PROBLEMS - Lesson 4: Data structures

Problems in yellow → Assessable problems (evaluable)

The rest of the problems are of intermediate difficulty. If you have any difficulty moving from one recommended problem to the next one, do some of the problems that appear in between for more progressive learning.

Problems in blue → Optional problems (not assessable but available at VPL for autoevaluation).

Problem 1

Make a program that asks the user to enter a tweet via keyboard (a string). The program will extract all hashtags and print each one on a separate line. For example, if you enter the string "Hello #goodmorning #IP" the program will print:

```
#goodmorning
#IP
```

Problem 2

Make a program that creates a tuple with every month of the year (January, February, March, April, May, June, July, August, September, October, November, December).

Then, the program should ask the user to enter a date in DD/MM/YYYY format (day, month, year separated by '/'). It should then display the date entered in the format: "NAME_MONTH DD, AAAA" where DD and YYYY are the day and year, and NAME_ MONTH is the full name (January, February, etc.) of the month number entered by the user. If the input month number is incorrect, print the message "Error: Incorrect month".

Problem 3

Make a program that implements "The sieve of Eratosthenes", an old algorithm to find all prime numbers up to a certain integer (MAX), which will be introduced via keyboard.

Algorithm:

1. *Analyze the numbers in the range [2, MAX]*
    a. *If the number is not included in the list NP (not prime), then it is prime, so we add it to the list P (prime) .*
    b. *We add all the multiples of this number into the NP list.*
2. *Print list P.*

Problem 4

Make a program that asks the user to enter a string via keyboard and count how many vowels there are in the string entered by the user. The program will display the final result with a message with the format: "The string "XXXXX" has NN vowels." where XXXXX is the string entered by the user and NN is the number of vowels contained in the string.

Note: When counting vowels, you do not need to consider accented vowels.

Problem 5

Make a program that asks the user to enter a series of integers and store then in a list. When a 0 is entered, it means that the sequence has finished. Next, check if there is any even number. If so, the program should display the message "Even number." Otherwise, it will display the message "No even number.".

Problems with Strings

Problem 6

Make a program that calculates the letter that corresponds to a DNI number. The program must ask the user to enter the DNI number (no letter) and then calculates the letter that corresponds to this DNI number by dividing the number by 23 and keeping the residual.

The letter is obtained using the residual as an index in the string "TRWAGMYFPDXBNJZSQVHLCKE". The program will display the full ID (number and letter without spaces between them).

Problem 7

Make a program that asks the user to enter a string by keyboard. Then the program asks the user to enter two values, namely A and B, between zero and the length of the string. If any value is invalid (value lower than zero or greater than the string length), the program will ask the user again to enter these two values, until they are correct. Afterwards, the program will display 4 substrings, each one on a different line:

- the characters corresponding to the indices that are between A and B (inclusive).
- the characters corresponding to the even indices between A and B (inclusive).
- the characters in the string ranging from the first character to the index A (included).
- the characters in the string that range from the index B (included) to the end of the string.

Problem 8

Make a program that asks the user to enter three words in lower case and without accents. The program must display the three words, sorted alphabetically and separated by a space between them.

Note: No need to check that the words are lowercase and without accents. We assume that the user will always follow the instructions of the program correctly.

Problem 9

Make a program that asks the user to enter words. Each word will be accumulated (concatenated) in a string so that each word will be separated from the previous one by a blank space.

The word entry will end when the user enters a period "." (a single point). This point will be added to the end of all words, without separation with respect to the last word entered. Finally the accumulated string will be displayed with the format: "Final sentence: YYYY BBBB CCCC DDDD." where YYYY, BBBB, etc. will be the words that the user has entered.

Problem 10

Make a program that reads a string via keyboard and stores it as a single string. The program will then ask the user to enter a position to modify and a value to assign to that position. The program must concatenate the new value with the parts of the string that are not modified, so that the string only updates the indicated position. The process must be repeated until the user enters an invalid position (outside the index limits of the string). Then, the program will display the final string, having done all modifications.

