**Билет 2**

Выполнил: Пищулин Игорь

Студент П1-17

25.06.2020

1. Ссылки в языках программирования [Приложение 1]

2. Разработать приложение для построения графика полинома n-й степени.

Коэффициенты полинома считываются из StringGrid или аналога.

[Приложение 2]

**Приложение 1**

**Ссылки в языках программирования**

Ссылка – это определенный объект, который указывает на определенные данные, но не хранит их. По сути это другое имя объекта. Это переменная, которая хранит ссылку на другую переменную. Операции над ссылками: копирование и разыменовыывание.

**Пример работы с ссылками в C++.**

Ссылки в C++ обязательно связаны с каким-либо объектом. Разыменовывание не требует дополнительных опреторов.

В данном примере ссылка помогает избежать копирования, присваивания и прочих операторов.

void sum(int x, int y, int &result)

{

result = x + y;

}

int main()

{

int res = 0;

sum(2, 10, res);

cout << "res = " << res << endl; // res = 12

return 0;

}

**Приложение 2**

Разработать приложение для построения графика полинома n-й степени.

Коэффициенты полинома считываются из StringGrid или аналога.

**Листинг 1. Главный модуль**

import matplotlib.pyplot as plt

from polynom import \*

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

# получаем корни уравнения

root = list(map(int, (input('Enter roots of polynom(example: 1 2 3 4 5): ')).split()))

# получаем интервал графика

interval = list(map(int, (input('Enter interval(example: -4 4): ')).split()))

# получаем шаг

step = float(input('Enter step: '))

pol = Polynom(root) # создаем объект полинома

# устанавливаем заголовки окна графика

plt.title(f"График полинома y(x) = {pol}")

plt.xlabel("x") # ось абсцисс

plt.ylabel("y") # ось ординат

# plt.grid() # включение отображение сетки

# получаем списки координат для отрисовки

[x, y] = pol.draw(interval, step)

# выводим четность и нечестность функции

plt.text (x[round(len(x) / 4) \* 3], y[-1], f"Odd: {pol.odd()}")

plt.text (x[round(len(x) / 4) \* 3], y[-1] - (((max(y) - min(y)) / 100) \* 7), f"Even: {pol.even()}")

# создаем график

plt.plot(x, y, label=f"y(x) = {pol}")

# создаем легенду

plt.legend(loc="upper left")

# отрисовка

plt.show()

**Листинг 2. Класс Polynom**

import numpy

# класс полинома

class Polynom:

def \_\_init\_\_(self, root):

self.root = root # корни уравнения

# строковое представление объекта

def \_\_repr\_\_(self):

l = len(self.root)

stra = []

for r in self.root:

l = l - 1

if r:

stra.append(''.join([str(r), f'x^{l}' if l else '']))

return ' + '.join(stra)

def f(self, x):

"""

получение f(x) для любого x

Args:

self: Объект класса.

x: Любой x.

Returns:

res: f(x).

Raises:

"""

l = len(self.root)

res = 0

for r in self.root:

l = l - 1

res += r \* pow(x, l)

return res

def even(self):

"""

четная ли функция

Args:

self: Объект класса.

Returns:

{bool}: Да или нет.

Raises:

"""

test = [i for i in range(1, 10)]

res = [self.f(i) == self.f(-i) for i in test]

if False in res:

return False

return True

def odd(self):

"""

нечетная ли функция

Args:

self: Объект класса.

Returns:

{bool}: Да или нет.

Raises:

"""

test = [i for i in range(1, 10)]

res = [self.f(i) == -(self.f(-i)) for i in test]

if False in res:

return False

return True

def draw(self, interval, step):

"""

нечетная ли функция

Args:

self: Объект класса.

interval: интервал координат

step: шаг

Returns:

[x, y]: Список координат.

Raises:

"""

[intervalL, intervalR] = interval

y = []

yR = []

x = [i for i in numpy.arange(intervalL, intervalR + step, step)]

y = [self.f(i) for i in x]

return [x, y]

**Запуск программы.**

Требуется ввести корни уравнения в виде {1, 2, 3, 4, 5 …}, интервал отрисовки графика и шаг

****

Рисунок 1. Запуск программы

Запускается окно с отрисованным графиком полинома любой степени

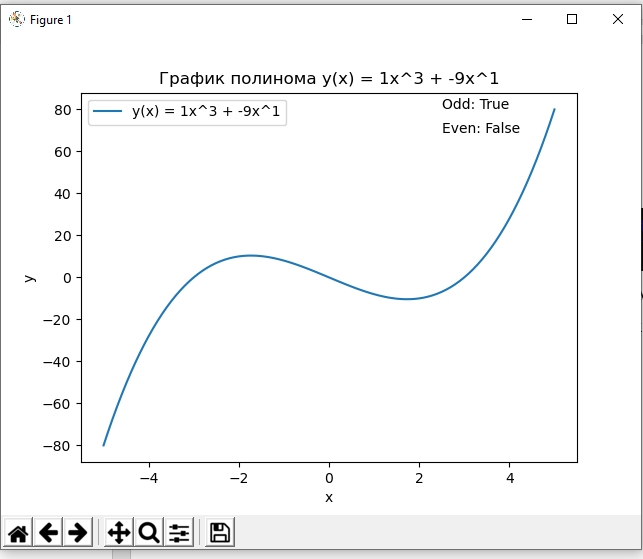


Рисунок 2. Отрисовка графика функции x^3 - 9x