**Лабораторная работа №8**

**«ПРОГРАММИРОВАНИЕ PYTHON-ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ TKinter»**

**Часть 1**

**Задание**

Требовалось разработать программу с графическим интерфейсом, созданным при помощи библиотеки “TKinter” для решения системы кусочно-ломанной функции.

Дана система:

Для решения этой системы требовалось создать разветвляющийся алгоритм (Алгоритм приведен в ЛР№2), и создать графический интерфейс в библиотеке “TKinter” на основе кода данного в методических указаниях.

**Ход работы**

Код вычисляющий значение кусочно-ломанной функции взят из ЛР№2 с удалением из него оператора return. Код создающий графический интерфейс создан на основе листинга 178[[1]](#footnote-1).

Далее приведен код программы.

|  |
| --- |
| Листинг 1 – код программы |
| import math  from tkinter import \*  import logging  class Application(Frame):  def \_\_init\_\_(self, fr):  super(Application, self).\_\_init\_\_(fr)  self.grid()  self.widget()  # Создаем метод для нахождения значения функции  def func(self):  x = float(self.textbox1.get())  y = float(self.textbox2.get())  a = float(self.textbox3.get())  b = float(self.textbox4.get())  result = 0  if -4 < x <= 1 and y > -2:  min = x \*\* y  if min > math.e \*\* x:  min = math.e \*\* x  if min > a:  min = a  result = math.tan(x + y \* x + a \*\* 2 \* min) \*\* 2  elif 1 < x <= 5 and -2 <= y < 8:  max = x + a \*\* 2  if max < y:  max = y  min = y \* math.sin(x)  if min > a:  min = a  try:  result = max / min  except Exception as e:  print("Деление на ноль!")  logging.error(str(e))  exit()  else:  result = b + math.sin(math.fabs(math.e \*\* x)) \*\* 2  self.textbox5.insert(0.0, result)  # Создаем виджеты на форме  def widget(self):  self.lbl1 = Label(self, text="Введите число x", font=("Times New Roman", 16), fg="blue")  self.lbl1.grid(row=0, column=0)  self.textbox1 = Entry(self, width=67, font=("Times New Roman", 16))  self.textbox1.grid(row=1, column=0)  # ----------------------------------------  self.lbl2 = Label(self, text="Введите значение y", font=("Times New Roman", 16), fg="blue")  self.lbl2.grid(row=2, column=0)  self.textbox2 = Entry(self, width=67, font=("Times New Roman", 16))  self.textbox2.grid(row=3, column=0)  # ----------------------------------------  self.lbl3 = Label(self, text="Введите значение a", font=("Times New Roman", 16), fg="blue")  self.lbl3.grid(row=4, column=0)  self.textbox3 = Entry(self, width=67, font=("Times New Roman", 16))  self.textbox3.grid(row=5, column=0)  # ----------------------------------------  self.lbl4 = Label(self, text="Введите значение b", font=("Times New Roman", 16), fg="blue")  self.lbl4.grid(row=6, column=0)  self.textbox4 = Entry(self, width=67, font=("Times New Roman", 16))  self.textbox4.grid(row=7, column=0)  # ----------------------------------------  self.btn = Button(self, font=("Times New Roman", 14, "bold"), fg="red")  self.btn["text"] = "Вычислить функцию"  self.btn["command"] = self.func  self.btn["width"] = 20  self.btn["height"] = 2  self.btn.grid(row=8, column=0)  self.lbl5 = Label(self, text="Значение функции", font=("Times New Roman", 14), fg="green")  self.lbl5.grid(row=9, column=0)  self.textbox5 = Text(self, width=30, height=1, font=("Times New Roman", 14))  self.textbox5.grid(row=10, column=0)  root = Tk()  root.title("Многозначные ветвления")  root.geometry("740x550")  app = Application(root)  root.mainloop() |

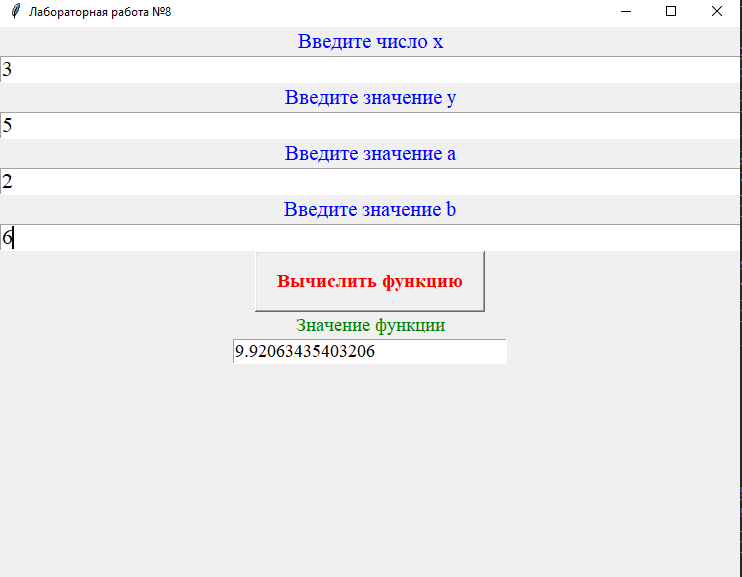
На рисунке 1 приведены результаты работы программы. На рисунке 2 изображены результаты работы программы из ЛР№2. На основании этих рисунков можно заключить о идентичности результатов работы программ.

Рисунок 1- Результат работы программы

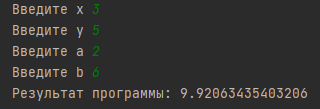


Рисунок 2 - Результат работы программы из ЛР№2

**Часть 2**

**Задание**

Требовалось разработать программу из ЛР№7 добавив функционал записи результатов работы программы в Excel, добавить функционал считывания значений из Excel.

Для выполнения задания следовало добавить в код ЛР№7 функции из библиотеки openpyxl в теле цикла расчета требуемого значения и построчно записывать получившиеся значения.

Также следовало в конце программы добавить код открытия Excel файла и считывания значений, с последующим выводом их на экран. Далее приведен код доработанной программы.

Рисунки по результатам работы программы приложены на отдельном листе к ЛР№7.

Далее приведен код программы.

|  |
| --- |
| Листинг 2 – доработанный код ЛР№7 |
| import math  from openpyxl import Workbook  from openpyxl.styles import Border, Side  import openpyxl  def calcPi(e):  pi = 0  err = 1  counter = int(0)  try:  workbook = openpyxl.load\_workbook('tabResult.xlsx')  except Exception:  workbook = Workbook()  sheet = workbook.active  sheets = workbook.sheetnames  sheet = workbook.active  double = Side(border\_style="double")  while (abs(err) > e/4):  err = (-1) \*\* counter \* 1/(2 \* counter + 1)  pi += err  cell = sheet.cell(row=counter + 1, column=2)  cell.border = Border(left=double, bottom=double)  cell.value = "Номер итерации:"  cell = sheet.cell(row=counter + 1, column=3)  cell.border = Border(bottom=double, right=double)  cell.value = str(counter)  cell = sheet.cell(row=counter + 1, column=4)  cell.border = Border(bottom=double)  cell.value = "Значение:"  cell = sheet.cell(row=counter + 1, column=5)  cell.border = Border(right=double, bottom=double)  cell.value = str(pi \* 4)  counter += 1  print("Номер итерации: " + str(counter) + " Вычисленное значения числа Пи: " + str(pi \* 4))  pi = pi \* 4  dims = {}  for row in sheet.rows:  for cell in row:  if cell.value:  dims[cell.column\_letter] = max((dims.get(cell.column\_letter, 0), len(str(cell.value))))  for col, value in dims.items():  sheet.column\_dimensions[col].width = (value+3)  workbook.save('tabResult.xlsx')  workbook = openpyxl.load\_workbook('tabResult.xlsx')  sheet = workbook.active  for i in range(0, sheet.max\_row):  for col in sheet.iter\_cols(1, sheet.max\_column):  print(col[i].value, end = "\t\t")  print('')  return pi |

**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

1. Гуриков, С. Р. Основы алгоритмизации и программирования на Python : учебное пособие / С.Р. Гуриков. — Москва : ИНФРА-М, 2022. — 283 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 [↑](#footnote-ref-1)