**LAB 2 – Writing a Complete Script in Python**

**Marks = 2**

**Submission =**

* **For this lab, you must work individually in the class, however, the final submission must be done as a group.**
* **Part 1 must be completed and checked in the class. Once you complete part 1, keep working on the other parts.**
* **After completing all the parts, submit your files (python files for all the parts) on Brightspace before the due date. Only one group member should submit the files. Only upload “.py” or “.txt” files.**

**Part 1 – Working with date and time types in Python.**

For this exercise, carefully read about the date and time types in Python from the following link: <https://docs.python.org/3/library/datetime.html>

**Tasks:**

1. Create a basic ‘timedelta’ and print it (expected result for example: 365 days, 5:01:00).
2. Print today’s date and current time (Ex. Today is: 2023-01-12 00:23:50.465243).
3. Print today’s date and time two years from now (Ex. Two years from now it will be: 2025-01-11 00:33:09.411866).
4. Create and print a timedelta with two arguments. E.g., In 2 weeks and 3 days it will be: 2023-01-29 00:33:09.411866. In this example, the output is a timedelta that generates its result based on two arguments ‘weeks and days’. The result is also randomly generated based on current date and time. Thus, your code should also perform something in a similar way.
5. Calculate the date three weeks ago and print it like a string. Ex. Three weeks ago it was Thursday December 22, 2022.
6. Here you need to find the number of days till next Christmas. Your code should first check if Christmas has already gone for this year. If this is the case, then you should replace this year’s Christmas date with next year’s date. In the end, your code should print the number of days left till the next Christmas. Ex. 347 days left till next Christmas.

**Part 2 - Write a complete Python script, with comments, to do the following**:

1. Open a text file called “*grocerylist.txt*”, attached with this lab, for reading. The file contains the *category* of available items (ex. tea), the list of *items* (ex. iced tea), and the available *quantities* (ex. 150), in a grocery store.
2. Define a list of strings L1 as follows: ***L1*** **= ['tea', 'bakery', 'soaps', 'rice', 'candies', 'lotions']**
3. Create an empty dictionary as *dict* **d1 = {}**
4. In a loop, do the following:
   1. Check each *category* in **L1** and see if that *category* matches any of the *categories* in the file.

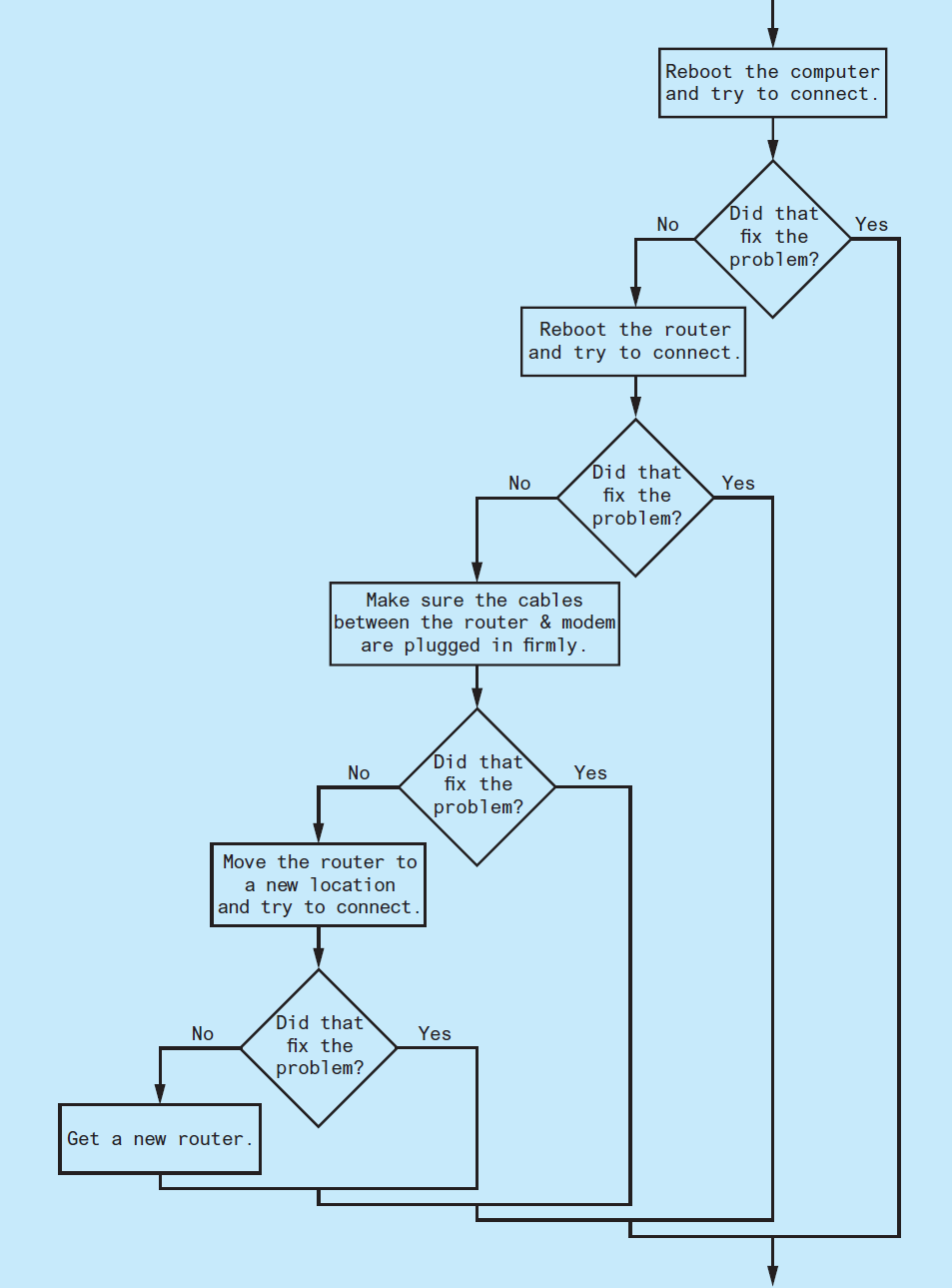
* **Hint**: use the function readline() to read a new line from the file and compare that line with the elements of **L1**.
  1. If there is a match, save the *item* and the *quantity*corresponding to that *category* in some variables. Create a *key-value* pair ***category:(item, quantity)*** for dict ***d1.*** Here, ***category* (**i.e*.* the *key***)** is the string found in *grocerylist.txt* and the tuple (***item, quantity)*** (i.e. the *value*) corresponds to the *item* itself with its *quantity*. Add the *key-valu*e pair to ***d1*** as {***category:(item, quantity)***}.
  2. Otherwise, if a *category* is not found in *grocerylist.txt*, print a suitable message indicating that the specified *category* is not available.

1. Next the program/script should ask the user to enter a string ***s***, (representing an item *category* such as “tea”) as an input.
2. If ***s*** corresponds to a valid *key* in ***d1***, then the program should retrieve the *item* and the *quantity* of ***s*** from ***d1*** and display it to the user.

* After displaying the *item* and the *quantity* corresponding to item ***s***, the program asks the user to enter a number ***n*** indicating how many items he/she would like to order.
* If ***n*** is not an integer, the program should catch the exception, display an appropriate error message, and exit.
* If ***n*** is a negative integer, the program should print ‘negative inputs are not accepted’ and then prompt the user to input another value for ***n***.
* If ***n*** is a positive integer, the program should print ‘Congratulations – Your order is successfully placed’ and then exit.

1. If an item’s *category* entered by a user **does not** correspond to a valid key, the program should catch an exception. When the exception occurs, the program should display an appropriate error message to the user and then prompt him/her to input a different *category* of an *item*.

**Part 3 - Write a python program for the Wi-Fi router at your home. The program should work according to the following figure which shows the steps to fix a bad Wi-Fi connection at your home.**

****