”, “Death Grips”)

Guess my favorite song: **Father Stretch my Hands**

Try again. Guess my favorite song: **Controlla**

Try again. Guess my favorite song: **BANG BANG BANG**

Try again. Guess my favorite song: **Brocolli**

Try again. Guess my favorite song: **Pop Style**

You have exceeded the number of tries.

Thank you for playing!

## Function Name: **illuminatiConfirmed**

Description:

Write a function that takes a secret message and determines whether it was sent from the Illuminati. **You must use a for loop** to determine whether there are three "@" signs in the message. If there are three "@" signs, create a new string from the secret message in which you replace each "@" sign with an "a". Case does not matter. Print this new string and then return "Illuminati Confirmed." If there are not three "@" signs, simply print the old string and return "Probably not the Illuminati."

**Hint:** Read the instructions carefully. You will be printing and returning depending on the condition in this function.

Parameters:

secretMsg – a string representing the message that may or may not be from the Illuminati

Return Value:

A string confirming if the illuminati exists or the original string.

```
Test Cases:

>>>newString = illuminatiConfirmed("Justice for H@r@m-
b@e!!")

Justice for Haram-bae!!

print(newString)

Illuminati Confirmed.

>>>newString2 = illuminatiConfirmed("Email addresses use the
"@" sign.")
Email addresses use the "@" sign.

print(newString2)

Probably not the Illuminati.
```

## Function Name: **decNum**

Description:

Write a function that takes a string as the first parameter and a single digit integer as the second parameter. Assume that the single digit is greater than 0. If the number is in the string, the function will replace it with that number **decremented by one**. The function will **return** this new string with the replacements. If the number is not in the string, the function will return the string "Try a different number."

Parameters:

aString – a string the user enters

aNum – number which will be decremented

Return Value:

newString

Test Cases:

```
>>>decNum("There are 5290 feet in a mile.", 9)
"There are 5280 feet in a mile."
>>>decNum("There are 377 days in 2016.", 7)
"There are 366 days in 2016."
>>>decNum("There are 377 days in 2016.", 8)
"Try a different number."
```

Function Name: **numberTie**

Parameters:

num – An integer between 2-9 specifying half the width of the tie.

Return Value:

None

Description:

Write a function that takes half the width of the tie as a parameter. The function will then draw a tie using three number pyramids (one upside down, one right-side up, and another upside down) using the print function. See below in the test cases for clarification. **Do not hard code this. You must use a for-loop.**

**Hint:** string manipulation and formatting may be helpful. Other Hint: you can break this down into three for loops, one for each pyramid.

Example print output with `numberTie(5)`

```
5555555555
 44444444
  333333
   2222
    11
    11
   2222
  333333
 44444444
5555555555
5555555555
 44444444
  333333
   2222
    11
```

Function Name: countDown

Parameters:

start – an integer marking the starting number

limit – an integer marking the minimum ending number

decrement – an integer marking the size of the decrement

Return Value:

None

Description:

Write a function that takes in three parameters: a starting number, a limit (the minimum ending number), and an increment. The function must decrement from the starting number until it is below the limit. The function must print out all the numbers that are more than or equal to the limit. The numbers should be printed **in the same line separated by commas**. It is okay if the output wraps around into a new line (the (major) key is to print a single string).

Test Cases:

```
>>>countDown(55, 22, 5)
55,50,45,40,35,30,25

>>>countDown(77, 48, 10)

77,67,57
```

# Grading Rubric

## 1. mixtapeFire

| | |
|---|---|
| Correct Heading | 1pts |
| Handle Invalid scores (rating >5) | 3pts |
| Determines which string to return based on values | 4pts |
| Returns a value of type String | 2pts |
| **TOTAL** | **10 PTS** |

## 2. myFavSong

| | |
|---|---|
| Ask the user to guess the song (user-generated) | 2pts |
| Uses a while loop | 2pts |
| Handles when tries have been exceeded | 2pts |
| Gives user hints if asked | 5pts |
| Handles correctly when user guesses the age (prints both number of tries and number of hints) | 5pts |
| Not case sensitive for hints or answers | 2pts |
| Prints thanks at the end | 2pts |
| **TOTAL** | **20 PTS** |

## 3. illuminatiConfirmed

| | |
|---|---|
| Iterates through the parameter string with for loop | 2pts |
| Determines if there are three "@" signs | 5pts |
| Correctly **prints** either replaced string or original string | 5pts |
| Correctly **returns** whether this is the Illuminati | 3pts |
| **TOTAL** | **15 PTS** |

**4. decNum**

| | |
|---|---|
| Correct header and parameters | 2pts |
| Function asks user to input a number | 2pts |
| Function correctly decrements number | 7pts |
| Returns new phrase if number present | 2pts |
| Returns correct statement if number not present | 2pts |
| **TOTAL** | **15 PTS** |

**5. numberTie**

| | |
|---|---|
| Correct header and parameters | 2pts |
| Uses three loops (2pts/loop) | 6pts |
| Prints the correct format of a tie (generally looks like a tie) | 12pts |
| **TOTAL** | **20 PTS** |

6. countDown

| | |
|---|---|
| Correct header and 3 parameters | 2pts |
| Function prints start number and decrements up to limit | 6pts |
| Numbers separated by commas (-3 pts for comma at end) | 6pts |
| Numbers printed on same line | 6pts |
| **TOTAL** | **20 PTS** |

FOR ALL PROBLEMS: (-10) for each syntax error per function.