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

Звіт до Теми №1

Функції та змінні

Під час виконання практичного завдання до Теми №1 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:

* 1. Спочатку створила змінну string та задала їй значення вказане в завданні, а саме "abcdefg123".
  2. Далі використала функцію print, що виводить на екран значення вказане в дужках. В дужках виконала string[::-1], що бере значення змінної string та перевертає його в зворотньому напрямку. Синтаксиз виразу [::-1]: [start:stop:step], де start - індекс, з якого починається сріз, stop - індекс, на якому закінчується сріз, step - крок, на який здійснюється переміщення від start до stop. У нашому випадку start і stop не вказані, тобто ми беремо всю послідовність, step встановлено в -1, тобто ми переміщуємося в зворотному напрямку.

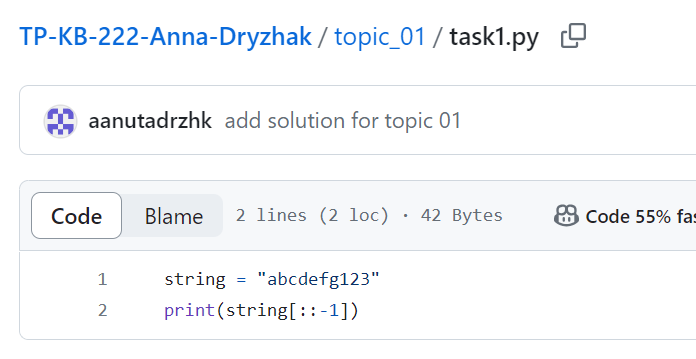
Текст програми:

|  |
| --- |
| string = "abcdefg123"  print(string[::-1]) |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_01/task1.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_01/task1.py)

Знімок екрану з посилання на github:



**Виконання функцій**

Необхідно виконати такі функції як: strip, capitalize, title, upper, lower та інші.

Хід виконання завдання:

Створила 6 однакових макетів коду для виконання 6 різних функцій. В макеті спочатку створено змінну pib значення якої дорівнює моєму ПІБ. За допомогою фунції print(pib) вивожу на екран своє ПІБ. Далі сворюю змінну testpib, що дорівнює значенню змінної pib з використанням певної функції для роботи з рядками. І наступною функцією print(testpib) я вивожу на екран значення змінної testpib. В кінці також додала print(“ “), щоб між виконанням різних функцій була відстань.

Значення 6 різних функцій, що я використала в завданні №2:

* + strip() - видаляє пробіли з початку та кінця рядка
  + сapitalize() - перша літера рядка стає великою
  + title() - перша літера кожного слова стає великою
  + upper() - всі літери у рядку стають великими
  + lower() - всі літери у рядку стають маленькими
  + swapcase() - всі великі літери у рядку стають маленькими, а всі маленькі - великими

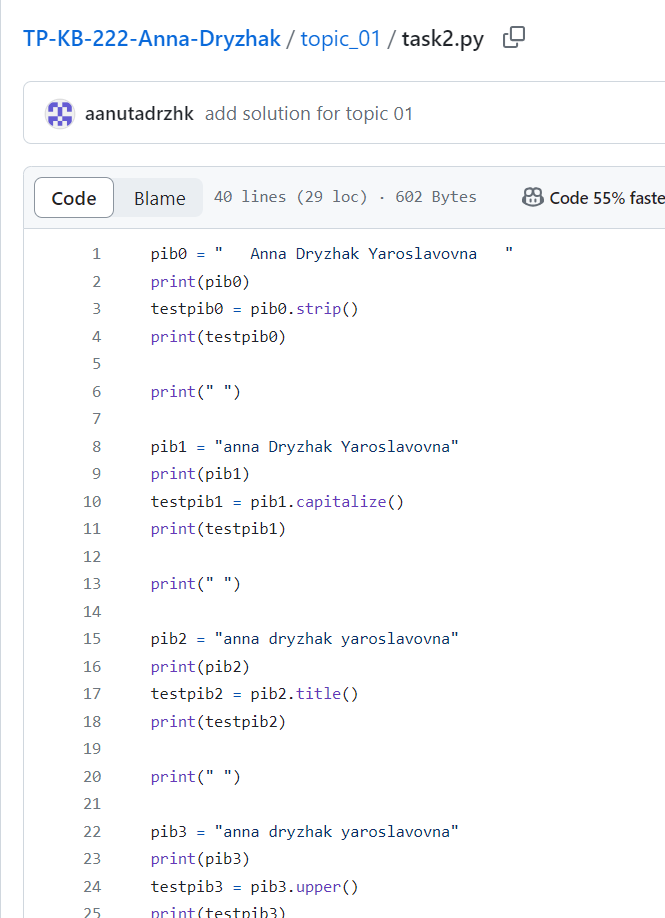
Текст програми:

|  |
| --- |
| pib0 = " Anna Dryzhak Yaroslavovna "  print(pib0)  testpib0 = pib0.strip()  print(testpib0)    print(" ")    pib1 = "anna Dryzhak Yaroslavovna"  print(pib1)  testpib1 = pib1.capitalize()  print(testpib1)    print(" ")    pib2 = "anna dryzhak yaroslavovna"  print(pib2)  testpib2 = pib2.title()  print(testpib2)    print(" ")    pib3 = "anna dryzhak yaroslavovna"  print(pib3)  testpib3 = pib3.upper()  print(testpib3)    print(" ")    pib4 = "ANNA DRYZHAK YAROSLAVOVNA"  print(pib4)  testpib4 = pib4.lower()  print(testpib4)    print(" ")    pib5 = "Anna Dryzhak Yaroslavovna"  print(pib5)  testpib5 = pib5.swapcase()  print(testpib5) |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_01/task2.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_01/task2.py)

Знімок екрану з посилання на github:



**Пошук дискримінанту квадратного рівняння**

Необхідно написати код, що буде шукати дискримінант квадратного рівняння, приймаючи значення a, b, c у користувача.

Хід виконання завдання:

* 1. За допомогою функції input("Please enter value a/b/c: "), користувач вводить значення для змінних a, b, c і ці значення конвертуються у цілі числа за допомогою int().
  2. Створюємо функцію findD(a, b ,c). Ця функція приймає значення a, b і c, що ввів користувач.
  3. В змінній result за допомогою формули дискримінанта b^2 – 4ac він вираховується.
  4. За допомогою слова return значення дискримінанта повертається з функції.
  5. Під кінець за допомогою фунції print(findD(a, b ,c)) результат пошуку дискримінанту виводиться на екран.

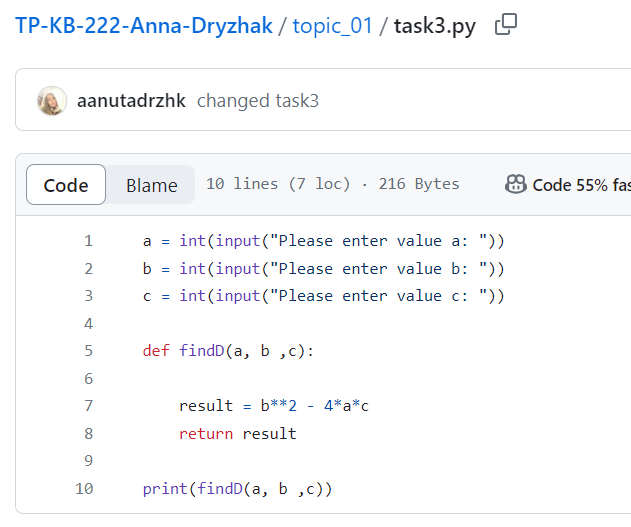
Текст програми:

|  |
| --- |
| a = int(input("Please enter value a: "))  b = int(input("Please enter value b: "))  c = int(input("Please enter value c: "))    def findD(a, b ,c):    result = b\*\*2 - 4\*a\*c  return result    print(findD(a, b ,c)) |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_01/task3.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_01/task3.py)

Знімок екрану з посилання на github:



Звіт до Теми №2

Умовний перехід

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

**Пошук коренів квадратного рівняння**

Необхідно написати код пошуку коренів квадратного рівняння, враховуючи значення дискримінанту.

