**IF100 – Fall 2021-2022**

**Take-Home Exam 2   
Due November 26th, 2021, Friday, 23:59 (Sharp Deadline)**

**Introduction**

The aim of this take-home exam is to practice decision making (conditional if statements), sequences, lists and methods. The use of if statements, string methods and lists are due to the nature of the problem; that is, you cannot finish this take-home exam without using them.

**Description**

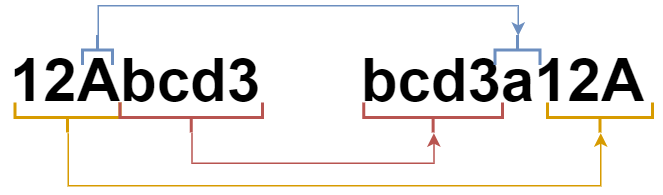
Assume that you are an influencer on Instagram and you want to increase your follower count by accepting new follow requests. However, you want this growth to be an organic one. Because of this, before giving permission to an account to follow you, you want to check whether there is a suspicious case in his/her (requester's) actions or not. And also, you want to grow in a specific topic/keyword; that is the particular type of posts that you would like to have access to. Because of this, you will only permit the accounts that belong to the requested category type. Luckily, there is a database in your local driver that keeps information about accounts. Due to security issues, names of these accounts are sorted in an encrypted form. Inside of this database, you are holding the name of the account, category(ies) that the account sends posts about, and the follower and following numbers.

In this task, you are asked to write a program that will take the database, username and category type as inputs, and will return whether this account has a potential of being bot or not. There are also some validity checks that your program needs to consider. First of all, your program needs to check if the username (obviously, the encrypted version) exists in the database or not. If it does not exist, then, your program should give an appropriate error message. Otherwise, i.e. if the encrypted username exists in the database, then your program needs to check if the provided account is a bot or not. To do this, you will calculate the ratio of the following/follower for the given Instagram account. If this ratio is greater than or equal to a predefined threshold, which is 10, then, your program will consider it as a bot account, and you will not accept his/her request. Otherwise, i.e. if the ratio is strictly smaller than the threshold (which is 10), then your program will ask for another input, the category of posts, that you want to take into consideration while deciding on whether to permit the account. If the entered category does not exist in the database, then your program will give an appropriate error message. Finally, if those of the categories match, then your program will indicate the fact that you can accept that account to be your new follower.

For this take-home exam (THE), you will write a Python program that will check whether the user’s account’s follow requests can be accepted or not, by taking the following three (3) information as inputs:

* Database that contains (i) usernames (encrypted version in a predefined format), (ii) type(s) of posts they produce (note that an account can belong to more than one type), (iii) follower count and (iv) following count.
* Account name that we want to make a decision over.
* Type that we want to consider.

Encrypted usernames are constructed by scrambling the original usernames in a specific way. First of all, we partition the username into two (2) from its middle. That is, we calculate the length of the word and divide it by two (2) via integer division. If we call this value as mid, then the first mid characters of the word will be the first part, and the rest of the characters will constitute the second part, as shown below. Now, the scrambling works as follows: we concatenate (i) second part of the username, (ii) lowercase version of the last letter of the first part of the username, and (iii) first part of the username.

****

For example, when the username is **12Abcd3**, the crypred username will become **bcd3a12A**.

**Inputs**

The inputs of your program and their order are explained below. It is extremely important to follow this order with the same characters since we automatically process your programs. ***Thus, your work will be graded as 0 unless the order is entirely correct***. Please see the "Sample Runs" section for some examples.

The prompts of the input statements to be used have to be exactly the same as the prompts of the "Sample Runs".

Here is the detailed information on the inputs and the input checks:

* Database - *Encrypted username, type(s), number of followers and number of followings  
  Format: username1:type1.1-type1.2:follower1:following1,.................,usernameN:typeN:followerN:followingN*   
   i.e.**"oodfDlf:food-cousine:1037:705,efulpFhop:travel:60:1124,wisemMUSM:food:4525:1124,ulleapfhelpf:travel-food-photography:10809:107,lexiblefstylef:art:509:320"**
  + Colon (":") is used to separate the username, type(s), follower count and following count from each other.
  + Comma (",") is used to separate the accounts from each other.
  + Dash ("-") is used to separate types from each other. However, keep in mind that some accounts may not belong to multiple types and **dash will not be used** on those accounts.
  + You may assume that there won't be any duplicated usernames within this input. You may also assume that the encrypted usernames, type(s), follower count and following count will not contain any punctuation marks.
  + A particular information will not have the separator characters (":", ",", "-").
* Username. *Format: username*  
   i.e. **"styleflexible"**
  + Keep in mind that the length of the username can be **anything**. Since integer division will be executed on the username, it does not require an even length.
* Type. *Format: type*

i.e. **"food"**

You may check *Sample Runs,* for more examples. However, please ***keep in mind*** that sample runs may not cover all possible cases mentioned in this document.

**Output**

If the encrypted version of *username* does not exist inthe *database,* thenyour program should display an error message saying

"*Username does not exist!*"

If the encrypted version of *username* exists inthe *database,* but the ratio of following/follower is higher than 10, then your program should display an error message saying

"*There is a high chance that username may be a bot!*"

If the encrypted version of *username* exists inthe *database*,and the ratio of following/follower is lower than 10, and *type* exist in database but *type* does not match with the chosen account’s type(s), then your program should display an error message saying

"*Interests do not match with username!*"

In any erroneous case, your program should terminate after displaying the respective error message and *without taking any further inputs* or *without displaying any other results*.