Problems with Tuples

Important: Do not use conditional instructions to do the main part of these problems. But you can use them to check for incorrect values.

Problem 11

The signs of the Chinese zodiac are determined from the result of calculating the result of the year module 12: 0 = "monkey", 1 = "rooster", 2 = "dog", 3 = "pig", 4 = "rat", 5 ="ox", 6 =" tiger", 7 =" rabbit", 8 = "dragon", 9 ="snake", 10 ="horse", 11 ="goat".

Make a program where at the beginning, a tuple is created that contains the signs of the Chinese zodiac, sorted as indicated in the previous paragraph. The program will ask the user to enter the year of birth and then it displays the message "Sign: XXXX" where XXXX will be our sign in the Chinese zodiac.

Problem 12

Make a program that implements a draw of topics in some oppositions to primary school teachers. The syllabus of the examinations consists of 5 topics that the program will save in a tuple such as "Educational models", "Teaching materials", "Relationship with the family", "Language development", "Logical-mathematical development".

When a candidate in the examinations executes the program, a random value between 0 and 4 will be generated, and the subject contained by the tuple in that position is what the candidate will have to develop. The program will inform the candidate with the message "Topic to be developed: XXXX" where XXXX will be the title of the topic that has touched him according to the draw.

Note: To generate a random value, write *"from random import randint"* in the first line of the program and when generating the value, use *"randint (0,4)"*.

Problem 13

Make a program that creates a tuple that contains the names of the days of the week (Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

Then, the program asks the user to enter integers. If the number entered is between 1 and the length of the tuple (i.e. 7), the program displays the name of the corresponding day (1 = Monday, 2 = Tuesday, etc.). Otherwise, it displays the error message "Error: Incorrect day". The program ends when the user enters a zero.

Problems with Lists

Problem 14
Make a program that stores in two separated lists the NIUs of all approved students and all suspended students in a class. At the beginning, the program should ask the user how many students are in the class and then, ask student data. For each student, it will ask the NIU and the mark, and include the NIU in the corresponding list (passed or failed). Once all data has been entered, the program will display the final result in the following format:

```
Passed: AA [NIU_AP1, NIU_AP2, ...]
Failed: SS [NIU_S1, NIU_S2, ...]
```

where AA and SS are the number of passed and failed, NIU_AP1, NIU_AP2 are the NIUs of the students who passed, and NIU_S1, NIU_S2 are the NIUs of the ones who failed.

Problem 15
Make a program that asks the user to enter person names. The program must classify the names entered into two lists according to whether they begin with a vowel or a consonant. If the user enters a name that begins with a character different than vowel or consonant, an error message will be displayed and it will ask to enter a name again.

The program ends when an empty string is entered. Then, the program will display the contents of the two lists.

Note: Just check that the first character is a letter. The rest of the characters within the name do not need to be controlled.

Problem 16
Make a program that asks the user to enter 10 integers that will be stored in a list. Then, the user will be prompted to enter a valid list position (value between 0 and 9), namely A. If the user enters an invalid position, it will show the message: "Error: Invalid position" and it will ask to enter a value until a valid number is introduced.

Then, the program will cut the list by the indicated position by the value A and exchange the two parts. The element in the position A will be in the second part. The resulting list, with the two parts exchanged, will be saved with the same variable name, and the list will be printed.

Example: If the input list is L = [4, 6, 7, 9, 1, 23, 56, 78, 55, 34] and the position is 3, the final content of the list should be L = [9, 1, 23, 56, 78, 55, 34, 4, 6, 7].

Problem 17
Make a program that implements a menu to perform basic operations over a list. The program should ask the user how many items will have the initial list, and then ask the user to enter the values.

