**Лабораторная работа №4**

**«СОЗДАНИЕ МЕНЮ В PYTHON-ПРОГРАММЕ. ИНКАПСУЛЯЦИЯ»**

**Задание**

Выполнить расчет системы кусочно-ломанной функции из ЛР№2 при вводимых с клавиатуры параметрах. Создать меню работы программы с пунктами, осуществляющими выход из программы и расчет выражения.

Для выполнения задания следовало разработать класс с методом-конструктором для расчета выражения, включающий в себя методы branch1(), branch2(), branch3(), рассчитывающие выражение для ветвей системы. Конструктор принимает в качестве аргумента входные параметры расчетной функции.

Для создания меню требовалось создать цикл, включающий в себя всю основную часть программы, в начале которого с клавиатуры вводится одно из двух действий, выраженное числом: выход из программы, вычисление функции.

**Ход работы**

Для выполнения поставленной задачи требовалось создать меню с помощью цикла, включающего всю основную часть программы. На рисунке 1 изображена блок схема основной части программы и реализующая функционал меню.

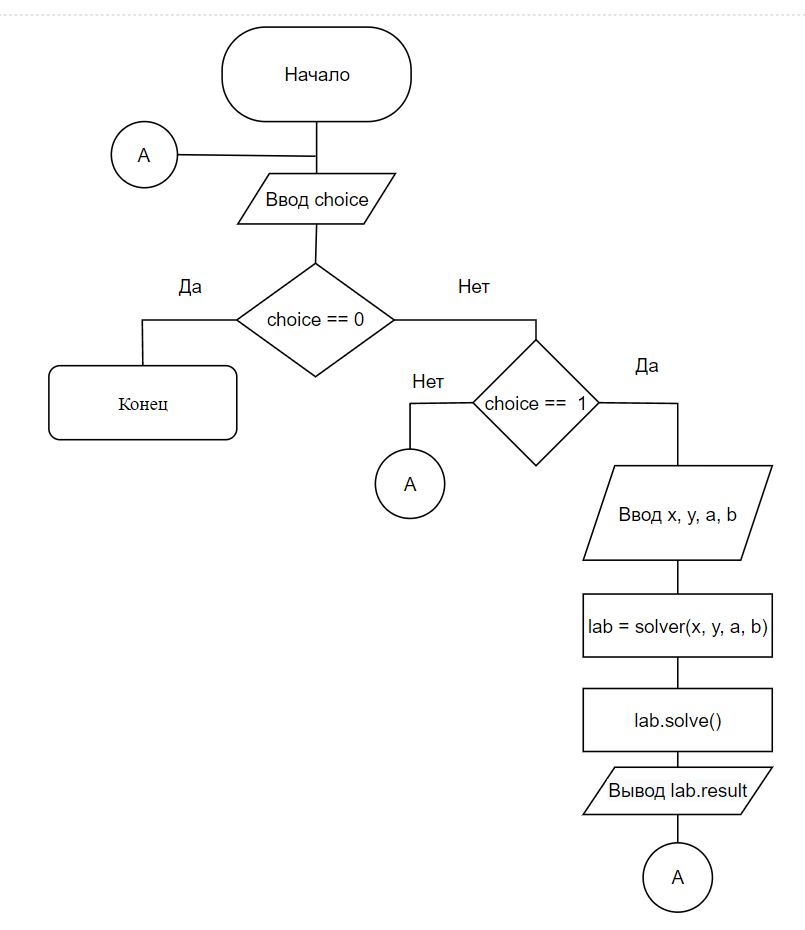


Рисунок 1 - Блок-схема основной части программы

Для расчета выражения был создан класс solver, с включающий в себя 4 метода, три метода для расчета ветвей системы и интерфейсный публичный метод в котором происходит выбор ветви. Необходимость создания данного метода вытекает из требований лабораторной работы по демонстрации приватных (закрытых) методов. На рисунках 2, 3 изображены блок-схемы методов класса.

