**2. ЗАВДАННЯ**

**2.1. Домашня підготовка до роботи**

1. Скласти блок-схему алгоритму та програму мовою Python згідно з варіантом, отриманим від викладача, виконати відповідне завдання за табл. 1. Відповідний алгоритм оформити у вигляді функції; програма повинна вивести на екран отриману матрицю.

**NB:** під час розроблення програми **НЕ МОЖНА** використовувати вбудовані в Python функції, які б могли вирішити завдання в один прохід. Наприклад, Напишіть функцію для перевертання рядка -> НЕ можна використати вбудовану функцію *reverse* для перевертання рядка. Її потрібно реалізувати самостійно

Таблиця 1

|  |  |
| --- | --- |
| **№ варiанта** | **Задача** |
| **1** | * + - 1. Хрестики-нулики   Реалізуйте функцію, яка перевіряє, чи стрічка має однакову кількість символів ‘x’ i ‘o’. Функція має повертати boolean і бути case insensitive (ігнорувати регістр). Стрічка може містити довільні символи.  Приклади:  XO("ooxx") => true  XO("xooxx") => false  XO("ooxXm") => true  XO("zpzpzpp") => true // якщо немає жодного 'x' чи 'o' повинно повертатись true  XO("zzoo") => false |
| **2** | Ізограми  Ізограма - це слово, у якому немає повторюваних букв, послідовних чи непослідовних. Реалізуйте функцію, яка визначає, чи рядок, що містить лише літери, є ізограмою. Припускайте, що порожній рядок є ізограмою. Ігноруйте регістр літер.  Приклади:  isIsogram "Dermatoglyphics" == true  isIsogram "aba" == false  isIsogram "moOse" == false |
| **3** | Зворотній рядок  Напишіть функцію для перевертання рядка. Вхідний рядок подається як String. Можна припустити, що всі символи складаються з друкованих символів ASCII.  Приклади:  reverseString(“test”) => “tset”  reverseString(“COVID19”) => “91DIVOC”  reverseString(“reverse”) => “esrever” |
| **4** | * + - 1. Знайдіть самотнього   У масиві цілих чисел кожен елемент з’являється двічі, крім одного. Знайдіть його. Алгоритм повинен мати лінійну складність виконання  Приклади:  [2,2,1] => 1  [4,1,2,1,2] => 4 |
| **5** | * + - 1. Знайдіть пропущену букву   Напишіть метод, який приймає послідовність (збільшувальних) літер як вхідні дані і повертає пропущену літеру в послідовності.  Припускайте, що завжди буде відсутня рівно одна буква. Послідовність завжди буде містити літери лише в одному регістрі.  Приклади:  ["a","b","c","d","f"] -> "e"  ["O","Q","R","S"] -> "P" |
| **6** | * + - 1. Кому це подобається?   Ви напевно знаєте систему лайків з Facebook та інших соцмереж. Люди можуть "лайкати" публікації у блозі чи фото. Потрібно створити текст, який повинен відображатися поруч із лайками. Реалізуйте функцію like() -> String, яка повинна приймати вхідний послідовність, що містить імена людей, яким подобається фото. Він повинен повертати відображуваний текст, як показано в прикладах:  likes([]) => "no one likes this"  likes(["Peter"]) => "Peter likes this"  likes(["Jacob", "Alex"]) => "Jacob and Alex like this"  likes(["Max","John","Mark"]) => "Max, John and Mark like this"  likes(["Alex", "Jacob", "Mark", "Max"]) => "Alex, Jacob and 2 others like this"  Для 4 і більше імен кількість у “2 інших” просто збільшується.  \* return має бути виділеним рядком  \* не змінювати введення |
| **7** | Знайдіть перше непослідовне число  Ваше завдання - знайти перший елемент послідовності, який не є послідовним.  Під не послідовним мається на увазі не точно на 1 більший за попередній елемент послідовності. Якщо весь масив є послідовним, поверніть значення null.  Послідовність завжди матиме принаймні 2 елементи, і всі елементи будуть числами. Усі номери також будуть унікальними та у порядку зростання.  Приклад:  [1,2,3,4,6,7,8] => 6 |
| **8** | Чи є рядок капсом?  Створіть метод is\_uppercase (), що перевіряє, чи є рядок ALL CAPS. Наприклад:  is\_uppercase ("c") == false  is\_uppercase ("C") == true  is\_uppercase ("") == true  is\_uppercase("hello I AM DONALD") == false  is\_uppercase("HELLO I AM DONALD") == true  is\_uppercase("ACSKLDFJSgSKLDFJSKLDFJ") == false  is\_uppercase("ACSKLDFJSGSKLDFJSKLDFJ") == true |
| **9** | Видаліть пробіли  Видаліть пробіли з рядка, а потім поверніть отриманий рядок. |
| **10** | Чи впорядковані цифри?  Визначити, чи вхідна послідовність є у порядку зростання. Кажуть, що послідовність знаходиться у порядку зростання, якщо немає двох сусідніх цілих чисел, де ліве ціле число перевищує ціле праве число.  Приклади:  in\_asc\_order({1,2,4,7,19}, 5); // true  in\_asc\_order({1,2,3,4,5}, 5); // true  in\_asc\_order({1,6,10,18,2,4,20}, 7); // false  in\_asc\_order({9,8,7,6,5,4,3,2,1}, 9); // false |
| **11** | * + - 1. Допоможіть Бобу порахувати літери та цифри   Боб - ледачий чоловік.  Створіть для нього метод, за допомогою якого можна визначити, скільки букв і цифр знаходиться в даному рядку.  Приклади:  "hel2!lo" --> 6  "wicked .. !" --> 6  "!?..A" --> 1 |
| **12** | * + - 1. Кількість десяткових цифр   Визначте загальну кількість цифр у цілому числі (n> = 0), поданому як вхід для функції. Наприклад, 9 - це одна цифра, 66 має 2 цифри, а 128685 - 6 цифр. |
| **13** | * + - 1. Індексна капіталізація   Маючи рядок і послідовність цілих чисел, що представляють індекси, замініть літери, що стоять на цих індексах на великі літери.  Приклади:  capitalize("abcdef",[1,2,5]) = "aBCdeF"  capitalize("abcdef",[1,2,5,100]) = "aBCdeF". |
| **14** | * + - 1. Парність елемента послідовності   Функція отримує на вхід послідовність цілих чисел, елементи яких мають як негативне, так і позитивне значення, за винятком одного цілого числа, яке є лише негативним або лише позитивним. Вашим завданням буде знайти це ціле число.  Приклади:  [1, -1, 2, -2, 3] => 3  [-3, 1, 2, 3, -1, -4, -2] => -4  [1, -1, 2, -2, 3, 3] => 3 (шукане число може повторюватись) |
| **15** | * + - 1. Сума двох мінімальних натуральних чисел   Створіть функцію, яка повертає суму двох мінімальних додатних чисел із заданої послідовності із мінімум 4 позитивних цілих чисел.  Приклади:  [19, 5, 42, 2, 77] => 7.  [10, 343445353, 3453445, 3453545353453] => 3453455. |
| **16** | * + - 1. Що домінує у вашій послідовності?   Домінатором послідовності називається значення, яке зустрічається більш ніж у половині елементів послідовності. Напишіть функцію знаходження домінатора. Функція повинна повертати -1, якщо послідовність не має домінатора. Усі значення в arr приймайте >= 0.  Приклад: [3,4,3,2,3,1,3,3] - Домінатором arr є 3, оскільки він зустрічається в 5 із 8 елементів arr, а 5 - більше половини 8. |
| **17** | * + - 1. Максимальний добуток   Маючи послідовність цілих чисел, знайдіть максимальний добуток, отриманий від множення 2 сусідніх чисел у масиві.  Приклади:  adjacentElementsProduct([1, 2, 3]); => 6  adjacentElementsProduct([9, 5, 10, 2, 24, -1, -48]); => 50  adjacentElementsProduct([-23, 4, -5, 99, -27, 329, -2, 7, -921])  =>  -14 |
| **18** | * + - 1. Не давай мені п’ять!   Напишіть функцію, що отримує початковий і кінцевий номер діапазону і повинна повернути кількість усіх чисел, крім тих, що містять 5. Початкове та кінцеве число - включно!  Приклади:  1,9 -> 1,2,3,4,6,7,8,9 => 8  4,17 -> 4,6,7,8,9,10,11,12,13,14,16,17 => 12 |
| **19** | * + - 1. Середнє проти медіани   Реалізувати метод meanVsMedian, який приймає послідовність цілих чисел непарної довжини і повертає одне з наступного:  'mean' - у випадку, якщо середнє значення більше медіани.  'median' - у випадку, якщо медіана послідовності більша за середнє значення.  'same'  - у випадку, якщо середнє значення та медіана мають однакове значення. |
| **20** | * + - 1. Збалансована послідовність   Маючи послідовність парного розміру N, знайдіть мінімальне значення, яке можна додати до елемента, щоб послідовність стала збалансованою. Послідовність є збалансованою, якщо сума лівої половини елементів послідовності дорівнює сумі правої половини. |
| **21** | * + - 1. Третій за величиною елемент   Дано послідовність різних елементів. Знайдіть у ньому третій за величиною елемент.  Приклади  {2,4,1,3,5} => 3  {10,2} = > -1 |
| **22** | * + - 1. Порахуйте можливі трикутники   Дано несортовану послідовність натуральних чисел. Знайдіть кількість трикутників, які можна сформувати з трьома різними елементами масиву як довжини трьох сторін трикутників. |
| **23** | * + - 1. Перетин двох послідовностей   Враховуючи дві послідовності A і B відповідно розміру N і M, надрукувати кількість елементів у перетині (або загальних елементів) двох послідовностей. Перетин двох послідовностей можна визначити як набір, що містить різні спільні елементи між двома послідовностями |
| **24** | * + 1. Peppa Pig   Свинка Пеппа вводить стрічку S. Після того вибирає ключ K - цифру від 1 до 10. Вона формує шифр С з кожного K-того введеного символу, поки не залишиться жодного. Довжина шифру С рівна довжині стрічки. Який шифр сформувала Пеппа?  Вхідні дані:  У першому рядку довільна стрічка S.  У другому рядку ключ K .  Вихідні дані: Шифр С  Приклад:   |  |  | | --- | --- | | Peppa Pig  3 | p gpiaePP | |
| **25** | * + 1. Щасливе число   Користувач вводить число N. Після того вибирає ключ K - цифру від 1 до 10. Програма формує нове число M з кожного K-того символу. Число вважається щасливим, якщо воно кратне 3. Яке число M? Чи воно щасливе? Виведіть “YES”,“NO” відповідно.  Вхідні дані:  У першому рядку довільне число N.  У другому рядку  ключ K .  Вихідні дані:  Число M  Приклад:   |  |  | | --- | --- | | 12**3456**7**89**  3 | 369485271  NO | |
| **26** | * + 1. Шифрування   Користувач ввів кросворд C певного розміру. У кожному рядку C - одне слово. Вивести зашифроване слово W, що є найдовшим словом по висоті.  Вхідні дані:  Кросворд C.  Вихідні дані:  Зашифроване слово W  Приклад:   |  |  | | --- | --- | | cat  salad  kiwi | ask | |
| **27** | Різниця послідовностей  Реалізувати функцію різниці послідовностей, яка віднімає одину послідовність з іншого і повертає результат.  Потрібно видалити всі значення зі списку a, які є у списку b, зберігаючи їх порядок. Якщо значення є в b, усі його входження повинні бути видалені з іншого |
| **28** | Рівні сторони послідовності  З послідовності цілих чисел знайти індекс N, де сума цілих чисел ліворуч від елемента з індексом N дорівнює сумі цілих чисел праворуч від елемента з індексом N. Якщо немає індексу, який би підійшов, повернути - 1.  Наприклад:  Масив {1,2,3,4,3,2,1}:  Функція поверне індекс 3, оскільки на третьому місці послідовності сума лівої частини індексу ({1,2,3}) та суми правої частини індексу ({3,2,1}) обидві рівні 6. |

