Project 1

Objectives:

This project is designed to reinforce understanding of elementary Python constructs such as the creation and use of variables, console input/output, data types, loops and iteration, conditional expressions, dictionaries, classes, and general top-down program design.

Task

Implement a python script that creates a ***dictionary*** of records representing several users who have requested access to one or more of the servers you manage as a System Administrator. Ideally, this might be implemented as an online form, but we are going to assume we have the information in some other form such as an email response. Imagine that this program commits those records to a database or other permanent storage. For now, we will just print the user-provided details to the screen as shown in the sample run below.

Create an Employee class with which employee objects will be created. This class should include the following attributes: employee id, first name, last name, department code, supervisor name, temporary password, the server they need to connect to, and the Linux account ID. Ensure that a constructor is created for the class. It is left to you to decide the parameters of this constructor. No other class methods are required (though you may want to create headers and getters for the attributes that are not specified for the constructor). For each employee requested, the program should prompt the user (the system administrator) for the required information, create an Employee object for that employee and store the record in a ***dictionary***. The key to this dictionary should be the employee's ID. This requires that the employee ID has no duplicates **(i.e., validate the employee ID)**. The program should keep creating more employee objects until the user indicates otherwise. After the last employee object is created, your script should create the report shown in the sample run below.

The sample shown on pages 2 and 3 accepts inputs for three (3) employee records and prints the required summary reports.

Sample Run

```
./create_user_records.py
Enter the user's employee id:  589
Enter the user's first name:  John
Enter the user's last name:  Smith
Enter the user's department number:  42
Enter the user's supervisor (full name):  Hank Jones
Enter the user's Linux account ID:  smithj1
Enter the user's temporary password:  H0lyM0$es
Enter the name of the server they've requested: t.u.com
Would you like to create another record (type yes or Y to continue): Y
Enter the user's employee id:  472
Enter the user's first name:  Jane
Enter the user's last name:  Doe
Enter the user's department number:  54
Enter the user's supervisor (full name):  Art Tatum
Enter the user's Linux account ID:  doej1
Enter the user's temporary password:  $ecrEtW0rd
Enter the name of the server they've requested: tt.u.com
Would you like to create another record (type yes or Y to continue): Y
Enter the user's employee id:  467
Enter the user's first name:  John
Enter the user's last name:  Doe
Enter the user's department number:  42
Enter the user's supervisor (full name):  Hank Jones
Enter the user's Linux account ID:  doej2
Enter the user's temporary password:  0bfu$c8tion
Enter the name of the server they've requested: t.u.com
Would you like to create another record (type yes or Y to continue): N
```

Summary Report of User Records

==========

3 records created:

Department Code	Employee Count
42	2
54	1

Servers Requested	Employee Count
tt.u.com	**1**
t.u.com	**2**

Employee Details

==========

Name: John Smith
Employee Id: 589
Department: 42
Supervisor: Hank Jones
Username: smithj1

Name: Jane Doe
Employee Id: 472
Department: 54
Supervisor: Art Tatum
Username: doej1

Name: John Doe
Employee Id: 467
Department: 42
Supervisor: Hank Jones
Username: doej2

Grading Rubric (Requirements)

1. Validate employee ID ***5pts***
2. Create Employee class ***10pts***
3. Implement a ***function*** (not a class member) to capture user records and populate the dictionary using requirements 1&2 above. ***10pts***
4. Implement a ***function*** (not a class member) that displays a simple report with the general format and header as shown in the sample run ***10pts***
5. Do not hardcode the number of records or prompt the user to enter the number of records. ***5pts***
6. Provide meaningful comments at the block level. Zero comments, comments on every line, or extraneous trivial comments will result in a point deduction up to the full value allotted. ***10pts***

Code Submission instructions

(1) I would prefer that you write the code and test on your VM.

(2) Include the interpreter line `#!/usr/bin/python3`

(3) Write a comment block in the file near the top of the script which documents the following:

```
# Program name
# Your name
# Course number, section, and semester
# Date completed
```

(4) Submit your script on Canvas