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

Звіт до Теми №1

Тема: Функції та змінні

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

1. Необхідно обернути рядок “abaufdksl” в зворотному порядку.

Хід виконання завдання:

Спочатку оголошуємо рядок/слово, який буде обернений. Використовуємо зворотній зріз, і виводимо вже готовий перевернутий рядок/слово.

Текст програми:

|  |
| --- |
| text="abaufdksl"  reversed=text[::-1]  print(reversed) |

2. Далі треба виконати тестування функцій, що працюють з рядками: strip(), capitalize(), title(), upper(), lower().

Хід виконання завдання:

Задаємо вхідний рядок. Далі пишемо операції для тестування функцій з обробки рядків. І виводимо результат на екран.

Текст програми:

|  |
| --- |
| text="HeLLo woRld!"  strip\_text=text.strip()  print(strip\_text)  capitalize\_text=text.capitalize()  print(capitalize\_text)  title\_text=text.title()  print(title\_text)  upper\_text=text.upper()  print(upper\_text)  lower\_text=text.lower()  print(lower\_text) |

3. В останньому завданні потрібно написати функцію пошуку дискримінанту квадратного рівняння.

Хід виконання завдання:

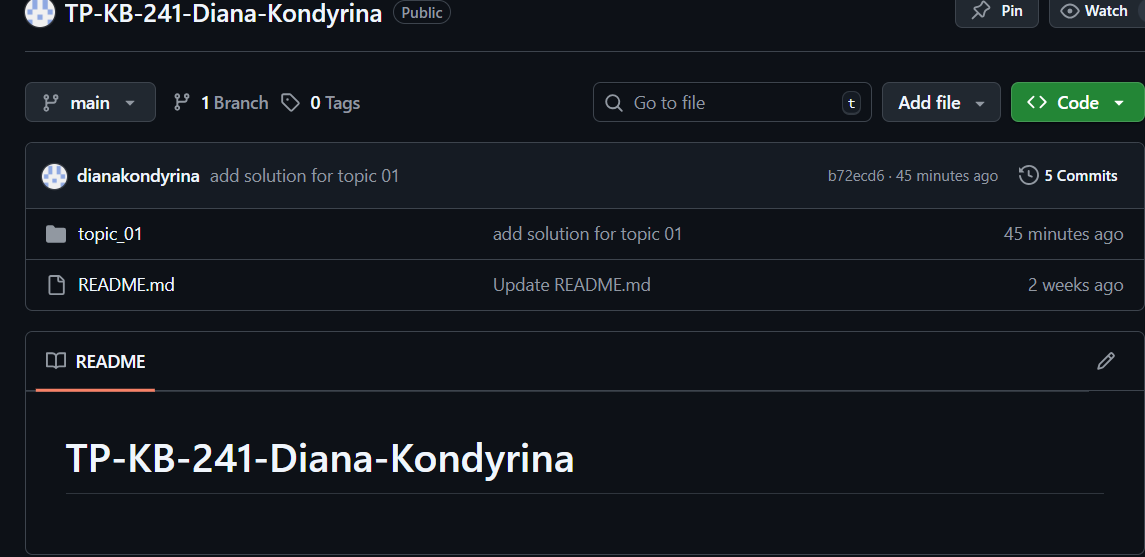
Оголошуємо функцію diskriminant, яка приймає три параметри, обчислюємо сам дискримінант, у програмі задаємо конкретні значення коефіцієнтів a, b, c. Викликаємо функцію diskriminant і зберігаємо результат у змінній D і виводимо отримане значення дискримінанта.

Текст програми:

|  |
| --- |
| def diskriminant(a,b,c):      return b\*\*2-4\*a\*c  a=3  b=21  c=32  D=diskriminant(a,b,c)  print (f"Дискримінант рівняння={D}") |

Посилання на github:https://github.com/dianakondyrina/TP-KB-241-Diana-Kondyrina

Знімок екрану з посилання на github:



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

Звіт до Теми №2

Тема: Умовний перехід

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

1.Написати функцію пошуку коренів квадратного рівняння використовуючи функцію розрахунку дискримінанту з попередньої теми та умовні переходи.

Хід виконання завдання:

Викликаємо функцію diskriminant, щоб знайти D. Далі записуємо умови: Якщо D<0: рівняння коренів немає. Якщо D>0: обчислюємо два різні корені. Якщо D=0: обчислюємо один корінь. Виводимо значення.

Текст програми:

|  |
| --- |
| from math import sqrt  def diskriminant(a,b,c):       return b\*\*2-4\*a\*c  def roots(a,b,c):       D=diskriminant(a,b,c)       if D<0:            return ("Рівняння немає жодного кореня")       elif D==0:            x=-b/(2\*a)            return ("рівняння має корінь: x={x}")       else:        x1=(-b-sqrt(D))/(2\*a)        x2=(-b+sqrt(D))/(2\*a)       return ("Рівняння має корені: x1={x1}, x2={x2}")  a=-12  b=3  c=23  print (roots (a,b,c)) |

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

Хід виконання завдання:

Текст програми:

|  |
| --- |
| a = float(input("Напишіть перше число: "))  b = float(input("Напишіть друге число: "))  operation = input("Оберіть операцію: ")  if operation == "+":       print(f"Результат: {a + b}")  elif operation == "-":      print(f"Результат: {a - b}")  elif operation == "\*":      print(f"Результат: {a \* b}")  else:      if b == 0:                print("на 0 не ділять")        else:                  print(f"Результат: {a / b}") |

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