Выполнил: студент группы ПР-21.102 Саая

ПР – 2

Индивидуальные:

1. Даны два целых числа А и В (А < В). Вывести в порядке возрастания

Все целые числа, расположенные между А и В (включая сами числа

А и В), а также количество N этих чисел.

a = int(input("Введите число A: "))

b = int(input("Введите число B: "))

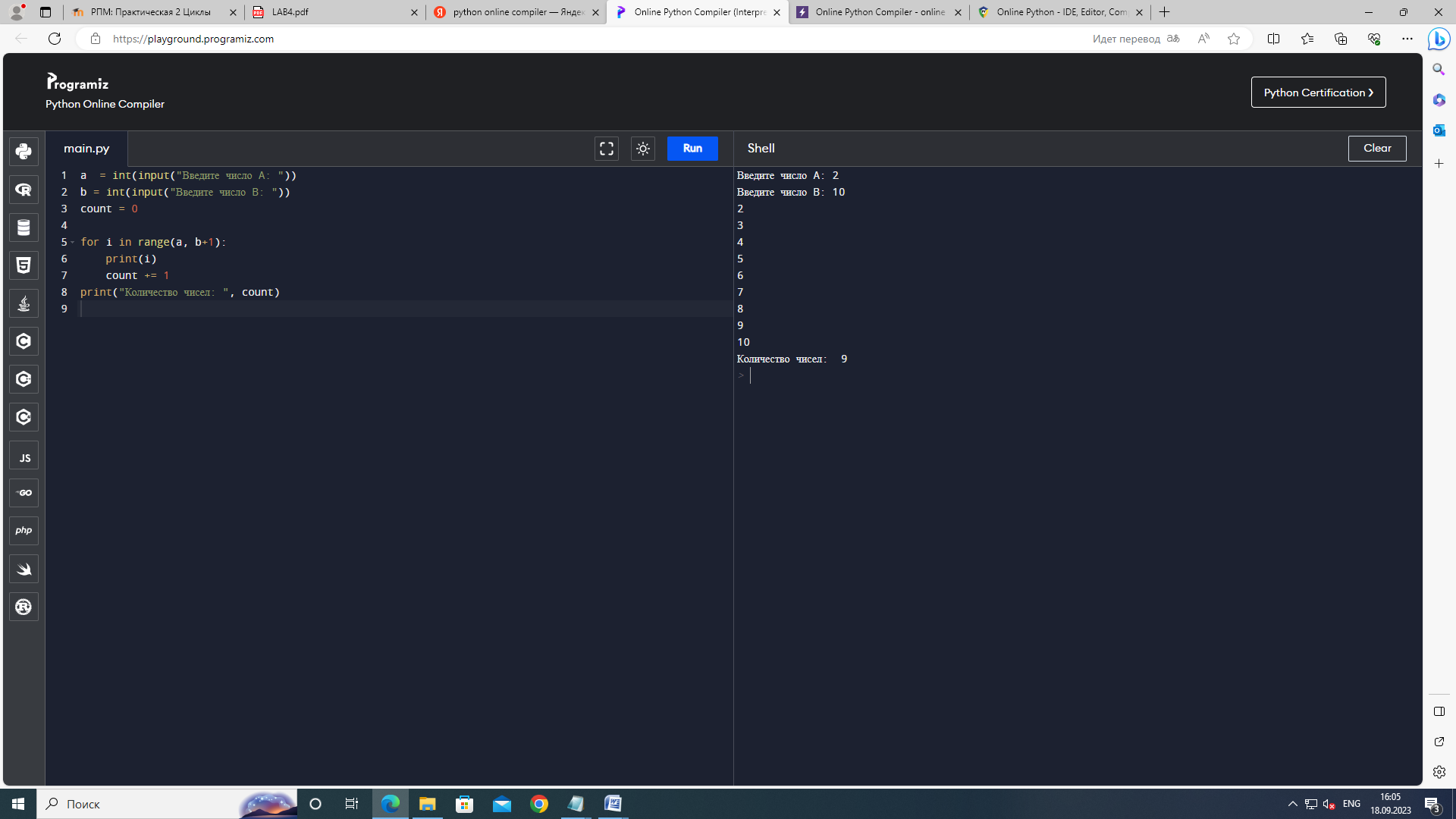
count = 0

for i in range(a, b+1):

print(i)

count += 1

print("Количество чисел: ", count)



1. Прямоугольник на плоскости a<x<b; c<y<d задается четырьмя

Числами (его габаритами): a, b, с, d. Последовательно вводятся

Габариты n прямоугольников. В процессе ввода находить площадь их

Пересечения, не запоминая самих габаритов.

def intersection\_area(rectangles):

minx = max(rectangles, key=lambda r: r[0])[0]

maxx = min(rectangles, key=lambda r: r[1])[1]

miny = max(rectangles, key=lambda r: r[2])[2]

maxy = min(rectangles, key=lambda r: r[3])[3]

width = maxx - minx

height = maxy - miny

if width < 0 or height < 0:

return 0

return width \* height

n = int(input("Введите количество прямоугольников: "))

rectangles = []

for i in range(n):

a = float(input("Введите a для прямоугольника {}: ".format(i+1)))

b = float(input("Введите b для прямоугольника {}: ".format(i+1)))

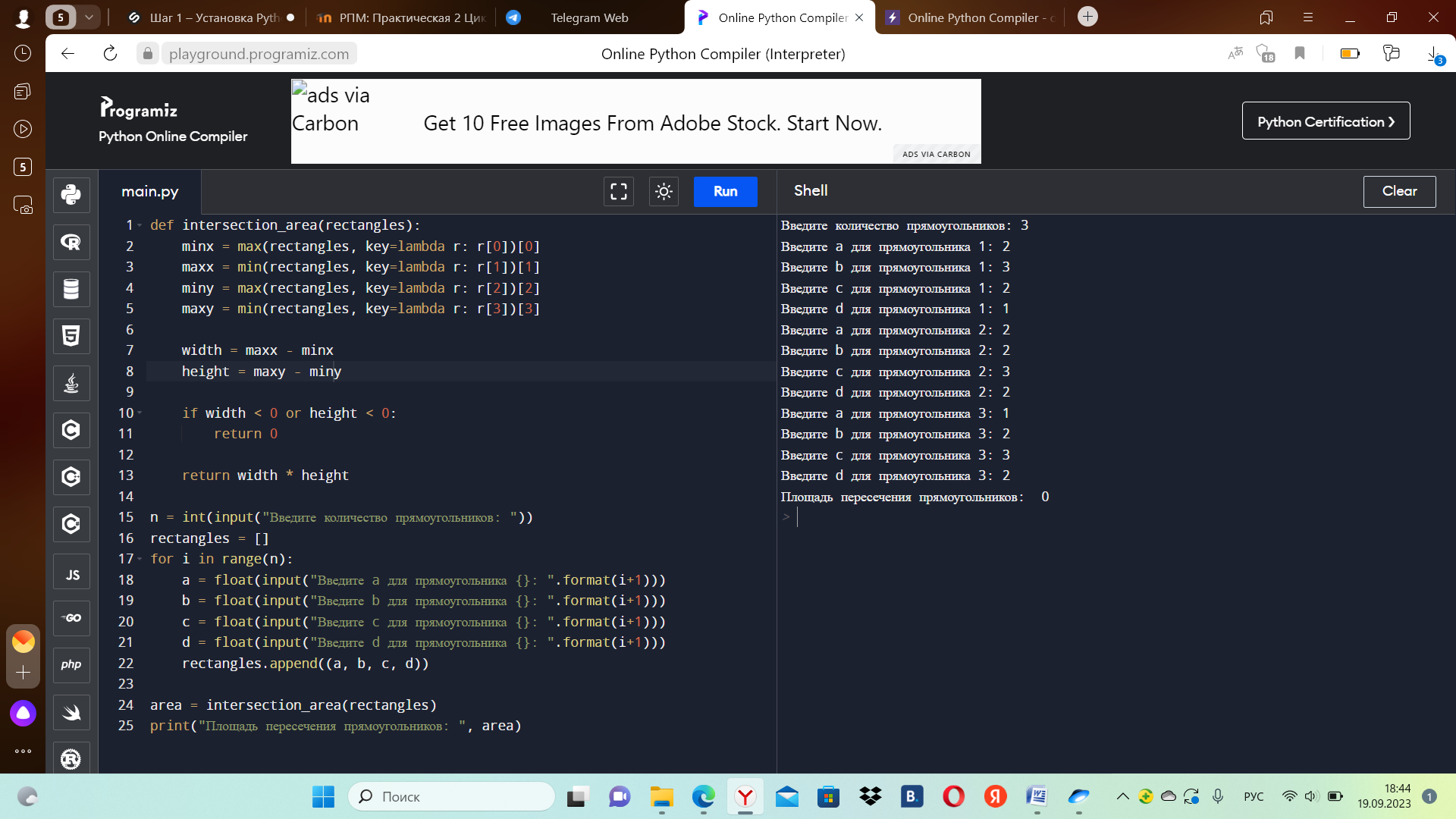
c = float(input("Введите c для прямоугольника {}: ".format(i+1)))

d = float(input("Введите d для прямоугольника {}: ".format(i+1)))

rectangles.append((a, b, c, d))

area = intersection\_area(rectangles)

print("Площадь пересечения прямоугольников: ",area)



Общие задания:

1. На тренировке спортсмен ежедневно пробегает некоторую дистанцию, с каждым днём увеличивая её на 10%. Составить программу, определяющую по расстоянию, преодолённому спортсменом в первый день тренировки, длину дистанции на k-й день;

def calcul(first, perc, k):

dist = first

for i in range(1, k):

dist += dist \* (perc / 100)