After entering the initial values of the list, the program will print the list and show the menu with the available options:

1. Add item at the end of the list
2. Add item in a certain position in the list
3. Remove the last item from the list
4. Remove the item at a certain position in the list
5. Remove the first appearance of a value in the list
6. Exit

Then the user will select one option. Depending on the option, the data to be requested will differ, as follows:

- Option 1 - The item to be added will be requested.
- Option 2 – The program will ask the item to be added and the position where it should be added (**it is important to respect this order**).
- Option 3 - No extra data will be asked.
- Option 4 - The position of the element to be removed will be requested.
- Option 5 - The value to be deleted will be requested.
- Option 6 - No extra data will be asked.

Once performed the corresponding operation, the program will display the contents of the list and the available options. The process will be repeated until the user enters the exit option.

Before exiting the program, the final contents of the list will be displayed.

In options 2 and 4 you need to check that the position is valid. Otherwise, the message "Error: Invalid position" will be displayed.

In option 5, check that the value exists in the list before attempting to delete it. If it does not exist, the message "Error: Value does not exist in the list" will be displayed.

If a non-existent option is entered, the message "Error: Option not available" will be displayed.

Problem 18

Make a program that asks for the dimensions of a matrix (rows and columns) and creates a list of lists, where each element of the main list is another list with the contents of a row of the array.

The program will ask the user to enter the elements (integer values) of the array, row by row. That is, it will ask for the items in the first row, and it will add them to a list. After reading all the items in the row, the created list (which contains all the items in the row) will be added to the main list. The program will do the same for the second row, and so on for the number of rows indicated by the user.

When all elements have been entered, the program will show each element in the main list (which corresponds to a row in the matrix) on a separated line.

Problems of route and search strings and lists

Problem 19
Make a program that asks the user to enter a string and a character. The program will count how many times the character appears in the string. The output message will be: "The character appears NN times." where NN is the number of times the character appears in the string.

Problem 20
Make a program that fills a list of 12 values entered by keyboard. Once the user has entered all values, the program will print the list (separated by a space). Then the program will replace all negative values in the list with a zero. Finally, the program will print the list so that the user can verify that the replacement of the negatives was correct.

Print one list in each line, where elements are separated by a blank space. Put the word "Input:" before printing the input list, and the word "Output:" in front of the updated list.

Problem 21
Make a program that asks the user to enter a string via keyboard and convert all lowercase letters to uppercase. The resulting string must be printed without any other message.

Note: You need the ord() and chr() functions. You can also use the codes in the ASCII table of letters.

Problem 22
A metal plate manufacturing company produces 20 plates per day, and they want to know the percentage of plates that are correctly produced at the end of the day.

Make a program that stores the daily production in a list. The program must display a message "Enter the thickness of the plate N:" to ask the operator to enter the thickness in millimeters of each of the 20 plates produced, and this information is stored in a list. If the operator enters a value equal to or less than zero, the program will ask to enter the value again.

Next, the program must calculate the percentage of plates that have a thickness between 5mm and 10mm (both included) which are the limits set by the company as correct values of the plates.

Problem 23
Make a program that allows to store in a suitable data structure the information about the month in which the babies of a nursery school have started to walk. The program will ask the user to enter the information with the message "Enter the age in months of the baby N when he/she first walked: " and will read the value of each of the 15 babies. The program will check that the measure introduced is valid (between 9 and 18, both inclusive), otherwise, the following error message will be given "Error: Value is not valid" and the value of baby N will be requested again.

Once all data has been entered, the program must calculate the average age at which the babies of the school began to walk and output: "Average age of walking: XX months" where XX will be the average age in months.

Note. In case of having walked before 9 months or after 18 months, the user will introduce 9 or 18 respectively.

Problem 24
Make a program in which the user enters positive integer values to fill in a list of 10 positions. It is necessary to check that the input values are positive. If it is not the case, the program should display an error message and ask the value again until a positive is introduced. Then the program will compute and display the minimum element in the list.

Problem 25
Make a program in which the user enters values to fill a list of 10 positions. Then the program must calculate and display which is the maximum in the list and at which position it is located with the message: "The maximum in the list is MM, located at index PP" where MM is the maximum and PP is the index within the list.

Problem 26
Make a program that stores the ages of members of a gym to make statistics. The program should create a list with as many positions as members in the gym (this value must be a constant defined in the program) and ask the user to introduce all ages. Then, it will calculate and display the following data:

a) Average age of the members
b) Maximum and minimum age
c) Number of people who are of legal age

Problem 27
Make a program that reads the average temperature of the twelve months of the year and stores them in a suitable data structure (temperatures will be integers). The program will calculate and display the month (identified by a number) with the minimum temperature and the one with the maximum one. The format will be as follows:

Month with minimum temperature: MM
Month with maximum temperature: NN

where MM and NN will be the numbers of the months (1 = January, 2 = February, etc.).

Then the program should calculate and display the average annual temperature with the message "Average temperature: MM" where MM is the average temperature. Finally, it should say which months had a temperature below, above or equal to the annual average. The format of the output messages (one for each month) will be:

The month MM had a temperature below the annual average.
The month MM had a temperature above the annual average.
The month MM had a temperature equal to the annual average.

Problem 28
Make a program that asks the user to enter a sentence (string) and calculates the histogram of the distribution of the vowels in the sentence, that is, how many 'a', 'e', etc. are in the sentence. The program must count both lowercase and lowercase vowels. The result will be displayed with a message in the format: "A: X1 - E: X2 - I: X3 - O: X4 - U: X5", where X1 is the number of 'a' and 'A' in the sentence, X2 is the number of 'e' and 'E' in the sentence, etc.

Problem 29
Make a program that asks the user for a sentence (string) and counts the number of words in the sentence. Consider a word as any text separated by the blank space character ' '. Note that words can be separated by more than one space. The output message will have the format: "Number of words: NN" where NN is the number of words in the sentence.

Problem 30
Make a program that asks the user to enter a string and checks if there is any blank space in the string or not. In case any blank is found, the program should display "There is a blank space." Otherwise, it should display the message "There is NO blank space.".

Problem 31
Make a program that reads a string and searches any question marks ('?') inside the string. If so, the program will display the message "There is a question in the string.". Otherwise, it will output "There are NO questions in the string.". Note that the ASCII code of the question mark character is 63.

Problem 32
Make a program that asks the user to enter odd numbers to fill a list of nine positions. We will perform this step with a ***for*** loop that will read the user's values without checking if the values are odd or not. Once the list is filled, the program will check if all values are odd. In this case, the program will display the message "All are ODD". Otherwise, it will display the message "NOT all are ODD".

Important: The program must avoid unnecessary operations. When an even number is detected, the program should not continue checking the remaining elements.

Problem 33
Make a program that stores integer values to fill a list of 12 positions. We will perform this step with a ***for*** loop that will read the user's values without making any checks. Then, the program must ask the user for a value and search that value in the list (without using the Python operator ***in***). If found, the program will display "Value found in position XX" where XX is the position where it has been found. If the value is not in the list, the program should display the message "Value not found".

Important: The program must not perform any unnecessary operations. If the value is found, you do not need to keep checking if the value is in the remaining positions.

Problem 3 4
Make a program that asks the user for a string and character and display the message "FOUND" if the character is in the string and "NOT FOUND" otherwise.

Problem 35
Make a program that asks 10 numbers (integers) and saves them in a list. Then it will check if the numbers are sorted in ascending order. If the list is sorted, it will display the message "List is sorted", otherwise, it will display “List is NOT sorted".

Important: The program must not perform any unnecessary operations. If any element is not in ascending order, the program should not continue checking the rest of elements.

Problem 36
Make a program to enter real numbers to fill a list of 20 positions. This step will be performed without making any checks. The program will then search for the position where the first value greater than 100 is located. If it finds any, it will display the message "The first element greater than 100 is in position XX" where XX will be the position where it found the element. If no item is greater than 100, the message "No item in the list is greater than 100" will be displayed.

Important: The program must not perform any unnecessary operations. When the first item above 100 is found, there is no need to keep on searching.

Problem 37

Make a program that asks the user to enter real values to fill a list of 10 positions. We will perform this step with a ***for*** loop without making any checks.

The program must then add the values in the list, starting from the first position, until the accumulated value is greater than 25. If this amount has been accumulated, the program will show the message "At position PP the accumulated sum is greater than 25" where PP will be the position in the list where the accumulated sum of 25 has been exceeded. We will display the positions starting from 1 (**note, however, that the indices in the list start at 0**). If the cumulative sum of all the list does not exceed 25, the program will display the message "The cumulative sum of the list is less than or equal to 25".

Important: The program must not perform any unnecessary operations. When the position where the accumulated sum is greater than 25 is found, it is no longer necessary to continue adding the remaining positions.

Problem 38

Make a program that reads integer values and fills two lists of 6 positions (first we will read one list, then, the second one). We will do this step with two loops, without making any checks. Then the program must check whether the values in the two lists are the same. If so, the program will output "EQUAL", otherwise, it will display "DIFFERENT".

Important: The program must not perform any unnecessary operations. Whenever it detects a different element, it will not continue checking the remaining positions.

Problem 39

Make a program that asks the user to enter two strings and show the first character of the first string that is different from the second one with the message "The first different character is CC" where CC will be the first different character. If the two strings are the same, the program should display the message, "There are no different characters"

Example: If we enter the strings "BOOK" and "FREE" the program should say that the first different character is 'B'. If we enter the strings "HELLO" and "HELLO", it must say that there aren't any different characters.

Problem 40

Make a program that reads a string and checks if it is a palindrome (if reading from right to left is equal to reading from left to right). If it is a palindrome, the program will display "The string is palindrome.", otherwise it will show "The string is NOT a palindrome.".

Problem 41

Make a program that asks the user to enter a password and checks if the password is in the correct format. A valid password must follow these conditions:

- Must be at least 8 characters long.
- Must have at least one uppercase and one lowercase character.
- Must have at least two digits (0,1,2, ..., 9).
- Must have at least one of the following symbols: $ & * € _ @ #

If the password format is not valid, it will display "Error: Invalid Password.". However, if the password format is correct, the user should be prompted to enter the password a second time and check that the two input strings are correct. Then, if the two passwords are identical, the program will display the message "Correct Password." Otherwise, it will display "Error: The two passwords are not the same."

Problem 42

Make a program that implements the (simplified) hanging game. The game of hanging consists in discovering a hidden word by guessing letters. When a letter is guessed, it shows in which position it is located. The game ends when the player discovers the word, or when a number of previously agreed-upon attempts, i.e., 10 have been exceeded.

The program will have the predefined hidden word ("python") stored in a variable (it will not be entered by the user). Initially the program will display the message "The hidden word has N letters:" and the next line will display a sequence of N hyphens (N is the number of letters of the hidden word. For example:

```
The hidden word has 6 letters:
------
```

The program will ask the user to enter a letter and the program will display the sequence of hyphens that represents the hidden word, showing the guessed characters in their respective positions. It will also show the number of attempts left. For example:

```
Enter a letter: o
----o- You have 9 attempts left
```

These steps will be repeated until the user discovers the hidden word or until the maximum number of attempts is reached. Important: the number of attempts always decreases, no matter if the letter guessed by the user is already contained in the hidden word or not.

If the user discovers it before all attempts are reached, the program will display the message "Congratulations, you discovered it!!!". Otherwise, it will display the message "Sorry, you lost, the word was: XXXXX" where XXXXX will be the hidden word of the game.

Notes: Consider that the hidden word will be all lowercase and remember that the strings are unalterable (they cannot be modified).