Звіт до Теми №1

Функції та змінні

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

**Перетворення рядка**

Хід виконання завдання:

1. Необхідно рядок, що має вигляд "Hello, World!" перетворити наступним чином "!dlroW ,olleH", вважаючи сталою довжину рядку в 10 символів.

Текст програми:

| def reverse\_string(text: str) -> str:  return text [::-1]  s = "Hello, World!"  print("Original:", s)  print("Reverse", reverse\_string(s)) |
| --- |

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

Текст програми:

| text = " Text "  print ("Original:", repr(text))  print ("strip():", repr(text.strip()))  print ("capitalize():", text.capitalize())  print ("title():", text.title())  print ("upper():", text.upper())  print ("lower():", text.lower()) |
| --- |

3. Треба написати функцію пошуку дискримінанту квадратного рівняння, використовуються числа a = 1, b = 2, c = 3.

Текст програми:

| def discriminant(a: float, b: float, c: float) -> float:  return b\*\*2 - 4\*a\*c  print(discriminant(1, 2, 3)) |
| --- |

Звіт до Теми №2

Умовний перехід

**Перетворення рядка**

Хід виконання завдання:

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

Текст програми:

| import math  def discriminant(a: float, b: float, c: float) -> float:  return b\*\*2 - 4 \* a \* c  def solve\_quadratic(a: float, b: float, c: float):  D = discriminant(a, b, c)  if D > 0:  x1 = (-b + math.sqrt(D)) / (2 \* a)  x2 = (-b - math.sqrt(D)) / (2 \* a)  return (x1, x2)  elif D == 0:  x = -b / (2 \* a)  return x  else:  return None    a = int(input("What's A: "))  b = int(input("What's B: "))  c = int(input("What's C: "))  result = solve\_quadratic(a, b, c)  if result is None:  print("There are no result")  elif isinstance(result, tuple):  print(f"x = {result[0]}, x2 = {result[1]}")  else:  print(f"x = {result}") |
| --- |

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

Текст програми:

| 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:  raise ValueError("Division by zero is impossible")  return x / y  def calculator(x, y, op: str):  if op == "+":  return add(x, y)  elif op == "-":  return subtract(x, y)  elif op == "\*":  return multiply(x, y)  elif op == "/":  return divide(x, y)  else:  raise ValueError(f"Uknown {op}")    x = float(input("Number 1: "))  y = float(input("Number 2: "))  op = input("Choose an action (+, -, \*, /): ")  try:  result = calculator(x, y, op)  print(f"Result: {result}")  except Exception as e:  print("Error: ", e) |
| --- |

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

Текст програми:

| def calculator\_match(x, y, op: str):  match op:  case "+":  return x + y  case "-":  return x - y  case "\*":  return x \* y  case "/":  if y == 0:  raise ValueError("Division by zero is impossible")  return x / y  case \_:  raise ValueError(f"Uknown: {op}")  x = float(input("Number 1: "))  y = float(input("Number 2: "))  op = input("Chose an action (+, -, \*, /): ")  try:  result = calculator\_match(x, y, op)  print (f"Result: {result}")  except Exception as e:  print ("Error: ", e) |
| --- |

Звіт до Теми №3

[Цикли](https://eln.stu.cn.ua/course/section.php?id=98203)

1. Треба 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.

| def calculator\_match(x, y, op: str):  match op:  case "+":  return x + y  case "-":  return x - y  case "\*":  return x \* y  case "/":  if y == 0:  raise ValueError("Division by zero is impossible")  return x / y  case \_:  raise ValueError(f"Uknown: {op}")  def main():  print("Calculator. Type 'exit' to quit.")  while True:  inp = input("Enter in format: number1 operator number2 - ")  if inp.lower().strip() == "exit":  print("Exiting program.")  break  parts = inp.split()  if len(parts) != 3:  print("Invalid format. Try again.")  continue  x\_str, op, y\_str = parts  try:  x = float(x\_str)  y = float(y\_str)  result = calculator\_match(x, y, op)  print("Result:", result)  except Exception as e:  print("Error:", e)  if \_\_name\_\_ == "\_\_main\_\_":  main() |
| --- |

