**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:

Результат виконання програми:



Текст програми:

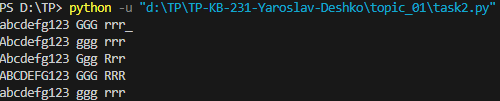
|  |
| --- |
| myStr = "abcdefg123"  print(myStr[::-1]) |

Тестування функцій

Виконати тестування функцій, що працюють з рядками: strip(), capitalize(), title(), upper(), lower().

Хід виконання завдання:

Результат виконання програми:



Текст програми:

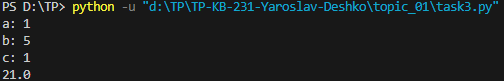
|  |
| --- |
| myStr = "abcdefg123 GGG rrr  "  print(myStr.strip() + "\_")  print(myStr.capitalize())  print(myStr.title())  print(myStr.upper())  print(myStr.lower()) |

Функція пошуку дискримінанта.

Написати функцію пошуку дискримінанту квадратного рівняння.

Хід виконання завдання:

Результат виконання програми:



Текст програми:

|  |
| --- |
| def discriminant(a: float, b: float, c: float) -> float:      return pow(b, 2) - 4\*a\*c  a = float(input("a: "))  b = float(input("b: "))  c = float(input("c: "))  print(discriminant(a, b, c)) |

Звіт до Теми №2

Умовний перехід

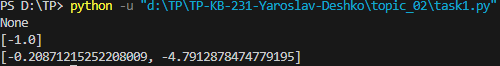
Під час виконання практичного завдання до Теми №2 було надано варіанти рішення до наступних задач:

Пошук коренів квадратного рівняння.

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

Хід виконання завдання:

Результат виконання програми:



Текст програми:

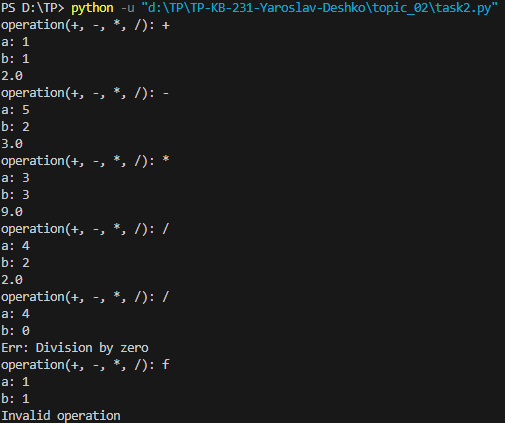
|  |
| --- |
| from math import sqrt  def discriminant(a: float, b: float, c: float) -> float:      return pow(b, 2) - 4\*a\*c  def findRoots(a: float, b: float, c: float):    d = discriminant(a, b, c)    if d == 0:       return [-(b/2\*a)]    elif d > 0:       return [((-b)+sqrt(d))/ 2\*a,               ((-b)-sqrt(d))/ 2\*a ]    return None  def test():    print(findRoots(1, 2, 2)) # D < 0    print(findRoots(1, 2, 1)) # D == 0    print(findRoots(1, 5, 1)) # D > 0      test() |

Калькулятор на **if else.**

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

Хід виконання завдання:

Результат виконання програми:



Текст програми:

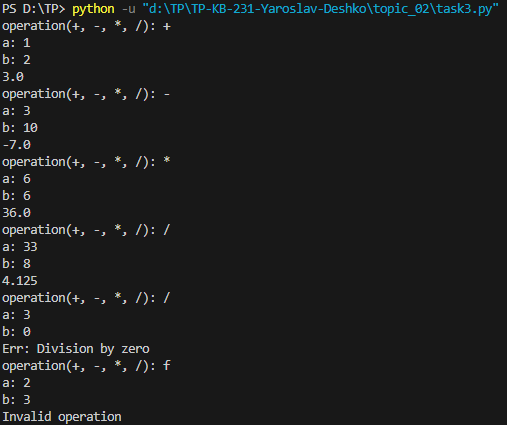
|  |
| --- |
| def calculator():      opr = input("operation(+, -, \*, /): ")      a, b = enterNum()      if opr == "+":         return add(a, b)      elif opr == "-":         return sub(a, b)      elif opr == "\*":         return mul(a, b)      elif opr == "/":         return div(a, b)      else: return "Invalid operation"  def enterNum():      a = input("a: ")      b = input("b: ")      return float(a), float(b)  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):      if b == 0 : return "Err: Division by zero"      return a/b  while True:      print(calculator()) |

Калькулятор на **match.**

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

Хід виконання завдання:

Результат виконання програми:



Текст програми:

|  |
| --- |
| def calculator():      opr = input("operation(+, -, \*, /): ")      a, b = enterNum()      match opr:          case "+":              return add(a, b)          case "-":              return sub(a, b)          case "\*":              return mul(a, b)          case "/":              return div(a, b)          case \_:              return "Invalid operation"  def enterNum():      a = input("a: ")      b = input("b: ")      return float(a), float(b)  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):      if b == 0 : return "Err: Division by zero"      return a/b  while True:      print(calculator()) |

Звіт до Теми №3

Цикли

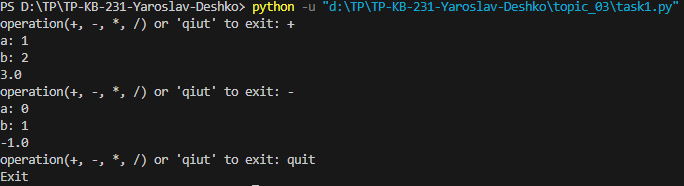
Під час виконання практичного завдання до Теми №3 було надано варіанти рішення до наступних задач:

Калькулятор з постійними запитами на введення.

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:

Результат виконання програми:



Текст програми:

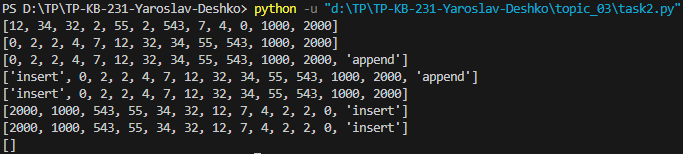
|  |
| --- |
| def calculator(opr: str):      a, b = enterNum()      match opr:          case "+":              return add(a, b)          case "-":              return sub(a, b)          case "\*":              return mul(a, b)          case "/":              return div(a, b)          case \_:              return "Invalid operation"      def enterNum():      a = input("a: ")      b = input("b: ")      return float(a), float(b)  def enterOpr():      opr = input("operation(+, -, \*, /) or 'qiut' to exit: ")      return opr  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):      if b == 0 : return "Err: Division by zero"      return a/b  while True:      opr = enterOpr()      if opr!="quit":          print(calculator(opr))      else:          print("Exit")          break |

Тестування функцій списків.

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

Хід виконання завдання:

Результат виконання програми:



Тест програми:

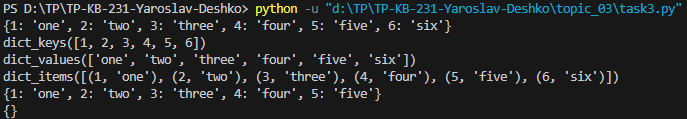
|  |
| --- |
| myList = [12,34,32,2,55,2,543,7,4,0]  myList.extend([1000, 2000])  print(myList)  myList.sort()  print(myList)  myList.append("append")  print(myList)  myList.insert(0, "insert")  print(myList)  myList.remove("append")  print(myList)  myList.reverse()  print(myList)  newList = myList.copy()  print(newList)  myList.clear()  print(myList) |

Тестування функцій словників.

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

Хід виконання завдання:

Результат виконання програми:



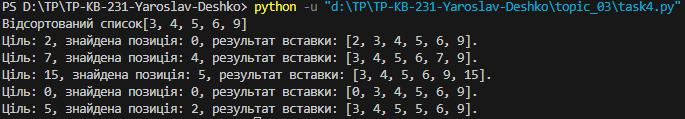
Текст програми:

|  |
| --- |
| myDict = {      1 : "one",      2: "two",      3: "three",      4: "four",      5: "five",  }  myDict.update({6:"six"})  print(myDict)  print(myDict.keys())  print(myDict.values())  print(myDict.items())  del myDict[6]  print(myDict)  myDict.clear()  print(myDict) |

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

Хід виконання завдання:

Результат виконання програми:



Текст програми:

|  |
| --- |
| def findInsertPosition(sortedList: list, target: int) -> int:      left = 0      right = len(sortedList)      while left < right:          mid = (left+right)//2          if sortedList[mid] < target:              left = mid+1          else: right = mid      return left  def insertNum(sortedList: list, num: int)-> str:      workList= sortedList.copy()      index = findInsertPosition(workList, num)      workList.insert(index, num)      return f"Ціль: {num}, знайдена позиція: {index}, результат вставки: {workList}."  sortedList = [3, 4, 5, 6, 9]  print(f"Відсортований список{sortedList}")  print(insertNum(sortedList, 2))  print(insertNum(sortedList, 7))  print(insertNum(sortedList, 15))  print(insertNum(sortedList, 0))  print(insertNum(sortedList, 5)) |

Звіт до Теми №4

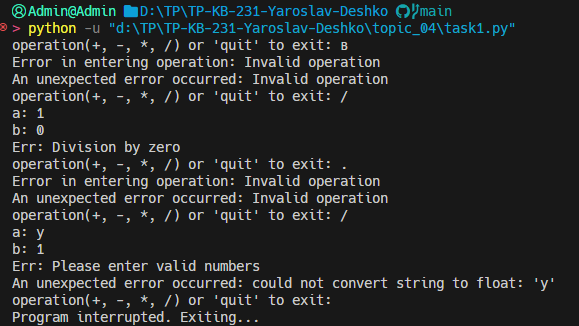
Виняткові ситуації

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

1. Розширити програму калькулятор функцією запитів даних для виконання операцій від користувача, що обробляє виняткові ситуації.
2. Розширити функцію ділення обробкою виняткової ситуації ділення но нуль
3. Ознайомитись зі списком виняткових ситуацій за посиланням <https://docs.python.org/3/library/exceptions.html>

Хід виконання завдання:

Результат виконання програми:



Текст програми:

|  |
| --- |
| def calculator(opr: str):      a, b = enterNum()      match opr:          case "+":              return add(a, b)          case "-":              return sub(a, b)          case "\*":              return mul(a, b)          case "/":              return div(a, b)  def enterNum():      try:          a = input("a: ")          b = input("b: ")          return float(a), float(b)      except ValueError:          print("Err: Please enter valid numbers")          raise  def enterOpr():      try:          opr = input("operation(+, -, \*, /) or 'quit' to exit: ").strip().lower()          valid\_operations = ['+', '-', '\*', '/', 'quit']          if opr not in valid\_operations:              raise ValueError("Invalid operation")            return opr        except Exception as e:          print(f"Error in entering operation: {e}")          raise    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:          return a / b      except ZeroDivisionError:          return "Err: Division by zero"    while True:      try:          opr = enterOpr()          if opr == "quit":              print("Exit")              break            result = calculator(opr)          print(result)        except KeyboardInterrupt:          print("\nProgram interrupted. Exiting...")          break      except Exception as e:          print(f"An unexpected error occurred: {e}") |

Посилання на github:

https://github.com/Kaspo02/TP-KB-231-Yaroslav-Deshko

Знімок екрану з посилання на github:

