**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕНЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.Шухова»**

**(БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения, вычислительной техники и автоматизированных систем

Лабораторная работа №2

По дисциплине: Системный анализ

Тема: универсальный метод получения случайных величин для произвольного закона распределения случайной величины

Выполнила: студентка группы ВТ-31

Проверил: Полунин А.И.

Белгород 2020

**Цель работы**: изучить универсальный метод получения случайной величины для произвольного закона распределения случайной величины, реализовать датчик случайных чисел по заданному закону распределения.

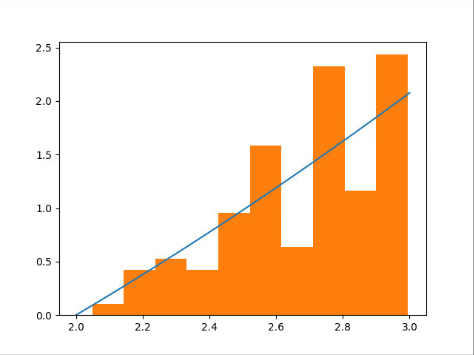
Проведем исследование функции, и выясним, что она принимает положительные значения на промежутках . Пусть a = 2, b = 3

1. Чтобы получить функцию распределения случайной величины из имеющейся функции, нормируем ее по формуле

Тогда функция плотности вероятности будет иметь вид:

2. Для получения случайной величины, необходимо решить такое уравнение: где v – случайная величина, распределенная по равномерному закону, ξ – требуемая случайная величина, распределенная по требуемому закону. Для применения численного метода решения (будет применен метод Ньютона) преобразуем к виду:

# Результаты работы программы



**Вывод**: я изучила универсальный метод получения случайной величины и реализовала датчик случайных чисел по заданному закону распределения.

# Код программы

sz = 100

a = 2

b = 3

result = []

arr = []

def fx(x):

return x\*\*2 + 4 \* x - 12

def gx(x):

return (x\*\*2 + 4 \* x - 12) / k

def prob\_num(xi,v):

s1 = integrate.quad(fx, a, xi)[0]

return (s1/k - v)

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

k = integrate.quad(fx, a, b)[0]

for i in range(0, sz):

arr.append(numpy.random.random\_sample())

for i in arr:

def t(x):

return prob\_num(x,i)

solve = optimize.fsolve(t, b)

result.append(solve[0])

numpy.round(solve, 3, solve)

x\_axis = numpy.linspace(a, b)

fig, ax = pyplot.subplots(1, 1)

pyplot.plot(x\_axis, numpy.apply\_along\_axis(gx, 0, x\_axis))

pyplot.hist(result, 10, density=True)

pyplot.show()