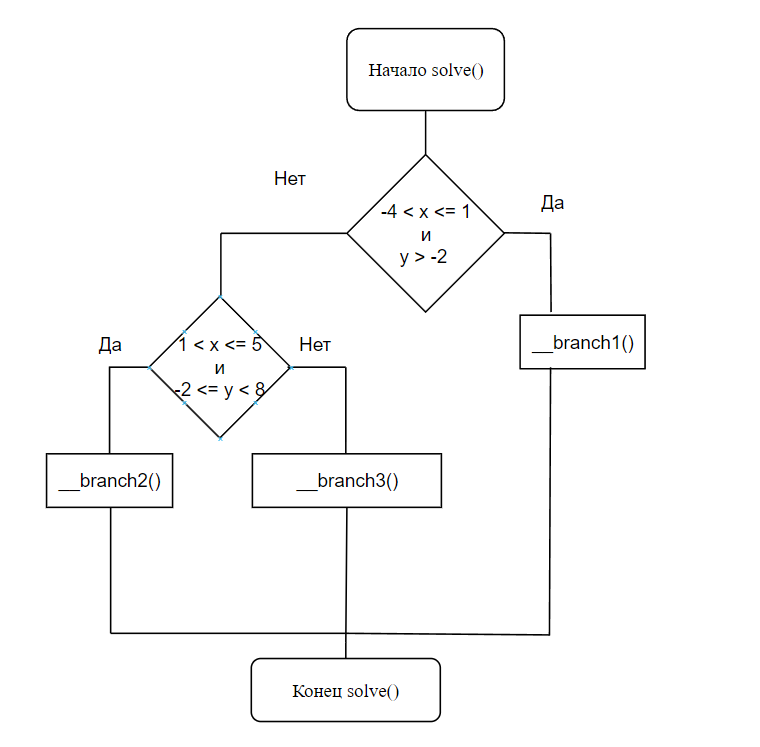
****

Рисунок 2 - Блок-схема метода solve()

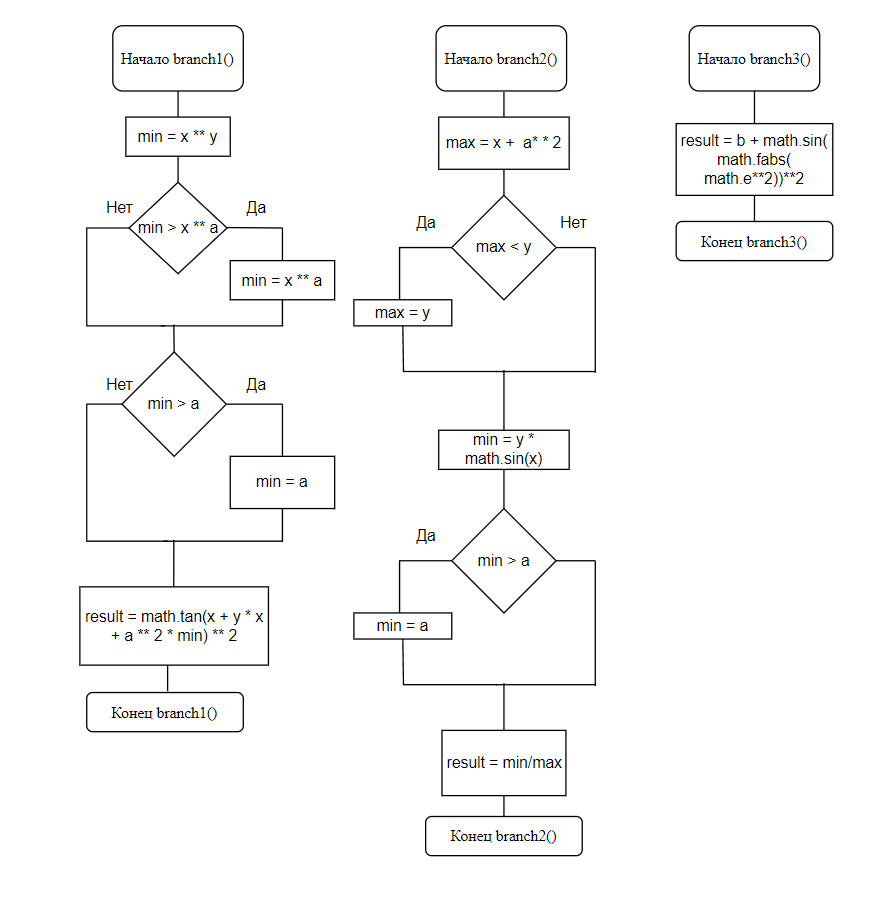
****

Рисунок 3 - Блок-схема методов для решения ветвей системы

|  |
| --- |
| Листинг 1 – код основной части программы |
| from classModule import \*  choice = None  while choice != "0":  print \  ("""  Меню  0 - Выйти  1 - Выполнить программу  """)  choice = input("Сделайте выбор ")  print()  if choice == "0":  print("Заканчиваем работу")  elif choice == "1":  print("Введите исходные данные ")  x = float(input("Введите x "))  y = float(input("Введите y "))  a = float(input("Введите a "))  b = float(input("Введите b "))  lab = solver(x, y, a, b)  lab.solve()  print("Результат вычислений: " + str(lab.result))  else:  print("Такого пункта в меню нет ", choice) |

|  |
| --- |
| Листинг 2 – код модуля с классом |
| import math  class solver:  def \_\_init\_\_(self, x, y, a, b):  self.\_\_x = x  self.\_\_y = y  self.\_\_a = a  self.\_\_b = b  self.result = 0  def \_\_branch1(self):  min = self.\_\_x \*\* self.\_\_y  if min > math.e \*\* self.\_\_x:  min = math.e \*\* self.\_\_x  if min > self.\_\_a:  min = self.\_\_a  self.result = math.tan(self.\_\_x + self.\_\_y \* self.\_\_x + self.\_\_a \*\* 2 \* min) \*\* 2  def \_\_branch2(self):  max = self.\_\_x + self.\_\_a \*\* 2  if max < self.\_\_y:  max = self.\_\_y  min = self.\_\_y \* math.sin(self.\_\_x)  if min > self.\_\_a:  min = self.\_\_a  try:  self.result = max / min  except Exception as e:  print("Деление на ноль!")  exit()  def \_\_branch3(self):  self.result = self.\_\_b + math.sin(math.fabs(math.e \*\* self.\_\_x)) \*\* 2  def solve(self):  if -4 < self.\_\_x <= 1 and self.\_\_y > -2:  self.\_\_branch1()  elif 1 < self.\_\_x <= 5 and -2 <= self.\_\_y < 8:  self.\_\_branch2()  else:  self.\_\_branch3() |

**Результаты**

На рисунках 4-7 изображены результаты тестирования программы и сравнения с результатами программы из ЛР№2. На рисунках 8 – 9 изображены результаты обращения к закрытым методам и атрибутам извне класса.

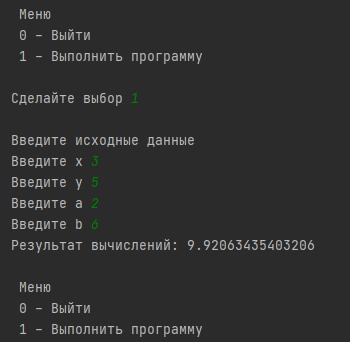


Рисунок 4 - Результат работы программы при вводимых данных из ЛР№2

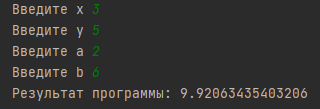


Рисунок 5 - Результат из ЛР №2

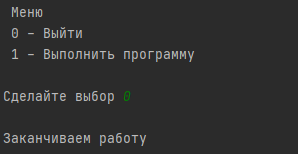
****

Рисунок 6 - Тестирование выхода из программы

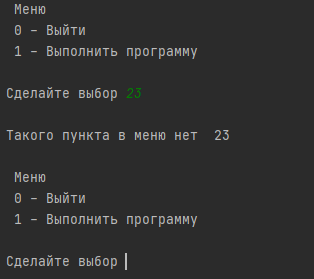


Рисунок 7 - Тестирование ввода некорректного действия

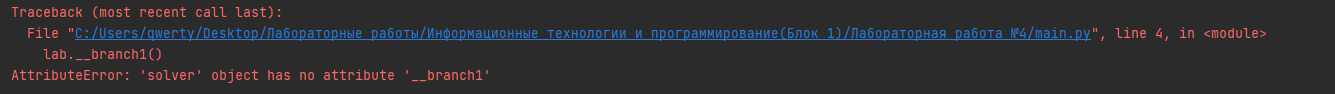
****

Рисунок 8 - Попытка обратиться к приватному методу

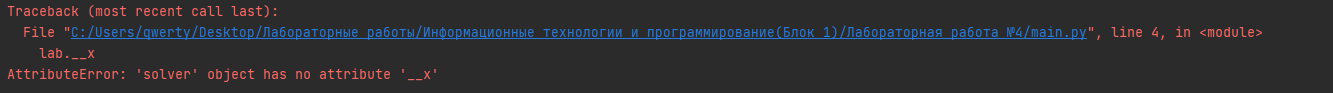
****

Рисунок 9 - Попытка обратиться к приватному атрибуту

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1) Гуриков, С. Р. Основы алгоритмизации и программирования на Python : учебное пособие / С.Р. Гуриков. — Москва : ИНФРА-М, 2022. — 343 с. — (Высшее образование: Бакалавриат). - ISBN 978-5-16-017142-5. - Текст : электронный. - URL: https://znanium.com/catalog/product/1356003 . – Режим доступа: по подписке. + библиотека МТУСИ

2) Дроботун, Н. В. Алгоритмизация и программирование. Язык Python : учебное пособие / Н. В. Дроботун, Е. О. Рудков, Н. А. Баев. — Санкт-Петербург : Санкт-Петербургский государственный университет промышленных технологий и дизайна, 2020. — 119 c. — ISBN 978-5-7937-1829-5. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/102400.html  
  
 3) Шелудько, В. М. Основы программирования на языке высокого уровня Python : учебное пособие / В. М. Шелудько. — Ростов-на-Дону, Таганрог : Издательство Южного федерального университета, 2017. — 146 c. — ISBN 978-5-9275-2649-9. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: https://www.iprbookshop.ru/87461.html (дата обращения: 17.10.2021). — Режим доступа: для авторизир. пользователей  
  
 4) Шелудько, В. М. Язык программирования высокого уровня Python. Функции, структуры данных, дополнительные модули : учебное пособие / В. М. Шелудько. — Ростов-на-Дону, Таганрог : Издательство Южного федерального университета, 2017. — 107 c. — ISBN 978-5-9275-2648-2. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: https://www.iprbookshop.ru/87530.html (дата обращения: 17.10.2021). — Режим доступа: для авторизир. пользователей

5) Коломейченко, А. С. Информационные технологии : учебное пособие для вузов / А. С. Коломейченко, Н. В. Польшакова, О. В. Чеха. — 2-е изд., перераб. — Санкт-Петербург : Лань, 2021. — 212 с. — ISBN 978-5-8114-7564-3. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/177030>