return dist

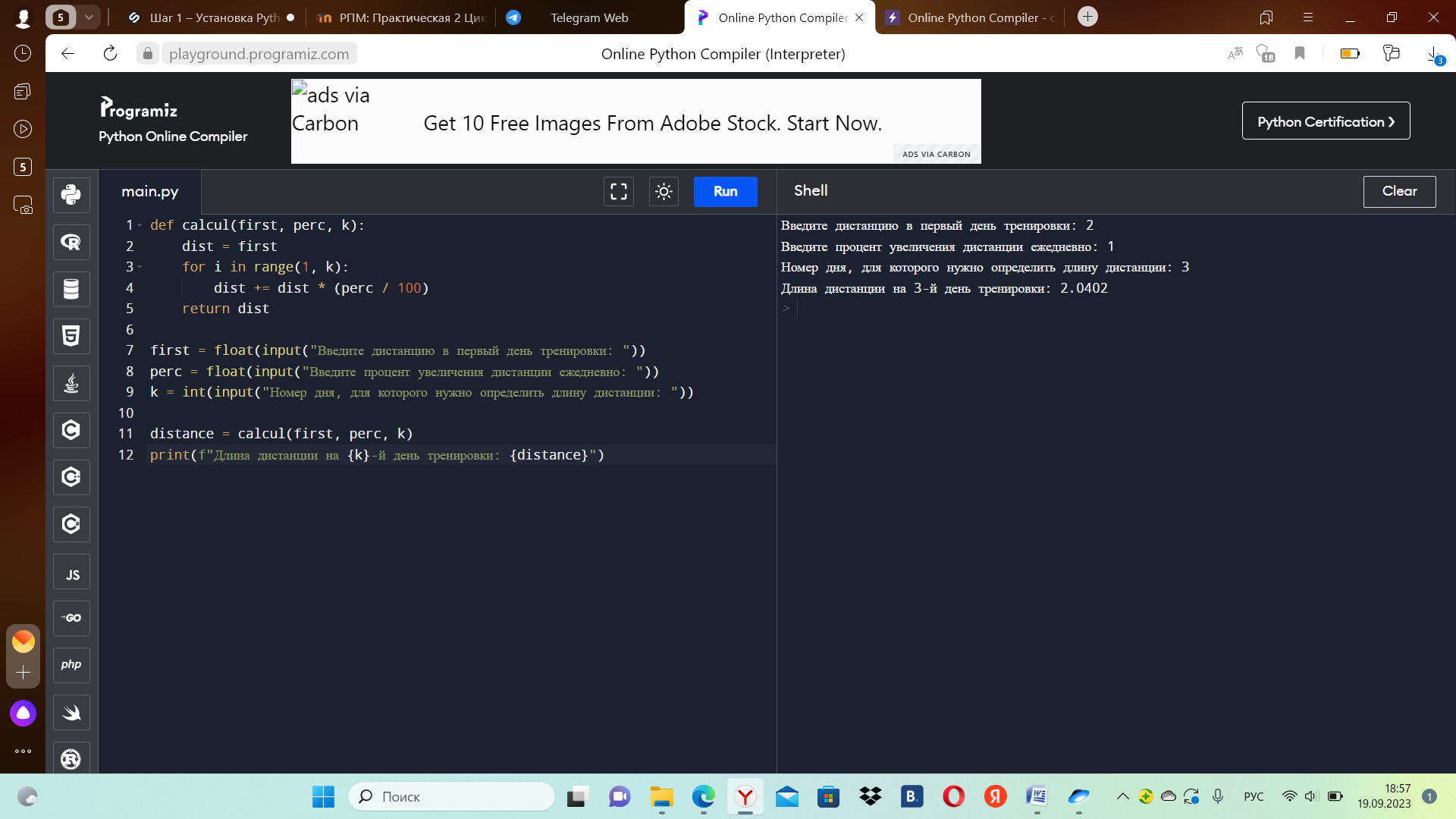
first = float(input("Введите дистанцию в первый день тренировки: "))

perc = float(input("Введите процент увеличения дистанции ежедневно: "))

k = int(input("Номер дня, для которого нужно определить длину дистанции: "))

distance = calcul(first, perc, k)

print(f"Длина дистанции на {k}-й день тренировки: {distance}")



1. Перевести введённое пользователем десятичное число в двоичное. Известно, что число меньше 256.

number = []

while True:

num = int(input("Введите целое число(0 для завершения):"))

if num == 0 :

break

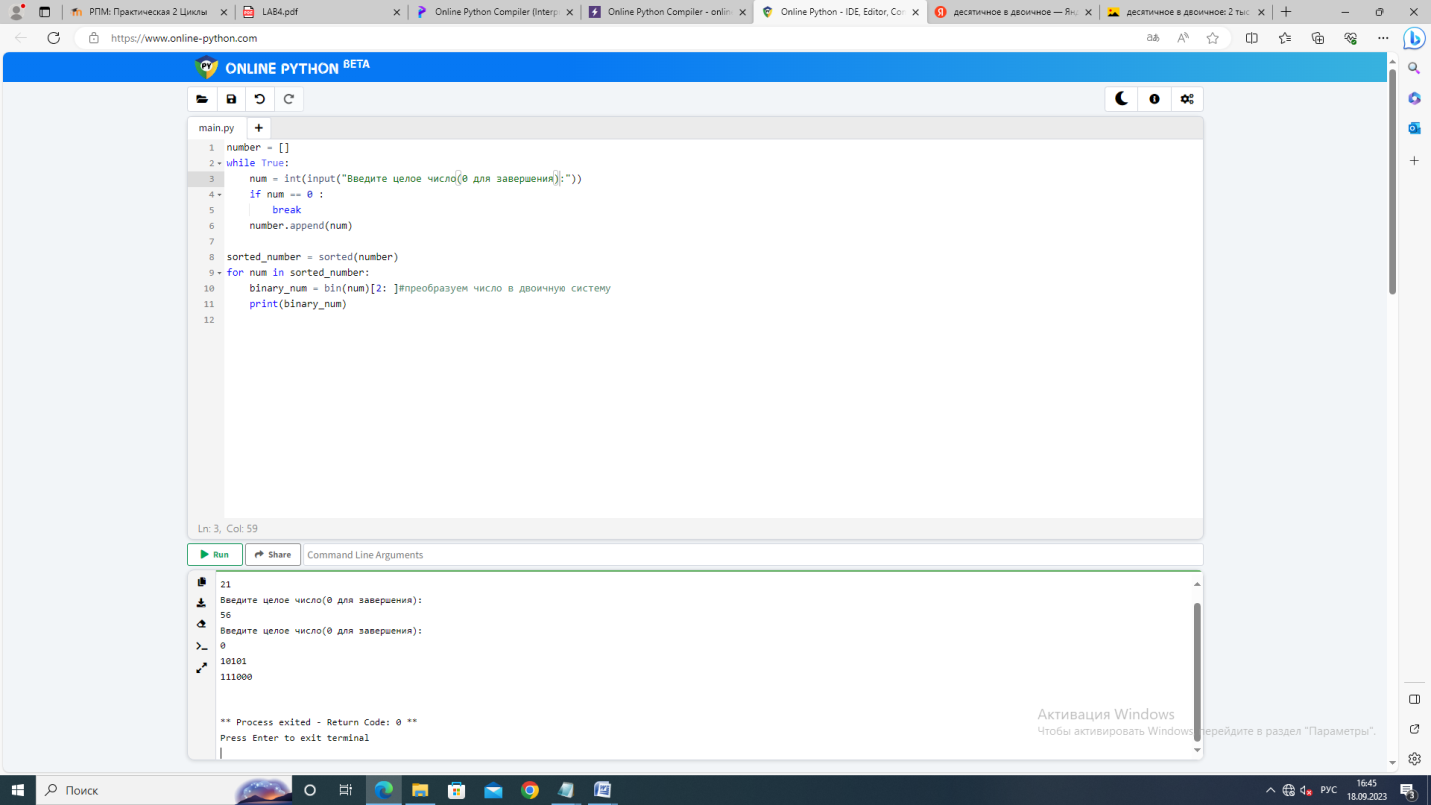
number.append(num)

sorted\_number = sorted(number)

for num in sorted\_number:

binary\_num = bin(num)[2: ]#преобразуем число в двоичную систему

print(binary\_num)



1. Разложить натуральное число на простые множители.

def facnum(n):

fact = []

i = 2

while i <= n:

if n % i == 0:

fact.append(i)

n = n // i

else:

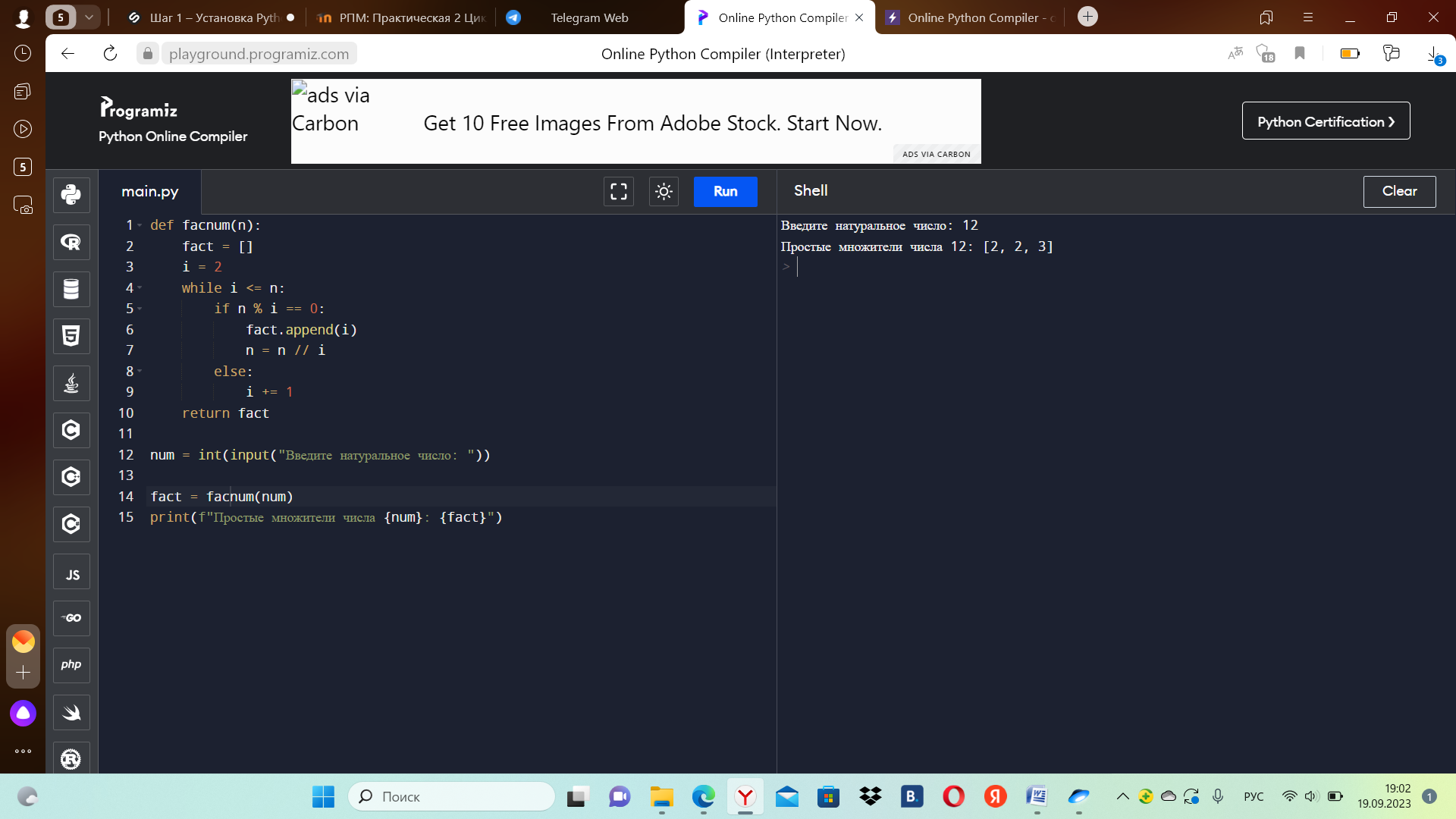
i += 1

return fact

num = int(input("Введите натуральное число: "))

fact = facnum(num)

print(f"Простые множители числа {num}: {fact}")



1. Население города на 2021 г. насчитывало 620 тыс. человек. Считая темп прироста населения за год равным 3,7%, определить, в каком году оно превысит 1,5 млн человек.

popul = 620000

rate = 0.037

tarpopul = 1500000

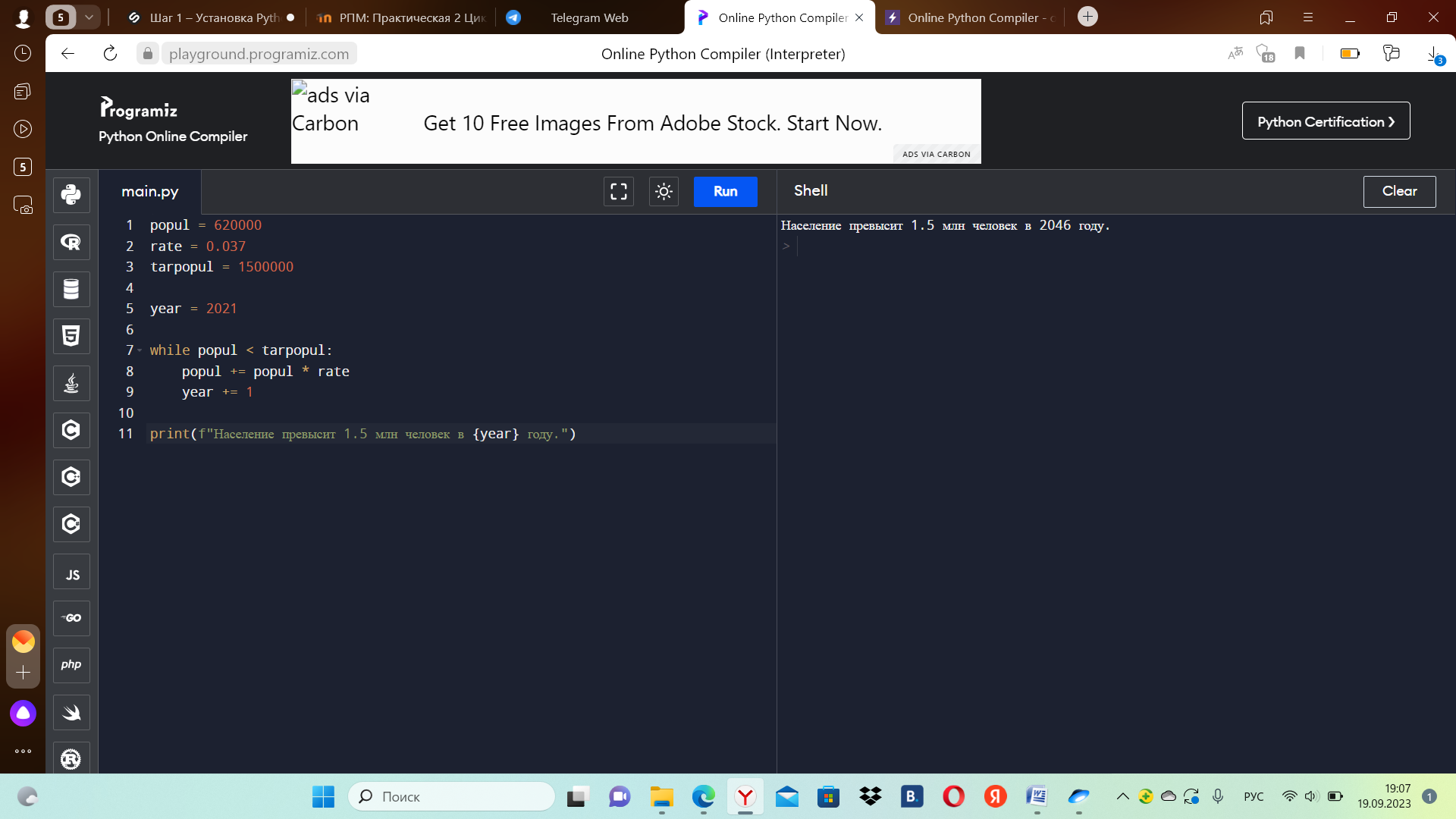
year = 2021

while popul < tarpopul:

popul += popul \* rate

year += 1

print(f"Население превысит 1.5 млн человек в {year} году.")



1. Найти сумму нечётных делителей введённого с клавиатуры натурального числа.

num = int(input("Введите натуральное число: "))

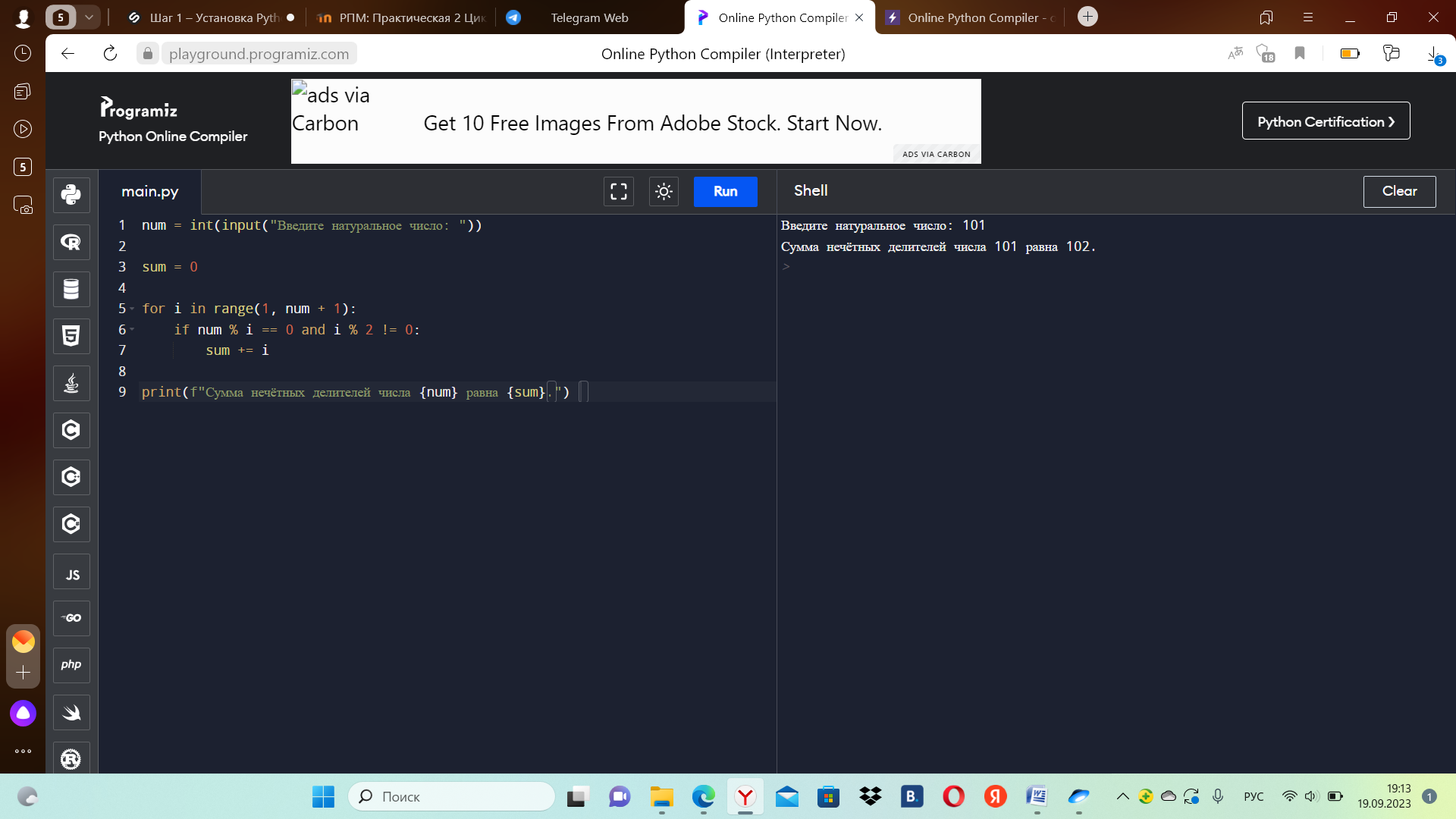
sum = 0

for i in range(1, num + 1):

if num % i == 0 and i % 2 != 0:

sum += i

print(f"Сумма нечётных делителей числа {num} равна {sum}.")



1. Найти все натуральные числа из отрезка [1; 200], у которых количество делителей равно n (где n вводится с клавиатуры)

n = int(input("Введите количество делителей: "))

for num in range(1, 201):

count = 0

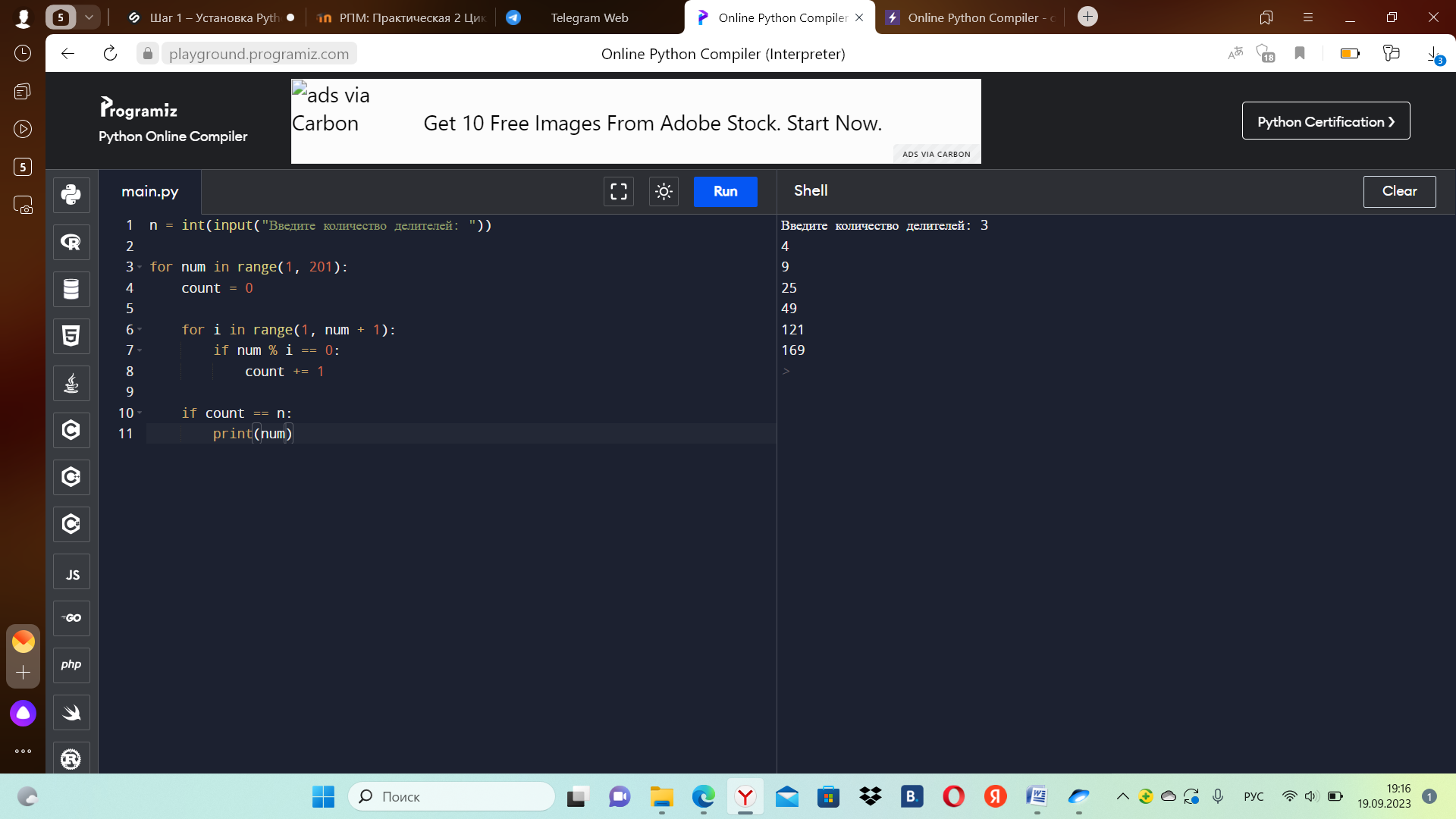
for i in range(1, num + 1):