If the encrypted version of *username* exists inthe *database,* and the entered *type* matches with *username’s type*, and the ratio of following/follower is lower than 10, *then your program should display the following message.*

"*It is safe to accept the following request of username.*"

You may check the "Sample Runs" section given below for some examples.

**Sample Runs**

Below, we provide some sample runs of the program that you will develop. The *italic* and **bold** phrases are inputs taken from the user. You have to display the required information in the same order and with the same words and characters as below.

**Sample Run 1**

Please enter the database: **oodfDlf:food-music:1037:705,efulpFhop:travel:60:1124,wisemMUSM:food:4525:1124,ulleapfhelpf:travel:10809:107,lexiblefstylef:art:509:320**

Please enter the username: **MUSMwaise**

Username does not exist!

**Sample Run 2**

Please enter the database: **oodfDlf:food-music:1037:705,efulpFhop:travel:60:1124,wisemMUSM:food:4525:1124,ulleapfhelpf:travel:10809:107,lexiblefstylef:art:509:320**

Please enter the username: **Fhopeful**

There is a high chance that Fhopeful may be a bot!

**Sample Run 3**

Please enter the database: **oodfDlf:food-music:1037:705,efulpFhop:travel:60:1124,wisemMUSM:food:4525:1124,ulleapfhelpf:travel:10809:107,lexiblefstylef:art:509:320**

Please enter the username: **styleflexible**

Please enter the type you are interested in: **food**

Interests do not match with styleflexible.

**Sample Run 4**

Please enter the database: **oodfDlf:food-music:1037:705,efulpFhop:travel:60:1124,wisemMUSM:food:4525:1124,ulleapfhelpf:travel:10809:107,lexiblefstylef:art:509:320**

Please enter the username: **helpfulleap**

Please enter the type you are interested in: **travel**

It is safe to accept the following request of helpfulleap.

**Sample Run 5**

Please enter the database: **oodfDlf:food-music:1037:705,efulpFhop:travel:60:1124,wisemMUSM:food:4525:1124,ulleapfhelpf:travel:10809:107,lexiblefstylef:art:509:320**

Please enter the username: **Dlfood**

Please enter the type you are interested in: **food**

It is safe to accept the following request of Dlfood.

**Sample Run 6**

Please enter the database: **oodfDlf:food-music:1037:705,efulpFhop:travel:60:1124,wisemMUSM:food:4525:1124,ulleapfhelpf:travel:10809:107,lexiblefstylef:art:509:320**

Please enter the username: **Dlfood**

Please enter the type you are interested in: **music**

It is safe to accept the following request of Dlfood.

**Sample Run 7**

Please enter the database: **oodfDlf:food-music:1037:705,efulpFhop:travel:60:1124,wisemMUSM:food:100:1000,ulleapfhelpf:travel:10809:107,lexiblefstylef:art:509:320**

Please enter the username: **MUSMwise**

There is a high chance that MUSMwise may be a bot!

**What and where to submit?**

You should prepare (or at least test) your program using Python 3.x.x. We will use Python 3.x.x while testing your take-home exam. Let us repeat,

* You must use Google Colab to develop your code from scratch (from beginning till the end), and then submit it **through SUCourse+ only**! Once you are done with developing your code on Google Colab, then you will copy your code to the CodeRunner to see if your program can produce the correct outputs. At the end, you will submit your code through CodeRunner (and SUCourse+). You should keep your Google Colab file until the end of the semester, we might want to look at this. If you fail to provide this Google Colab file anytime in the semester, you may not earn any credits from this Take Home Exam.
* In the CodeRunner, there are some visible and invisible (hidden) test cases. You will see your final grade (including hidden test cases) before submitting your code. Thus, it will be possible to know your THE grade before submitting your solution.
* **There is no re-submission**. You don't have to complete your task in one time, you can continue from where you left last time but you should not press submit before finalizing it. Therefore, you should make sure that it’s your final solution version before you submit it.
* **General Take-Home Exam Rules**
* Successful submission is one of the requirements of the take-home exam. If, for some reason, you cannot successfully submit your take-home exam and we cannot grade it, your grade will be 0.
* There is NO late submission. You need to submit your take-home exam before the deadline. Please be careful that SUCourse+ time and your computer time may have 1-2 minutes differences. You need to take this time difference into consideration.
* You must use Google Colab to develop your code from scratch (from beginning till the end), and then submit it **through SUCourse+ only**! Once you are done with developing your code on Google Colab, then you will copy your code to the CodeRunner to see if your program can produce the correct outputs. At the end, you will submit your code through CodeRunner (and SUCourse+). You should keep your Google Colab file until the end of the semester, we might want to look at this. If you fail to provide this Google Colab file anytime in the semester, you may not earn any credits from this Take Home Exam.
* Do NOT submit your take-home exam via email or in hardcopy! SUCourse+ is the only way that you can submit your take-home exam.
* If your code does not work because of a syntax error, then we cannot grade it; and thus, your grade will be 0.
* Please submit your **own** work only. It is really easy to find "similar" programs!
* Plagiarism will not be tolerated. Please check our plagiarism policy given in the syllabus of the course.

Good luck!

Özgün Yargı

& IF100 Instructors