Хід виконання завдання:

* 1. Спочатку за допомогою функції input запрошуємо у користувача ввести значення a, b, c і переводимо ці значення в числа за допомогою int.
  2. Далі як в завданні з минулої теми робимо функцію для пошуку дискримінанту.
  3. Функція FindRoots(a, b, c) викликає findD(a, b, c) для обчислення дискримінанту. Потім вона перевіряє дискримінант:
     + Якщо D більше 0, то рівняння має два корені, і функція обчислює їх за формулами та повертає значення x1, x2 .
     + Якщо D дорівнює 0, то рівняння має один корінь, і функція повертає значення x1.
     + У всіх інших випадках рівняння не має коренів, і функція повертає None, тобто нічого.
  4. Результат виконання функції FindRoots(a, b, c) зберігається в змінній roots.
  5. Далі відбувається перевірка, чи має рівняння корені:
     + Якщо roots не є None, далі перевіряємо якщо roots є кортежем, то виводяться два корені x1 і x2. Якщо roots не є кортежем, то виводиться один корінь x1.
     + Якщо roots дорівнює None, то виводиться повідомлення, що рівняння не має коренів.

Текст програми:

|  |
| --- |
| a = int(input("Будь ласка, введіть значення a: "))  b = int(input("Будь ласка, введіть значення b: "))  c = int(input("Будь ласка, введіть значення c: "))    def findD(a, b ,c):  D = b\*\*2 - 4\*a\*c  return D      def FindRoots(a, b, c):  D = findD(a, b, c)  if D > 0:  x1 = (-b + D\*\*0.5) / 2\*a  x2 = (-b - D\*\*0.5) / 2\*a  return x1, x2  elif D == 0:  x1 = -b / 2\*a  return x1  else:  return None    roots = FindRoots(a, b, c)  if roots is not None:  if isinstance(roots, tuple):  x1, x2 = roots  print("Рівняння має два корені:")  print("x1 =", x1)  print("x2 =", x2)  else:  x1 = roots  print("Рівняння має один корінь:")  print("x1 =", x1)  else:  print("Рівняння не має коренів.") |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_02/task1.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_02/task1.py)

Знімок екрану з посилання на github:



**Калькулятор**

Необхідно написати код для обчислення операції додавання, віднімання, множення та ділення.

Хід виконання завдання:

* 1. Спочатку за допомогою функції input запрошуємо у користувача ввести значення першого числа, другого та операцію, яку треба зробити з ними і переводимо перші два значення в числа за допомогою int.
  2. Далі створюємо 4 аналогічні функції додавання, віднімання, множення та ділення. У функцію ділення додаємо умову, якщо число b дорівнює 0, то виводимо текст, що ділення на 0 не можливе. В інших випадках виконуємо операцію ділення.
  3. Якщо операція, що ввів користувач дорівнює “+”, тобто додавання, то в змінній result виконуємо створену нами функцію addition(a,b). Аналогічно робимо з 3 іншими значеннями (-,\*,/). А якщо користувач ввів якесь інше значення, то в змінну result записуємо, що операція введена неправильно.
  4. За допомогою функції print виводимо значення змінної result на екран.

Текст програми:

|  |
| --- |
| a = int(input("Введіть перше число: "))  b = int(input("Введіть друге число: "))  op = (input("Операція, яку треба виконати: "))    def addition(a,b):  return a + b  def subtraction(a,b):  return a - b  def multiplication(a,b):  return a \* b  def division(a,b):  if b == 0:  return "Ділення на 0 не можливе"  return a / b    if op == "+":  result = addition(a,b)  elif op == "-":  result = subtraction(a,b)  elif op == "\*":  result = multiplication(a,b)  elif op == "/":  result = division(a,b)  else:  result = "Неправильно введена операція, спробуйте \*,/,-,+"    print(result) |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_02/task2.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_02/task2.py)

Знімок екрану з посилання на github:



**Калькулятор 2**

Необхідно написати код для обчислення операції додавання, віднімання, множення та ділення з використанням match.