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

Текст програми:

| def test\_list\_methods():  a = [1, 2, 3]  print("Initial list:", a)  a.append(4)  print("After append(4):", a)  a.extend([5, 6])  print("After extend([5,6]):", a)  a.insert(0, 100)  print("After insert(0, 100):", a)  a.remove(2)  print("After remove(2):", a)  b = a.copy()  print("After list (b):", b)  b.reverse()  print("b after reverse():", b)  b.sort()  print("b after sort():", b)  b.clear()  print("b after clear():", b)  if \_\_name\_\_ == "\_\_main\_\_":  test\_list\_methods() |
| --- |

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

Текст програми:

| def test\_dict\_methods():  d = {"a": 1, "b": 2}  print("Initial dictionary:", d)  d.update({"b": 20, "c": 30})  print("After update({'b':20, 'c':30}):", d)  del d["a"]  print("After del d['a']:", d)  print("keys():", list(d.keys()))  print("values():", list(d.values()))  print("items():", list(d.items()))    d2 = d.copy()  print("copy d2:", d2)  d2.clear()  print("d2 after clear():", d2)  if \_\_name\_\_ == "\_\_main\_\_":  test\_dict\_methods |
| --- |

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

Текст програми:

| def find\_insert\_position(sorted\_list: list, value) -> int:  low = 0  high = len(sorted\_list)  while low < high:  mid = (low + high) // 2  if sorted\_list[mid] <= value:  low = mid + 1  else:  high = mid  return low  def test\_insert\_position():  lst = [1, 3, 5, 7, 9]  print(lst)  for v in [0, 1, 2, 5, 6, 10]:  pos = find\_insert\_position(lst, v)  print(f"Insert {v} at position {pos}")  for v in [0, 4, 8, 10]:  pos = find\_insert\_position(lst, v)  new = lst.copy()  new.insert(pos, v)  print(f"After inserting {v}: {new}")  if \_\_name\_\_ == "\_\_main\_\_":  test\_insert\_position()  except Exception as e:  print ("Error: ", e) |
| --- |

Звіт до Теми №4

[Виняткові ситуації](https://eln.stu.cn.ua/course/section.php?id=98204)

1. **Попередні умови**: реалізована програма калькулятор, що використовує метод нескінченного введення даних для обробки. Всі дії (додавання, віднімання, множення, ділення) реалізовані як окремі функції та використовуються у відповідних місцях.
2. Розширити програму калькулятор функцією запитів даних для виконання операцій від користувача, що обробляє виняткові ситуації.
3. Розширити функцію ділення обробкою виняткової ситуації ділення но нуль
4. Ознайомитись зі списком виняткових ситуацій за посиланням <https://docs.python.org/3/library/exceptions.html>

| 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):  try:  return x / y  except ZeroDivisionError as e:  raise ZeroDivisionError("Cannot divide by zero") from e  def get\_number(prompt: str) -> float:  while True:  s = input(prompt)  try:  value = float(s)  return value  except ValueError:  print("Invalid number. Please try again.")  def get\_operator(prompt: str) -> str:  valid\_ops = {"+", "-", "\*", "/"}  while True:  op = input(prompt).strip()  if op in valid\_ops:  return op  print(f"Invalid operator: {op}. Choose one of {valid\_ops}.")  def calculator\_loop():  print("Calculator. Type 'exit' to quit at any prompt.")  while True:  cmd = input("Press Enter to continue or type 'exit' to quit: ").strip().lower()  if cmd == "exit":  print("Exiting program.")  break  x = get\_number("Enter first number: ")  y = get\_number("Enter second number: ")  op = get\_operator("Enter operator (+, -, \*, /): ")  try:  if op == "+":  result = add(x, y)  elif op == "-":  result = subtract(x, y)  elif op == "\*":  result = multiply(x, y)  elif op == "/":  result = divide(x, y)  else:  raise ValueError(f"Unsupported operator: {op}")  print(f"Result: {result}")  except ZeroDivisionError as zde:  print("Error:", zde)  except Exception as e:  print("Unexpected error:", e)  if \_\_name\_\_ == "\_\_main\_\_":  calculator\_loop() |
| --- |

Посилання на github: [turovetsvanya/TP-KB-242-Turovets-Ivan](https://github.com/turovetsvanya/TP-KB-242-Turovets-Ivan#)

Знімок екрану з посилання на github:

