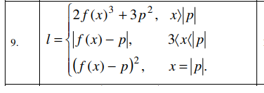
**ПРАКТИЧЕСКАЯ РАБОТА №2. ПРОГРАММИРОВАНИЕ РАЗВЕТВЛЯЮЩИХСЯ АЛГОРИТМОВ**

**Выполнил студент группы ИСП21.1А Жигач Д.Н. Вариант №9**

Цель лабораторной работы: научиться пользоваться простейшими компонентами организации переключений (RadioButton). Написать и отладить программу разветвляющегося алгоритма.

Шаг 1. Задание варианта:



Шаг 2. Создал программу:

import tkinter as tk

from tkinter import messagebox

from math import sin, cos, exp, fabs

class App:

    def \_\_init\_\_(self):

        self.window = tk.Tk()

        self.window.title("pr2")

        self.x\_entry = self.create\_entry("Введите значение X", (0, 0))

        self.y\_entry = self.create\_entry("Введите значение Y", (1, 0))

        self.z\_entry = self.create\_entry("Введите значение Z", (2, 0))

        self.function\_var = tk.StringVar(value="sin")

        self.create\_radio\_button("Sin(x)", "sin", (0, 1))

        self.create\_radio\_button("Cos(x)", "cos", (1, 1))

        self.create\_radio\_button("Exp(x)", "exp", (2, 1))

        self.result\_box = tk.Listbox()

        self.result\_box.grid(row=3, column=0, columnspan=2)

        self.create\_button("Выполнить", self.calculate, (4, 0))

        self.create\_button("Очистить", self.clear, (4, 1))

        self.window.mainloop()

    def create\_entry(self, text, position):

        label = tk.Label(text=text)

        label.grid(row=position[0], column=position[1])

        entry = tk.Entry()

        entry.grid(row=position[0], column=position[1]+1)

        return entry

    def create\_radio\_button(self, text, value, position):

        radio\_button = tk.Radiobutton(text=text, variable=self.function\_var, value=value)

        radio\_button.grid(row=position[0], column=position[1]+2)

    def create\_button(self, text, command, position):

        button = tk.Button(text=text, command=command)

        button.grid(row=position[0], column=position[1])

    def calculate(self):

        try:

            x = float(self.x\_entry.get())

            y = float(self.y\_entry.get())

            z = float(self.z\_entry.get())

        except ValueError:

            messagebox.showerror("Ошибка", "Введены некорректные значения")

            return

        self.result\_box.delete(0, 'end')

        self.result\_box.insert('end', f"X={x}")

        self.result\_box.insert('end', f"Y={y}")

        self.result\_box.insert('end', f"Z={z}")

        try:

            u = self.calculate\_u(x, y, z)

            self.result\_box.insert('end', f"Результат U={round(u, 3)}")

        except Exception as e:

            self.result\_box.insert('end', "Решение не найдено")

    def calculate\_u(self, x, y, z):

        if self.function\_var.get() == "sin":

            func = sin

        elif self.function\_var.get() == "cos":

            func = cos

        elif self.function\_var.get() == "exp":

            func = exp

        else:

            raise ValueError("Неверная функция")

        if x > fabs(z):

            return 2 \* func(x)\*\*3 + (3 \* z\*\*2)

        elif x == fabs(z):

            return (func(x) - z)\*\*2

        else:

            return fabs(func(x) - z)

    def clear(self):

        self.x\_entry.delete(0, 'end')

        self.y\_entry.delete(0, 'end')

        self.z\_entry.delete(0, 'end')

        self.result\_box.delete(0, 'end')

if \_\_name\_\_ == "\_\_main\_\_":

    App()

Шаг 3. Результат:

