# УПРАЖНЕНИЯ

1. Напишите программу, которая бесконечно считывает коэффициенты квадратного уравнения и считает его корни. Выведите уравнение на консоль и обработайте исключения.

# Импортируем функцию sqrt() (Вычисление корня) и модуля math  
from math import sqrt  
  
  
# Функция для ввода аргументов, которая принимает имя вводимого аргумента  
def get\_coefficient(name):  
 # Проверяем, ввели ли нам число  
 try:  
 # Считываем значение коэффициента и приводим его к типу int  
 coefficient = int(input(name + ' = '))  
 return coefficient  
   
 # Если приведение не удалось выбрасываем исключение  
 except ValueError:  
 raise Exception('Вы ввели не число')  
  
  
# Метод, который выводит каждое отдельное слагаемое  
# need\_plus - аргумент, отвечающий за то, надо ли выводить '+', так как перед первым коэффициентом выводить его не нужно  
# need\_one - если коэффициент равен 1, то выводить его надо только в том случае, если без x  
def print\_coefficient(coefficient, str\_x, need\_plus=True, need\_one=False):  
 if coefficient != 0:  
 if coefficient < 0:  
 print('-', end=' ')  
 elif need\_plus:  
 print('+', end=' ')  
 abs\_coef = abs(coefficient)  
 if abs\_coef != 1 or need\_one:  
 print(abs\_coef, end='')  
 print(str\_x, end=' ')  
  
  
# После введения аргументов выводим получившееся уравнение на консоль  
def print\_equation(a, b, c):  
 print\_coefficient(a, 'x^2', need\_plus=False)  
 print\_coefficient(b, 'x')  
 print\_coefficient(c, '', need\_one=True)  
  
 print('= 0')  
  
  
# Функция, возвращающая значение дискриминанта  
def get\_d(a, b, c):  
 return b \*\* 2 - 4 \* a \* c  
  
  
# Метод, который выводит дискриминант на консоль  
def print\_d(d):  
 print('D =', d)  
  
  
# Функция, возвращающая значение корней  
def get\_roots(a, b, d):  
 # Если дискриминант больше 0 - считаем оба значения   
 if d > 0:  
 x1 = (- b + sqrt(d)) / (2 \* a)  
 x2 = (- b - sqrt(d)) / (2 \* a)  
 return x1, x2  
 # Если дискриминант меньше 0 - выбрасываем исключение о невозможности расчётов  
 elif d < 0:  
 raise Exception('Корней на множестве действительных чисел не существует')  
 # Если дискриминант равен 0 - возвращаем два одинаковых значения   
 else:  
 x = - b / (2 \* a)  
 return x, x  
  
  
# Метод, который выводит значение корней  
def print\_roots(x1, x2, d):  
 # Если дискриминант равен 0 - выводим два одинаковых значения   
 if d == 0:  
 print('x1 = x2 =', x1)  
 # Иначе - выводим два значения  
 else:  
 print('x1 =', x1)  
 print('x2 =', x2)  
  
  
# Метод для всех расчётов  
def main():  
 # Вводим коэффициенты  
 a = get\_coefficient('a')  
 # Выбрасываем исключение, если a==0, так как в таком случае получаем неквадратное уравнение  
 if a == 0:  
 raise Exception('Коэффициент a не может равняться 0')  
 b = get\_coefficient('b')  
 c = get\_coefficient('c')  
 # Выводим уравнение  
 print\_equation(a, b, c)  
  
 # Считаем дискриминант  
 d = get\_d(a, b, c)  
 # Выводим дискриминант  
 print\_d(d)  
   
 # Считаем x1 и x2  
 x1, x2 = get\_roots(a, b, d)  
 # Выводим x1 и x2  
 print\_roots(x1, x2, d)  
  
  
# Запускаем бесконечный цикл  
while True:  
 try:  
 main()  
 # Если внутри метода main() произошло исключение - выводим его текст на консоль   
 # и говорим о том, что не смогли досчитать до конца  
 except Exception as err:  
 print(err)  
 print('Не смогли досчитать до конца')  
  
 print()

Вывод на консоль:

a = **1**

b = **-5**

c = **6**

x^2 - 5x + 6 = 0

D = 1

x1 = 3.0

x2 = 2.0

a = **1**

b = **4**

c = **4**

x^2 + 4x + 4 = 0

D = 0

x1 = x2 = -2.0

a = **1**

b = **1**

c = **1**

x^2 + x + 1 = 0

D = -3

Корней на множестве действительных чисел не существует

Не смогли досчитать до конца

a = **0**

Коэффициент a не может равняться 0

Не смогли досчитать до конца

a = **1**

b = **2**

c = **три**

Вы ввели не число

Не смогли досчитать до конца

1. Напишите программу, которая считывает размер вклада , годовой процент и количество лет , на которое делается вклад, и считает, какая сумма получится через заданный промежуток времени (Каждый год размер вклада увеличивается на и деньги прибавляются к сумме вклада, на них в следующем году тоже будут начислены проценты).

# Функция, возвращающая количество денек, накопившихся за years лет  
def bank(x, years, percent):  
 # Если количество лет == 0 - возвращаем количество  
 if years == 0:  
 return x  
   
 # Прибавляем к вкладу percent процентов, уменьшаем год и снова вызываем функцию  
 return bank(x + x \* percent, years - 1, percent)  
  
  
def main():  
 x = input('Размер вклада: ')  
 percent = input('Процент (%): ')  
 years = input('Срок (в годах): ')  
 print('Если вы вложите', x, 'руб.', end=' ')  
 print('под', percent + '% годовых,', end=' ')  
 print('то через ' + years + ' лет', end=' ')  
 print('получите', bank(int(x), int(years), int(percent) / 100), 'руб.')  
  
  
while True:  
 main()  
 print()

Вывод на консоль:

Размер вклада: 12000

Процент (%): 7

Срок (в годах): 9

Если вы вложите 12000 руб. под 7% годовых, то через 9 лет получите 22061.510549041857 руб.

Размер вклада: 100000

Процент (%): 9

Срок (в годах): 5

Если вы вложите 100000 руб. под 9% годовых, то через 5 лет получите 153862.39549 руб.

1. Напишите программу, которая считывает координаты 2-ух точек и выводит на консоль уравнение прямой, проходящей через эти точки. Обработайте исключения.
2. Напишите функцию, которая принимает число n и выводит на экран n чисел Фибоначчи. Первые 2 числа в последовательности чисел Фибоначчи – 0 и 1, а все последующие получаются путем складывания двух последних. Например, 6 чисел Фибоначчи:

0, 1, 1 (1+0), 2 (1+1), 3 (1+2), 5 (2+3)

1. Напишите функцию, которая принимает число n и выводит на экран n-ое простое число. Например, 5-ое простое число: 11.
2. Напишите функцию, которая принимает 3 аргумента: день, месяц и год; и проверяет, существует ли такая дата в календаре.
3. Напишите функцию, которая принимает список, состоящий из чисел, и число, а возвращает список той же длины, который заполнен значениями: True – если элемент списка меньше переданного числа, False – иначе. Выведите список на консоль. Например, передаем в метод список [3, 4, 5, 2, 1, 0] и число 4, вывод должен быть:

[True, False, False, True, True, True]