2.5 Python Assignment

Program documentation (Countdown timer)

Author: Kenan Kozlica

In completing this assignment, I used the following resources and tools:

- Shovic, John Carrol; Simpson, Alan (2024). *Python essentials for dummies*. Hoboken (N. J.) : J. Wiley & Sons, cop. 2024
- Borjigin, Chaolemen (2023). *Python Data Science*. Singapore : Springer Nature, cop. 2023
- ChatGPT
- Stratvert, Kevin. (25. 3. 2021). 👩‍💻 *Python for Beginners Tutorial*. YouTube. https://www.youtube.com/watch?v=b093aqAZiPU

I would like to sincerely thank Dr. Uroš Kunaver from the Central Technical Library at the University of Ljubljana for all his advice and assistance in completing this assignment.

Introduction

The purpose of this assignment was to develop a simple program that would demonstrate the use of the Python programming language for automating simple tasks.

The Waterfall model, which is a linear software development model, was used to develop the program. First, we defined the requirements, which included the ability to enter a time or date for real-time countdown, display of the remaining time, and a final message. In the analysis phase, we checked how the program would work and defined the logic for countdown and calculating the remaining time. In the design phase, we defined the structure of the program, which includes importing libraries, the user interface, the main countdown loop, displaying the remaining time, and ending the program. During implementation, we wrote code in Python that uses loops and functions to calculate and display the remaining time. Testing the program involved various inputs, verifying that the program correctly displays the remaining time and stops when finished.

Problem description

The task is to create a countdown timer that has two functions. The first function allows the user to enter any number of seconds. After the value is entered, the countdown timer counts down until the entered time expires. The second function enables the user to enter one or more times (date and time). The countdown timer then shows how much time is left for all the entered times. The countdown timer runs until all counters reach zero (0). The first function of the program could be useful for counting the time available for a task (e.g., the duration of physical exercise, etc.). The second function of the program could be useful for websites where sales take place. In this way, the time until the end of the promotional period could be counted down.

Program design

The program is written in the Python programming language and runs in the command line. The main purpose of the program is to demonstrate the operation of a countdown timer written in Python. After starting the program, the user chooses one of the two functionalities (seconds countdown timer or countdown timer for one or more times (defined by date and time)). If the user selects the first functionality, he enters the number of seconds that should pass until the end of the countdown. After the entry is made, the countdown timer starts counting down and displays how much time is left, showing the time in the form of days, hours, minutes, and seconds. The program ends when the time runs out and the message "Time is up!" is displayed. If the user selects the second function, he can enter multiple times (dates and times, in the format YYYY-MM-DD HH:MM:SS). In addition to manually entering individual dates, the user can also specify a text file (e.g., a .txt file) in which the dates and times are stored in separate lines in the same format as described above. The user does this by entering "file:<filename>" (e.g., "file:C:\Users\Kenan\Desktop\datumi.txt") instead of the date. An example of a text file (datumi.txt) is included in the documentation. The user ends the date entries by entering the word "end" instead of the date and time. The timer then displays the target times and the time remaining until each target time. In the output, all times are displayed in a single line, one after the other. The times are updated every second. The remaining time is displayed in the form of days, minutes, hours, and seconds. The program runs until all timers expire. When an individual timer expires, the text "Time is up!" is displayed in its place.

The program is 124 lines long. It uses the time and datetime libraries. The time library allows the program to wait one second between countdown updates, while the datetime library allows the calculation of the remaining time. The program uses an array where the user entries (date and time) are stored. The program uses several types of loops: it uses a "While" loop to read user input, to read lines from the selected file with times, and to count down the time. The "For" loop is used to parse the array containing target times. The program uses the "If" statement in several places: to decide on the selected functionality, to detect the end of date and time input, to detect file input, and to detect the end of remaining time. The program reads the times from the file if the user selects this option. In addition to the main function (main_program), the program uses three auxiliary functions with the following interfaces:

· convert_to_dhms (seconds) -> days, hours, minutes, seconds

· countdown_seconds (duration)

· countdown_to_date (target list [])

At the beginning of the program, a simple flowchart is drawn in the comment section.

Program testing/verification

The program was tested using various data. To test the first functionality (counting a specified number of seconds), I entered various values ranging from a few seconds, a few minutes, a few hours, to several days. In each case, I observed the program's operation and checked the correctness of the calculation of the number of days, hours, minutes, and seconds from the the remaining seconds until the end of the countdown. I also tried to enter zero and negative values, to which the program responded correctly (displaying a message about an invalid number of seconds), and I also entered an invalid date and time format. In this case also, the program responded correctly (displayed a message about an incorrect entry).

To test the second functionality, I selected different times in the future, one just a few seconds, another a few minutes, a few hours, and several days away. I observed the program's operation and checked the correctness of the calculation of days, hours, minutes, and seconds from the difference between the current time and the target time. In addition, I also checked the reading of different times (in the format YYYY-MM-DD HH:MM:SS), entered directly from the keyboard, as well as entered via a file. I also tested various combinations of manual entry and entry from files, and in all cases the program worked as expected (i.e., all target times were entered into the program in the order in which they were entered, regardless of the input method). In addition, I also tried entering times that were before the program launch time (i.e., in the past) and entries in the wrong format (i.e., different from YYYY-MM-DD HH:MM:SS). In both cases, the program responded correctly by displaying the appropriate error message. During testing, I noticed only one error or malfunction. In case of too many target dates the display of all the countdown timers extends beyond the width of the console window and some of the countdown timers are displayed in a new line. As a result, the next time the line is refreshed, the countdown timers are not displayed on the same line as before (which should make it look like they are in the same place), but instead they are displayed on a new line. Since this behaviour only occurs with a large number of counters (depending on the width of the window) and since the purpose of the program is only to demonstrate how such a counter could work, I did not fix this error, as it would require a complete change in the way the counters are displayed.