Хід виконання завдання:

* 1. Частина коду до виконання функції match аналогічна до минулого завдання.
  2. Далі виконуємо функцію match над значенням змінної op:
     + Якщо користувач ввів +, то виводимо на екран результат функції addition(a,b).
     + Якщо користувач ввів -, то виводимо на екран результат функції subtraction(a,b).
     + Якщо користувач ввів \*, то виводимо на екран результат функції multiplication(a,b).
     + Якщо користувач ввів /, то виводимо на екран результат функції division(a,b).
     + А якщо користувач ввів якесь інше значення, то виводимо на екран напис, що операція введена неправильно.

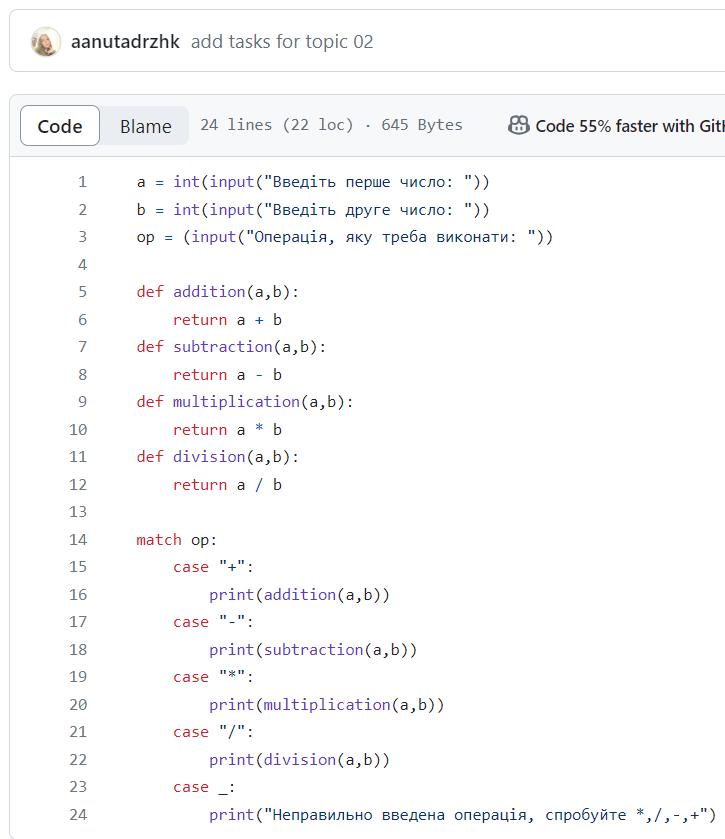
Текст програми:

|  |
| --- |
| a = int(input("Введіть перше число: "))  b = int(input("Введіть друге число: "))  op = (input("Операція, яку треба виконати: "))    def addition(a,b):  return a + b  def subtraction(a,b):  return a - b  def multiplication(a,b):  return a \* b  def division(a,b):  if b == 0:  return "Ділення на 0 не можливе"  return a / b  match op:  case "+":  print(addition(a,b))  case "-":  print(subtraction(a,b))  case "\*":  print(multiplication(a,b))  case "/":  print(division(a,b))  case \_:  print("Неправильно введена операція, спробуйте \*,/,-,+") |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_02/task3.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_02/task3.py)

Знімок екрану з посилання на github:



Звіт до Теми №3

Цикли

Під час виконання практичного завдання до Теми №3 було надано варіанти рішення до наступних задач:

**Калькулятор 3**

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:

* 1. За основу беремо програму калькулятор з минулої теми.
  2. Додаємо цикл while, який виводить на екран, що треба ввести, щоб виконати відповідну операцію та далі відповідно просить ввести значення та розраховує їх. Оскільки цей цикл є безперервним (бо умовою є True), то я додала умову, якщо в операцію було введенно “0”, то цикл закінчується.

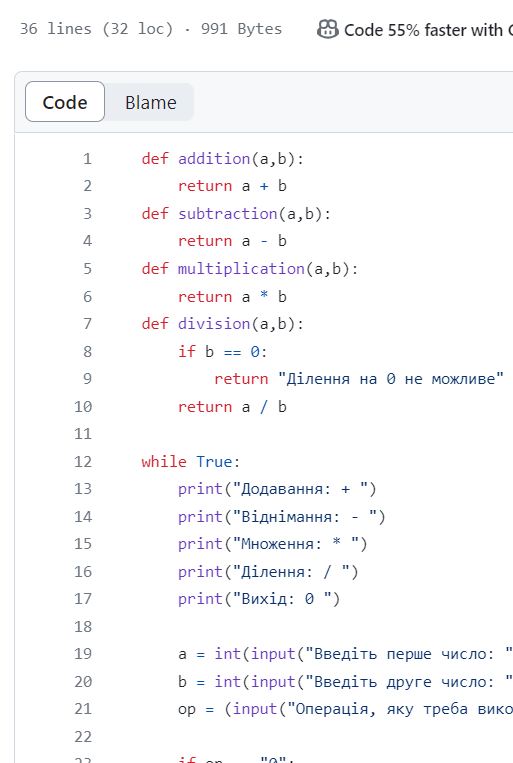
Текст програми:

|  |
| --- |
| def addition(a,b):  return a + b  def subtraction(a,b):  return a - b  def multiplication(a,b):  return a \* b  def division(a,b):  if b == 0:  return "Ділення на 0 не можливе"  return a / b  while True:  print("Додавання: + ")  print("Віднімання: - ")  print("Множення: \* ")  print("Ділення: / ")  print("Вихід: 0 ")  a = int(input("Введіть перше число: "))  b = int(input("Введіть друге число: "))  op = (input("Операція, яку треба виконати: "))  if op == "0":  break  match op:  case "+":  print(addition(a,b))  case "-":  print(subtraction(a,b))  case "\*":  print(multiplication(a,b))  case "/":  print(division(a,b))  case \_:  print("Неправильно введена операція, спробуйте \*,/,-,+") |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_03/task1.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_03/task1.py)

Знімок екрану з посилання на github:



**Тестування функцій 1**

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

Хід виконання завдання:

* 1. Спочатку створюємо змінну list зі значенням списку 1, 2, 3. Та виводимо на екран початковий список.
  2. Тестуємо функції:
     + extend(iterable): дозволяє додати всі елементи в кінець поточного списку.
     + append(value): додає один вказаний елемент в кінець списку.
     + insert(index, value): дозволяє вставити значення value на позицію index у списку.
     + remove(value):видаляє елемент у списку зі значенням value
     + clear(): очищає весь вміст списку.
     + sort(): сортує елементи списку в зростаючому порядку за замовчуванням.
     + reverse(): обертає порядок елементів у списку.
     + copy(): створює та повертає копію списку.

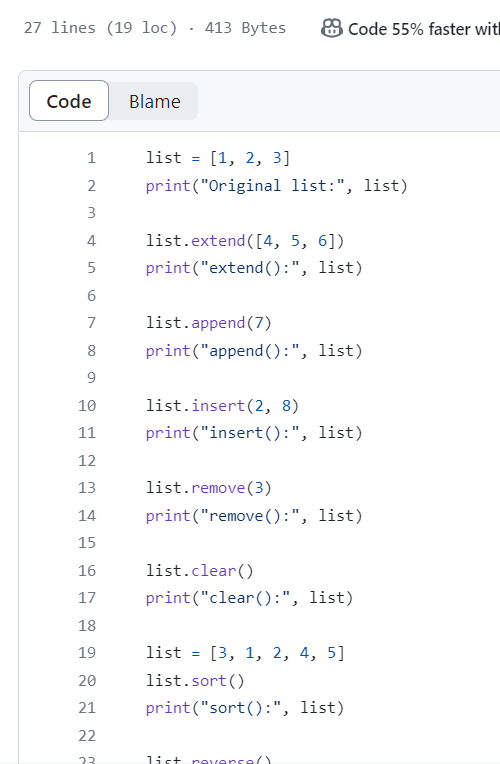
Текст програми:

|  |
| --- |
| list = [1, 2, 3]  print("Original list:", list)  list.extend([4, 5, 6])  print("extend():", list)  list.append(7)  print("append():", list)  list.insert(2, 8)  print("insert():", list)  list.remove(3)  print("remove():", list)  list.clear()  print("clear():", list)  list = [3, 1, 2, 4, 5]  list.sort()  print("sort():", list)  list.reverse()  print("reverse():", list)    list\_copy = list.copy()  print("copy():", list\_copy) |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_03/task2.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_03/task2.py)

Знімок екрану з посилання на github:



**Тестування функцій 2**

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

Хід виконання завдання:

* 1. Спочатку створюємо змінну dict зі значенням списку 1, 2, 3. Та виводимо на екран початковий словник.
  2. Тестуємо функції:
     + update(other\_dict): додавання пар ключ-значення з іншого словника other\_dict до поточного словника. Якщо ключі вже існують у поточному словнику, значення будуть оновлені.
     + del dict[key]: видалення ключа та його відповідного значення зі словника.
     + clear(): очищення всього вмісту словника.
     + keys(): отримання списку всіх ключів у словнику.
     + values(): отримання списку всіх значень у словнику.
     + items(): отримання списку всіх пар ключ-значення у словнику.

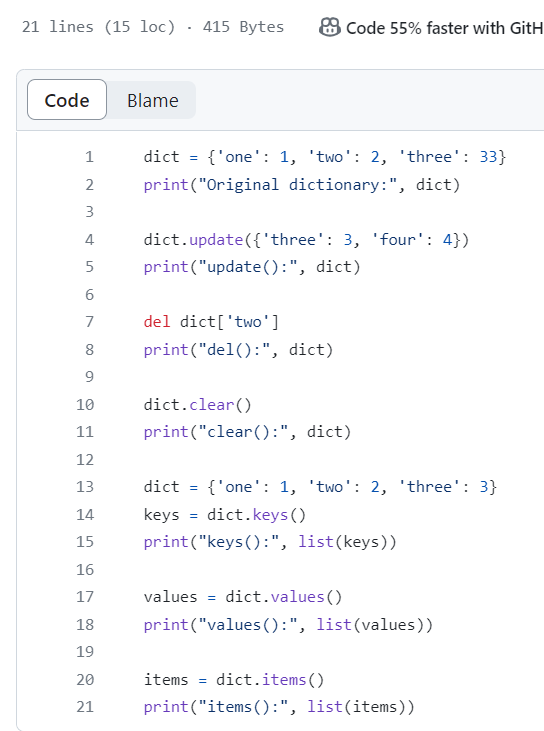
Текст програми:

|  |
| --- |
| dict = {'one': 1, 'two': 2, 'three': 33}  print("Original dictionary:", dict)  dict.update({'three': 3, 'four': 4})  print("update():", dict)  del dict['two']  print("del():", dict)  dict.clear()  print("clear():", dict)  dict = {'one': 1, 'two': 2, 'three': 3}  keys = dict.keys()  print("keys():", list(keys))  values = dict.values()  print("values():", list(values))  items = dict.items()  print("items():", list(items)) |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_03/task3.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_03/task3.py)

Знімок екрану з посилання на github:



**Пошук позицій**

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

Хід виконання завдання:

* 1. Спочатку створюємо список numbers, який містить початкові значення [1, 3, 5].
  2. Далі створюємо безкінечний цикл while True, який продовжується, поки користувач не введе слово "вихід":
     + Програма виводить поточний стан списку numbers, використовуючи print().
     + Користувач повинен ввести нове число або вийти з програми, використовуючи input(). Введене значення зберігається у змінній new\_number.
     + Перевіряється, чи введене значення є "вихід". Якщо так, цикл завершується за допомогою break.
     + Якщо введене значення не є "вихід", то new\_number перетворюється в ціле число за допомогою int().
     + Визначаємо змінну insert\_index, яка дорівнює 0.
     + Потім виконується ітерація по числах у списку numbers за допомогою циклу for. Для кожного числа num у списку порівнюється new\_number та num. Якщо new\_number більше за num, змінна insert\_index збільшується на одиницю. Це визначає позицію, на яку потрібно вставити new\_number.
     + Далі функція insert() використовується для вставки new\_number на визначену позицію insert\_index у списку numbers.
     + Цикл while продовжується, дозволяючи користувачеві вводити інші числа або вийти з програми.

Текст програми:

|  |
| --- |
| numbers = [1, 3, 5,]  while True:  print("Список:", numbers)  new\_number = input("Введіть нове число, чи введіть 'вихід', якщо хочете вийти: ")  if new\_number == 'вихід':  break  new\_number = int(new\_number)  insert\_index = 0  for num in numbers:  if new\_number > num:  insert\_index += 1  numbers.insert(insert\_index, new\_number) |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_03/task4.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_03/task4.py)

Знімок екрану з посилання на github:



Звіт до Теми №4

Виняткові ситуації

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

**Калькулятор 4**

Розширити програму калькулятор функцією запитів від користувача, що обробляє виняткові ситуації.

Хід виконання завдання:

* 1. Частина коду аналогічна до минулої реалізації калькулятора.
  2. Додано функцію value, яка отримує рядок promt та виводить його на екран і очікує від користувача введення цілого числа. Якщо користувач введе ціле число, то це число повертається з функції. А якщо користувач введе рядок або нечислове значення, виникає виняток ValueError і виводиться повідомлення "Неправильне значення (введіть число)", і цикл продовжує очікувати правильного введення від користувача.
  3. Ця функція використовується в змінних для значень a та b.

Текст програми:

|  |
| --- |
| def addition(a,b):  return a + b  def subtraction(a,b):  return a - b  def multiplication(a,b):  return a \* b  def division(a,b):  if b == 0:  return "Ділення на 0 не можливе"  return a / b  def value(promt:str):  while True:  try:  return int(input(promt))  except ValueError:  print("Неправильне значення (введіть число)")    while True:  print("Додавання: + ")  print("Віднімання: - ")  print("Множення: \* ")  print("Ділення: / ")  print("Вихід: 0 ")  a = value("Введіть значення a: ")  b = value("Введіть значення b: ")  op = (input("Операція, яку треба виконати: "))  if op == "0":  break  match op:  case "+":  print(addition(a,b))  case "-":  print(subtraction(a,b))  case "\*":  print(multiplication(a,b))  case "/":  print(division(a,b))  case \_:  print("Неправильно введена операція, спробуйте \*,/,-,+") |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_04/task1.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_04/task1.py)

Знімок екрану з посилання на github:



**Калькулятор 5**

Розширити функцію ділення обробкою виняткової ситуації ділення но нуль.

Хід виконання завдання:

* 1. Частина коду аналогічна до минулої реалізації калькулятора.
  2. Змінено функцію division(a,b). Спочатку йде перевірка чи b дорівнює 0, якщо b рівне 0, то вона робить виняток ZeroDivisionError з повідомленням "Ділення на нуль неможливе". Якщо b не дорівнює 0, то функція виконує операцію ділення a / b і повертає результат цієї операції.
  3. Далі якщо все ж таки b дорівнює 0 і виникне виняток ZeroDivisionError, функція обробляє цей виняток у блоку except. У цьому блоку виняток ZeroDivisionError зберігається у змінній e. Потім функція конвертує цей виняток у рядок за допомогою str(e) і повертає його як результат. Тобто функція поверне рядок "Ділення на нуль неможливе".
  4. Також вкінці змінений код. Додано змінну result зі значенням None. Далі використовується конструкція if еlif else, щоб визначити, яку математичну операцію вибрав користувач і зберігається в змінній op.
  5. Після виконання відповідної операції, перевіряється, чи змінна result містить результат обраної операції. Якщо result не рівне None, то виводиться результат на екран.
  6. Якщо змінна result залишилася рівною None, виводиться повідомлення "Неправильно введена операція, спробуйте \*, /, -, +".

Текст програми:

|  |
| --- |
| def addition(a,b):  return a + b  def subtraction(a,b):  return a - b  def multiplication(a,b):  return a \* b  def division(a,b):  try:  if b == 0:  raise ZeroDivisionError("Ділення на нуль неможливе")  return a / b  except ZeroDivisionError as e:  return str(e)  def value(promt:str):  while True:  try:  return int(input(promt))  except ValueError:  print("Неправильне значення (введіть число)")    while True:  print("Додавання: + ")  print("Віднімання: - ")  print("Множення: \* ")  print("Ділення: / ")  print("Вихід: 0 ")  a = value("Введіть значення a: ")  b = value("Введіть значення b: ")  op = (input("Операція, яку треба виконати: "))  if op == "0":  break  result = None  if op == "+":  result = addition(a, b)  elif op == "-":  result = subtraction(a, b)  elif op == "\*":  result = multiplication(a, b)  elif op == "/":  result = division(a, b)  if result is not None:  print(f"{result}\n")  else:  print("Неправильно введена операція, спробуйте \*, /, -, +") |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_04/task2.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_04/task2.py)

Знімок екрану з посилання на github:



**Різноманітні виняткові ситуації**

Ознайомитись зі списком виняткових ситуацій за посиланням <https://docs.python.org/3/library/exceptions.html>.

Хід виконання завдання:

* 1. Перейшла за наданим посиланням.
  2. Ознайомилась з винятковими ситуаціями.
  3. Реалізувала деякі з них:
     + ValueError - функція отримує аргумент правильного типу, але з недійсним значенням.
     + FileNotFoundError - намагаєтеся відкрити файл, який не існує.
     + KeyError - намагаєтеся отримати значення за ключем, який не існує в словнику.
     + IndexError - намагаєтеся звернутися до неіснуючого індексу в списку чи кортежі.
     + TypeError - викликаєте функцію з аргументом невірного типу.
     + NameError - намагаєтеся використовувати змінну, яка не була раніше оголошена.
     + KeyboardInterrupt - прериваєте виконання програми натисканням клавіші Ctrl+C.

Текст програми:

|  |
| --- |
| try:  number = int("не число")  except ValueError as e:  print("Недійсне значення")  try:  with open("non-existent\_file.txt", "r") as file:  content = file.read()  except FileNotFoundError:  print("Неіснуючий файл")  try:  my\_dict = {"name": "John"}  value = my\_dict["age"]  except KeyError:  print("Значення ключа не існує в словнику")  try:  my\_list = [1, 2, 3]  item = my\_list[5]  except IndexError:  print("Неіснуючий індекс")  try:  result = "10" + 5  except TypeError:  print("Аргумент невірного типу")  try:  print(undefined\_variable)  except NameError as e:  print("Неіснюча змінна")  try:  while True:  pass  except KeyboardInterrupt:  print("Програму перервано користувачем") |

Посилання на github:

[TP-KB-222-Anna-Dryzhak/topic\_04/task3.py at main · aanutadrzhk/TP-KB-222-Anna-Dryzhak (github.com)](https://github.com/aanutadrzhk/TP-KB-222-Anna-Dryzhak/blob/main/topic_04/task3.py)

Знімок екрану з посилання на github:

