**Звіт про виконання практичних завдань до лекцій з курсу Технології програмування**

Звіт до Теми №0

Функції та змінні

Під час виконання практичного завдання до Теми №0 було надано варіанти рішення до наступних задач:

**Перетворення рядка**

Необхідно рядок, що має вигляд "abcdefg123" перетворити наступним чином "321gfedcba", вважаючи сталою довжину рядку в 10 символів.

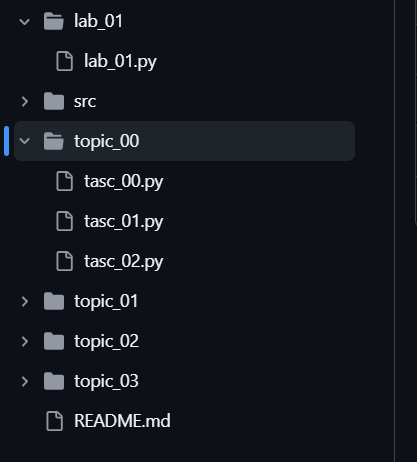
**Хід виконання завдання**

Текст програми:

|  |
| --- |
|  |

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman>

Знімок екрану з посилання на github:



**Тестування функцій strip, capitalize, title, upper, lower**

Необхідно виконати тестування функцій strip, capitalize, title, upper, lower та показати, що робить кожна з функцій.

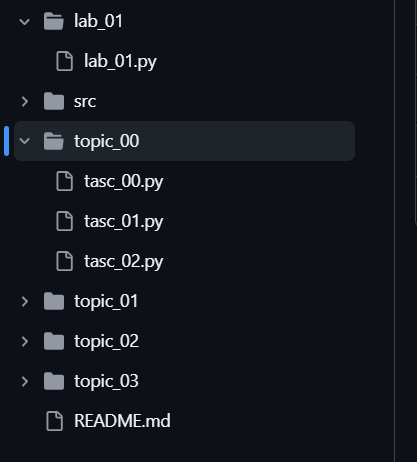
**Хід виконання завдання**

Текст програми:

|  |
| --- |
|  |

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman/blob/main/topic_00/tasc_01.py>

Знімок екрану з посилання на github:



**Написати функцію, яка буде рахувати дискримінант**

Необхідно написати функцію, яка буде рахувати дискримінант з чисел, які будуть введені користувачем

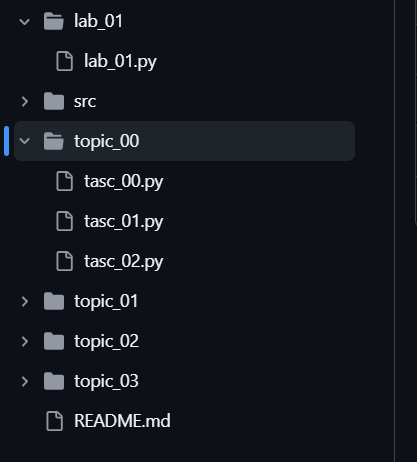
**Хід виконання завдання**

|  |
| --- |
|  |

Текст програми:

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman/blob/main/topic_00/tasc_01.py>

Знімок екрану з посилання на github:



Звіт до Теми №1

Умовний перехід

Під час виконання практичного завдання до Теми №1 було надано варіанти рішення до наступних задач:

**Написати функцію пошуку коренів квадратного рівняння**

Необхідно написати функцію пошуку коренів квадратного рівняння використовуючи функцію розрахунку дискримінанту та умовні переходи.

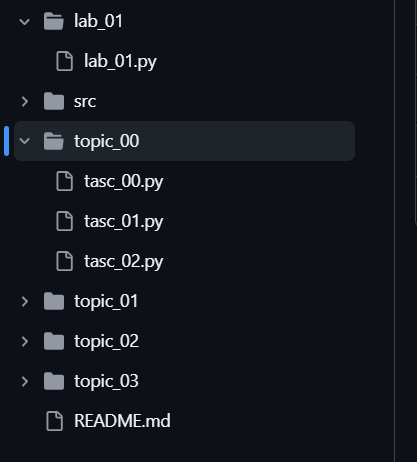
**Хід виконання завдання**

Текст програми:

|  |
| --- |
| import math  # Функція для обчислення дискримінанту  def discriminant(a, b, c):      return b\*\*2 - 4\*a\*c  # Функція для пошуку коренів квадратного рівняння  def quadratic\_roots(a, b, c):      d = discriminant(a, b, c)      if d > 0:          x1 = (-b + math.sqrt(d)) / (2\*a)          x2 = (-b - math.sqrt(d)) / (2\*a)          print(f"Два дійсних корені: x₁ = {x1}, x₂ = {x2}")      elif d == 0:          x = -b / (2\*a)          print(f"Один дійсний корінь: x = {x}")      else:          print("Дійсних коренів немає")  # Приклад виклику  a = float(input("Введіть a: "))  b = float(input("Введіть b: "))  c = float(input("Введіть c: "))  quadratic\_roots(a, b, c) |

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman/blob/main/topic_00/tasc_01.py>

Знімок екрану з посилання на github:



**Написати програму калькулятор використовуючи if else конструкцію**

Написати програму калькулятор використовуючи if else конструкцію. Кожна операція має бути виконана в окремій функції.

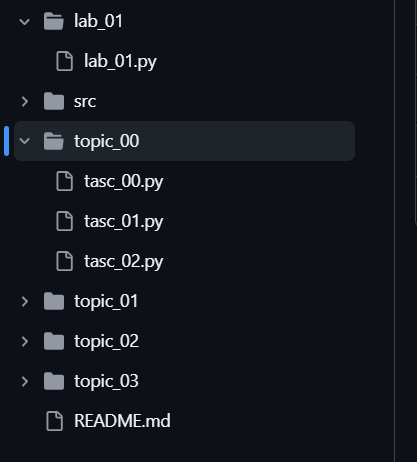
**Хід виконання завдання**

Текст програми:

|  |
| --- |
| # Функції для кожної операції  def add(x, y):      return x + y  def subtract(x, y):      return x - y  def multiply(x, y):      return x \* y  def divide(x, y):      if y != 0:          return x / y      else:          return "Ділення на нуль неможливе!"  # Основна програма  print("Оберіть операцію: +, -, \*, /")  op = input("Введіть операцію: ")  a = float(input("Введіть перше число: "))  b = float(input("Введіть друге число: "))  if op == '+':      print("Результат:", add(a, b))  elif op == '-':      print("Результат:", subtract(a, b))  elif op == '\*':      print("Результат:", multiply(a, b))  elif op == '/':      print("Результат:", divide(a, b))  else:      print("Невідома операція!") |

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman/blob/main/topic_00/tasc_01.py>

Знімок екрану з посилання на github:



**Написати програму калькулятор використовуючи match конструкцію**

Написати програму калькулятор використовуючи match конструкцію. Кожна операція має бути виконана в окремій функції.

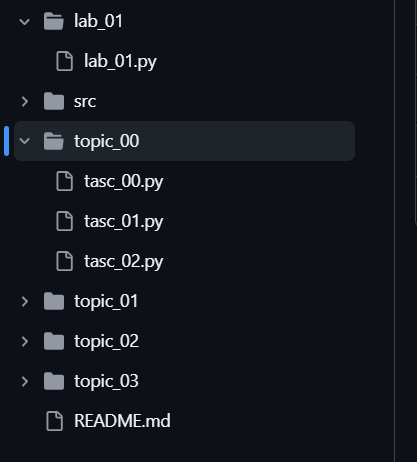
**Хід виконання завдання**

Текст програми:

|  |
| --- |
| # Функції для кожної операції  def add(x, y):      return x + y  def subtract(x, y):      return x - y  def multiply(x, y):      return x \* y  def divide(x, y):      if y != 0:          return x / y      else:          return "Ділення на нуль неможливе!"  # Основна програма  print("Оберіть операцію: +, -, \*, /")  op = input("Введіть операцію: ")  a = float(input("Введіть перше число: "))  b = float(input("Введіть друге число: "))  match op:      case '+':          print("Результат:", add(a, b))      case '-':          print("Результат:", subtract(a, b))      case '\*':          print("Результат:", multiply(a, b))      case '/':          print("Результат:", divide(a, b))      case \_:          print("Невідома операція!") |

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman/blob/main/topic_00/tasc_01.py>

Знімок екрану з посилання на github:



Звіт до Теми №2

Цикли

Під час виконання практичного завдання до Теми №2 було надано варіанти рішення до наступних задач:

**Написати програму калькулятор з постійними запитами на введення нових даних та операцій**

Написати програму калькулятор з постійними запитами на введення нових даних та операцій. За основу взяти програму калькулятор з попередньої теми. Реалізувати механізм завершення програми після отримання відповідної команди.

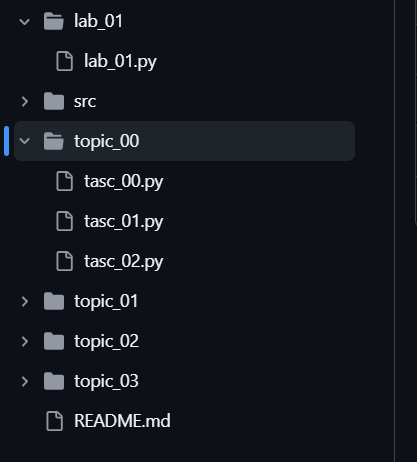
**Хід виконання завдання**

Текст програми:

|  |
| --- |
| def add(a, b): return a + b  def sub(a, b): return a - b  def mul(a, b): return a \* b  def div(a, b): return a / b if b != 0 else "Ділення на нуль!"  while True:      print("\nОберіть операцію (+, -, \*, /) або введіть 'exit' для виходу")      op = input("Операція: ")      if op.lower() == 'exit':          print("Програма завершена.")          break      a = float(input("Перше число: "))      b = float(input("Друге число: "))      if op == '+':          print("Результат:", add(a, b))      elif op == '-':          print("Результат:", sub(a, b))      elif op == '\*':          print("Результат:", mul(a, b))      elif op == '/':          print("Результат:", div(a, b))      else:          print("Невідома операція!") |

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman/blob/main/topic_00/tasc_01.py>

Знімок екрану з посилання на github:



**Написати програму тестування функцій списків**

Написати програму тестування функцій списків таких як: extend(), append(), insert(id, val), remove(val), clear(), sort(), reverse(), copy().

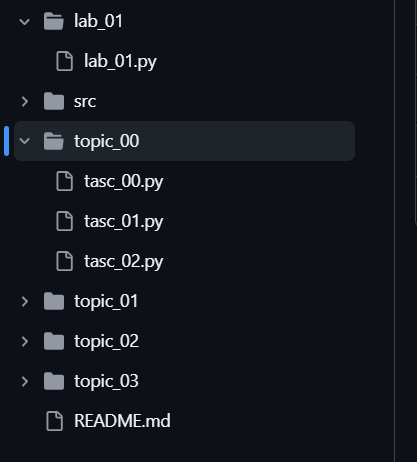
**Хід виконання завдання**

Текст програми:

|  |
| --- |
| # Створюємо список  nums = [1, 2, 3]  print("Початковий список:", nums)  # append()  nums.append(4)  print("Після append(4):", nums)  # extend()  nums.extend([5, 6])  print("Після extend([5, 6]):", nums)  # insert()  nums.insert(2, 99)  print("Після insert(2, 99):", nums)  # remove()  nums.remove(99)  print("Після remove(99):", nums)  # sort()  nums.sort()  print("Після sort():", nums)  # reverse()  nums.reverse()  print("Після reverse():", nums)  # copy()  nums\_copy = nums.copy()  print("Копія списку:", nums\_copy)  # clear()  nums.clear()  print("Після clear():", nums) |

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman/blob/main/topic_00/tasc_01.py>

Знімок екрану з посилання на github:



**Написати програму тестування функцій словників**

Написати програму тестування функцій словників таких як: update(), del(), clear(), keys(), values(), items()

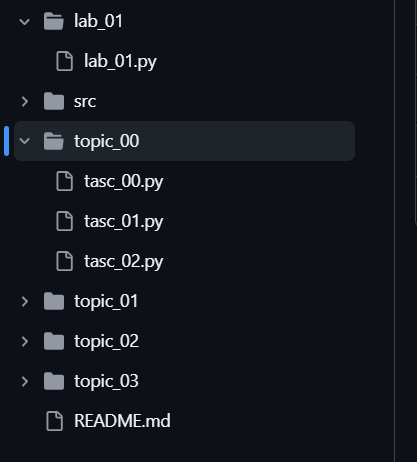
**Хід виконання завдання**

Текст програми:

|  |
| --- |
| # Створюємо словник  student = {"name": "Roman", "group": "KB-242", "age": 19}  print("Початковий словник:", student)  # update()  student.update({"age": 20, "city": "Kyiv"})  print("Після update():", student)  # keys()  print("Ключі:", list(student.keys()))  # values()  print("Значення:", list(student.values()))  # items()  print("Пари ключ-значення:", list(student.items()))  # del  del student["city"]  print("Після del student['city']:", student)  # clear()  student.clear()  print("Після clear():", student) |

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman/blob/main/topic_00/tasc_01.py>

Знімок екрану з посилання на github:



**Маючи відсортований список, написати функцію пошуку позиції для вставки нового елементу в список.**

Маючи відсортований список, написати функцію пошуку позиції для вставки нового елементу в список.

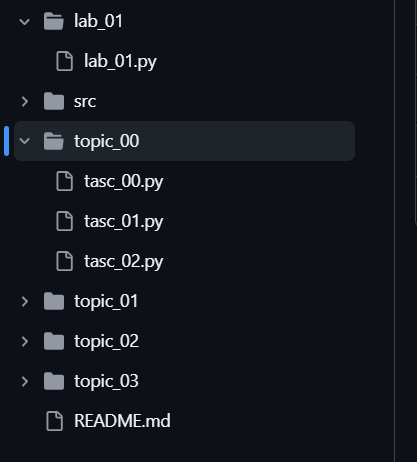
**Хід виконання завдання**

Текст програми:

|  |
| --- |
| def find\_insert\_position(sorted\_list, new\_value):      for i in range(len(sorted\_list)):          if new\_value < sorted\_list[i]:              return i      return len(sorted\_list)  # якщо новий елемент більший за всі — вставляємо в кінець  # Приклад використання:  numbers = [1, 3, 5, 7, 9]  print("Вихідний список:", numbers)  value = int(input("Введіть число для вставки: "))  pos = find\_insert\_position(numbers, value)  print(f"Елемент {value} потрібно вставити на позицію {pos}")  # Вставимо елемент для перевірки:  numbers.insert(pos, value)  print("Новий список:", numbers) |

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman/blob/main/topic_00/tasc_01.py>

Знімок екрану з посилання на github:



Звіт до Теми №3

Виняткові ситуації

Під час виконання практичного завдання до Теми №4 було надано варіанти рішення до наступних задач:

**Розширити програму калькулятор функцією запитів даних, що обробляє виняткові ситуації введення некоректних даних для математичних операцій.**

Необхідно розширити програму калькулятор функцією запитів даних для виконання операцій від користувача, що обробляє виняткові ситуації введення некоректних даних для математичних операцій.

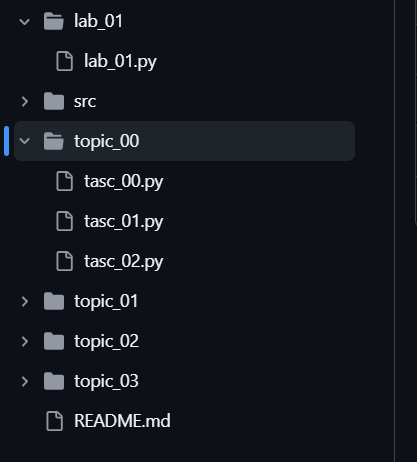
**Хід виконання завдання**

Текст програми:

|  |
| --- |
| # --- Функції операцій ---  def add(a, b):      return a + b  def sub(a, b):      return a - b  def mul(a, b):      return a \* b  def div(a, b):      try:  #  обертаємо рядок ділення          return a / b      except ZeroDivisionError:          return " Помилка: ділення на нуль!"  # --- Основна програма ---  while True:      print("\nОберіть операцію: +, -, \*, / або 'exit' для виходу")      op = input("Операція: ")      if op.lower() == "exit":          print(" Програма завершена.")          break      try:  #  обертаємо рядки, де вводяться числа          a = float(input("Введіть перше число: "))          b = float(input("Введіть друге число: "))      except ValueError:  # якщо введено не число          print(" Помилка: введено не число!")          continue  # повертаємось на початок циклу      # --- Виконуємо операцію ---      if op == '+':          print("Результат:", add(a, b))      elif op == '-':          print("Результат:", sub(a, b))      elif op == '\*':          print("Результат:", mul(a, b))      elif op == '/':          print("Результат:", div(a, b))      else:          print(" Невідома операція!") |

Посилання на github: <https://github.com/TP-KB-242-Kobzar-Roman/TP-KB-242-Kobzar-Roman/blob/main/topic_00/tasc_01.py>

Знімок екрану з посилання на github:



**Розширити функцію ділення обробкою виняткової ситуації ділення на нуль.**

Необхідно розширити функцію ділення обробкою виняткової ситуації ділення на нуль.