if num % i == 0:

count += 1

if count == n:

print(num)



1. Найти все четырёхзначные числа, у которых сумма крайних цифр равна сумме средних (например, 3221).

for num in range(1000, 10000):

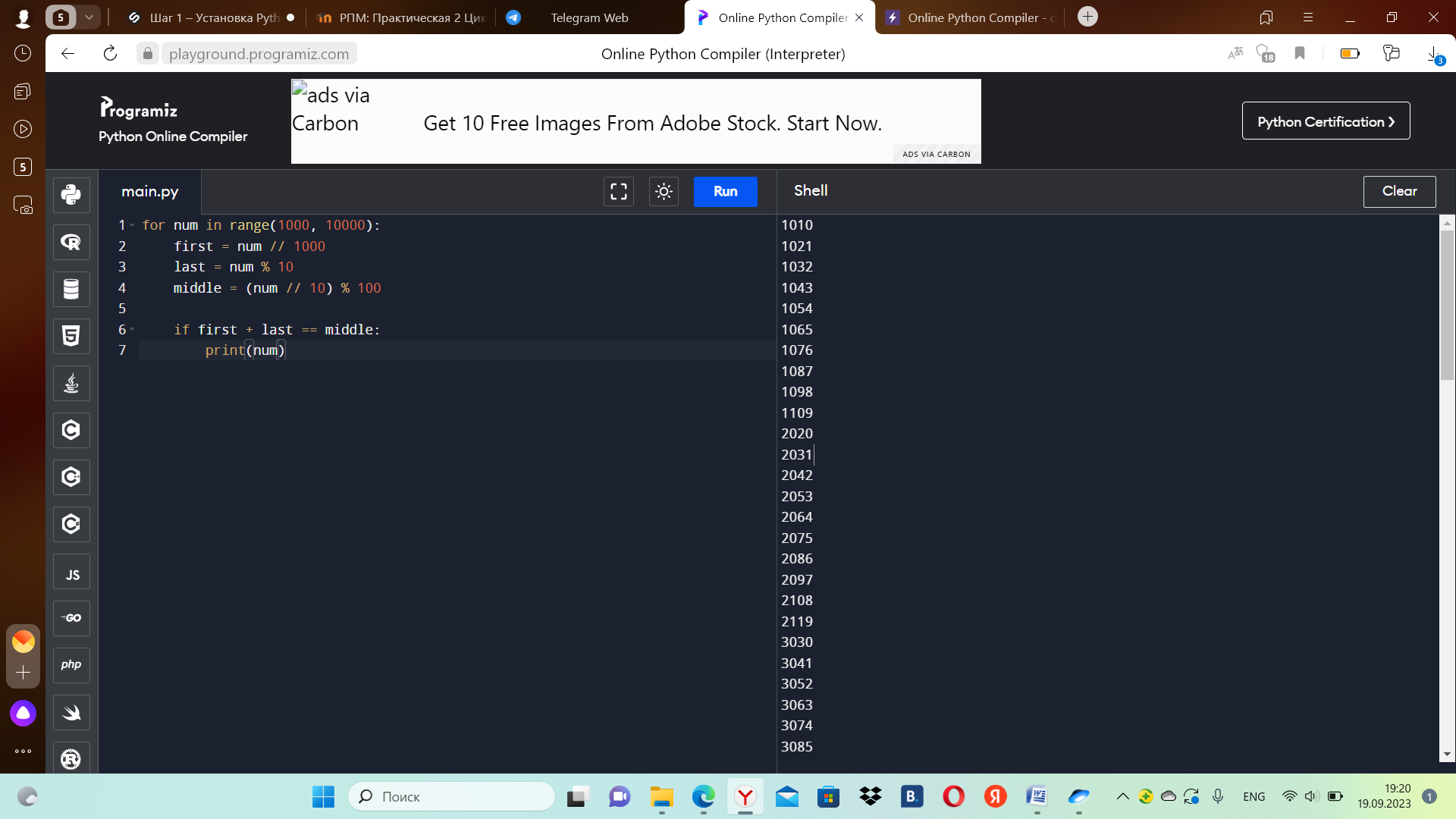
first = num // 1000

last = num % 10

middle = (num // 10) % 100

if first + last == middle:

print(num)

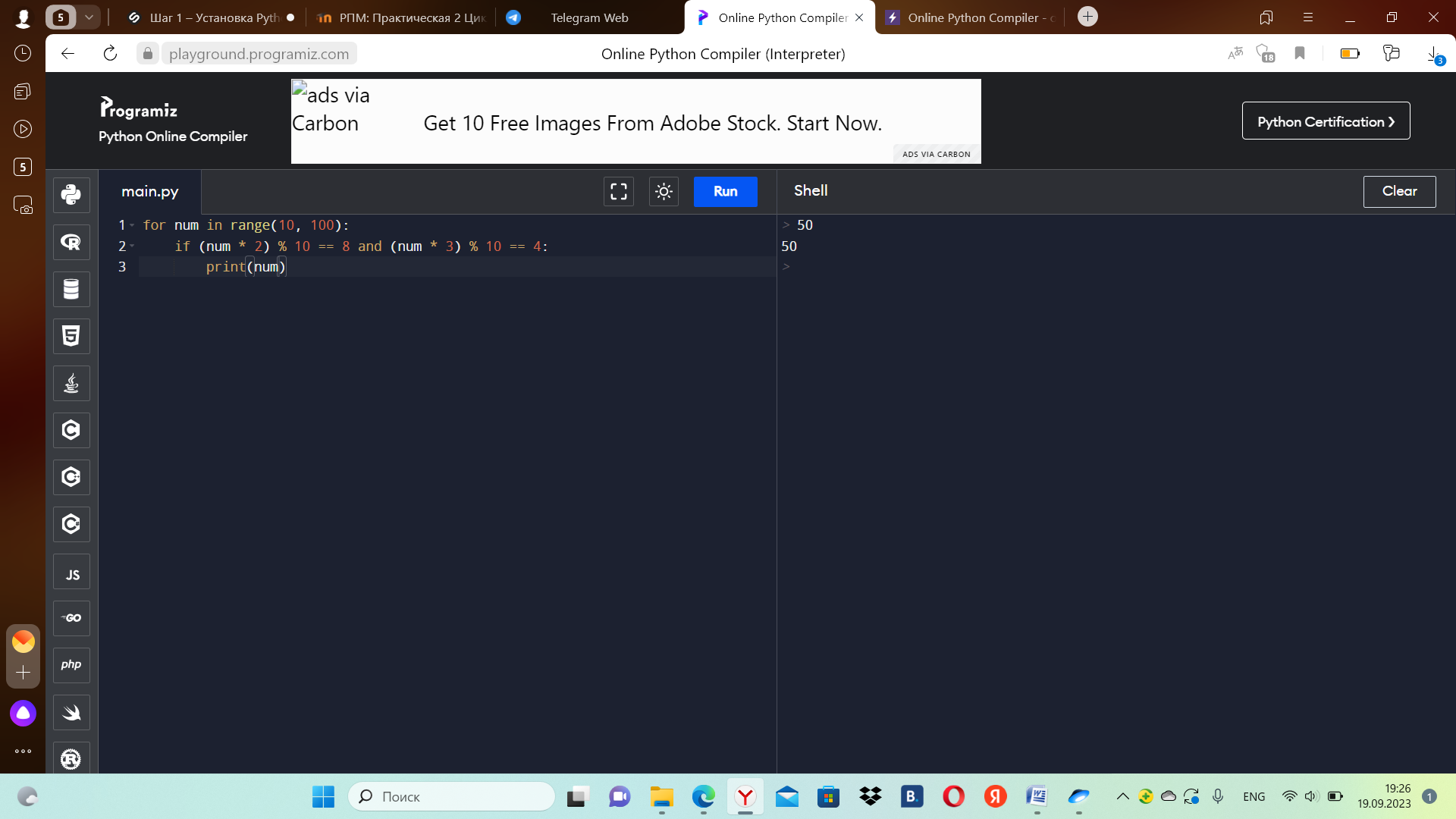


1. Найти все двухзначные числа, которые при умножении на 2 заканчиваются на 8, а при умножении на 3 – на 4.

for num in range(10, 100):

if (num \* 2) % 10 == 8 and (num \* 3) % 10 == 4:

print(num)



1. В данных задачах дана функция и ее разложение в ряд или произведение. Необходимо численно убедиться в справедливости равенства, для чего для заданного значения аргумента х вычислить левую его часть и разложение, стоящее в правой части, с заданной погрешностью е. Испытать разложение на сходимость при разных значениях аргумента, оценить скорость сходимости, для чего вывести число итераций n (слагаемых или сомножителей), необходимых для достижения заданной точности. В некоторых задачах указан интервал допустимых значений аргумента х, при которых сходимость гарантируется.

def nlog(x, epsilon):  
 if x <= 0:  
 return None # Логарифм определен только для положительных аргументов  
  
 n = 1  
 sum = 0  
  
 while True:  
 form = (2 \* (x - 1) / (x + 1)) \*\* (2 \* n - 1) / (2 \* n - 1)  
 sum += form  
  
 if abs(form) < epsilon:  
 break  
  
 n += 1  
  
 return 2 \* sum  
  
# Заданный аргумент x и погрешность eps  
x = 2.0  
eps = 0.01  
  
result = nlog(x, eps)  
  
if result is not None:  
 print(f"ln({x}) приближенно равен {result:.6f} с погрешностью {eps:.1e}")  
else:  
 print("x должен быть >0")