**2.2. Робота в лабораторії**

1. Ввести в комп'ютер програму, написану мовою Python згідно з отриманим завданням.
2. Відлагодити програму. При необхідності скоригувати блок-схему алгоритму та програму у відповідності з виявленими логічними та синтаксичними помилками.
3. Остаточні версії блок-схеми, програми та отримані результати занести у звіт з лабораторної роботи.
4. Здати звіт з лабораторної роботи.

**Питання для самоконтролю:**

1. Що таке список (list) в Python та як визначити пустий список?
2. Як додати елемент до списку в Python?
3. Як видалити елемент із списку в Python?
4. Як відобразити певний елемент списку за індексом?
5. Як відобразити частину списку (зріз) в Python?
6. Що таке кортеж (tuple) в Python і як він відрізняється від списку?
7. Як визначити словник (dictionary) в Python і як він працює?
8. Як додавати та видаляти пари ключ-значення в словнику?
9. Як отримати значення за ключем у словнику?
10. Що таке множина (set) в Python і як вона працює?
11. Як додавати та видаляти елементи у множині?
12. Як перевірити наявність елементу в множині?
13. Як визначити функцію (function) в Python?
14. Як передавати аргументи у функцію?
15. Як повернути значення з функції?
16. Що таке анонімні функції (lambda functions) в Python?
17. Як викликати вбудовані функції, такі як len, sum, і т. д.?
18. Як створити та використовувати генератори списків (list comprehensions)?