МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КПІ ім. Ігоря Сікорського»

Кафедра АПЕПС

Лабораторна робота №6

Варіант №7

Виконав:

студент 2-го курсу

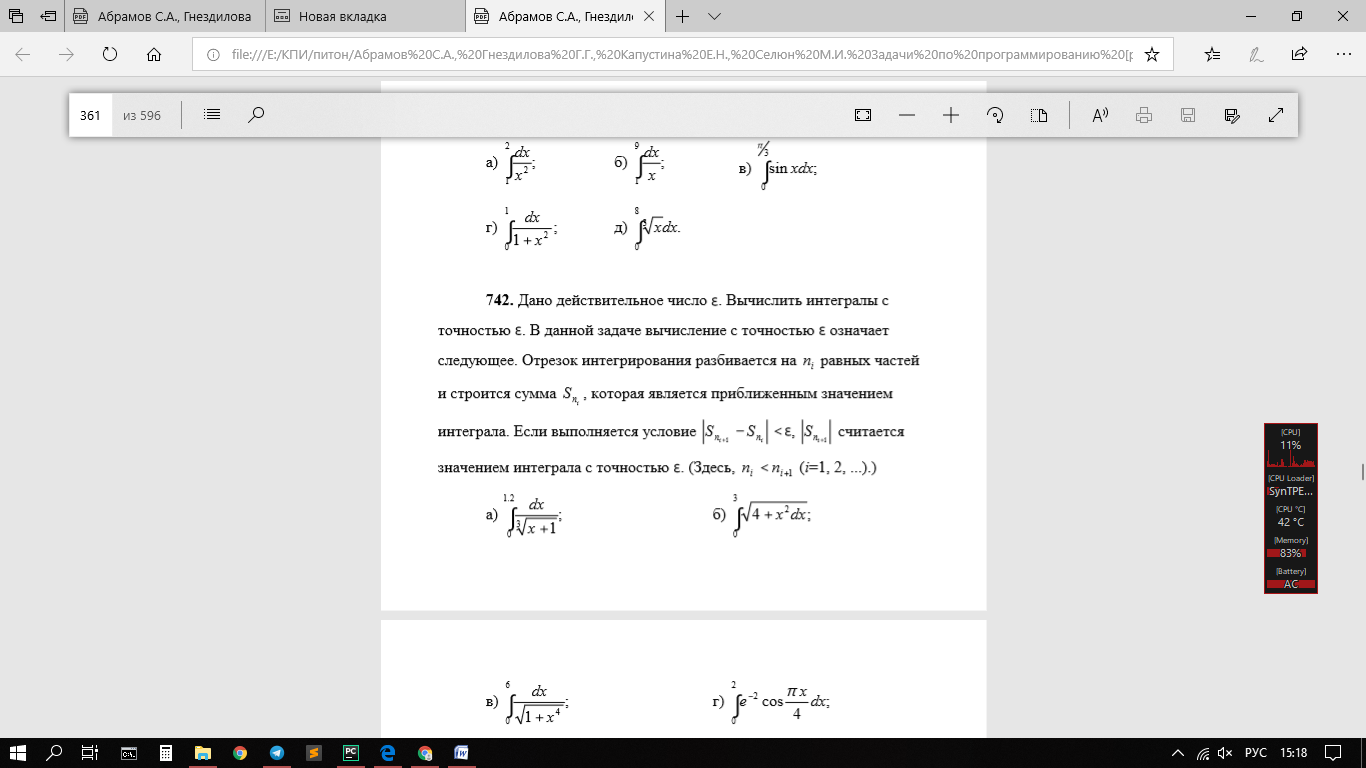
ТЕФ, гр. ТІ-72

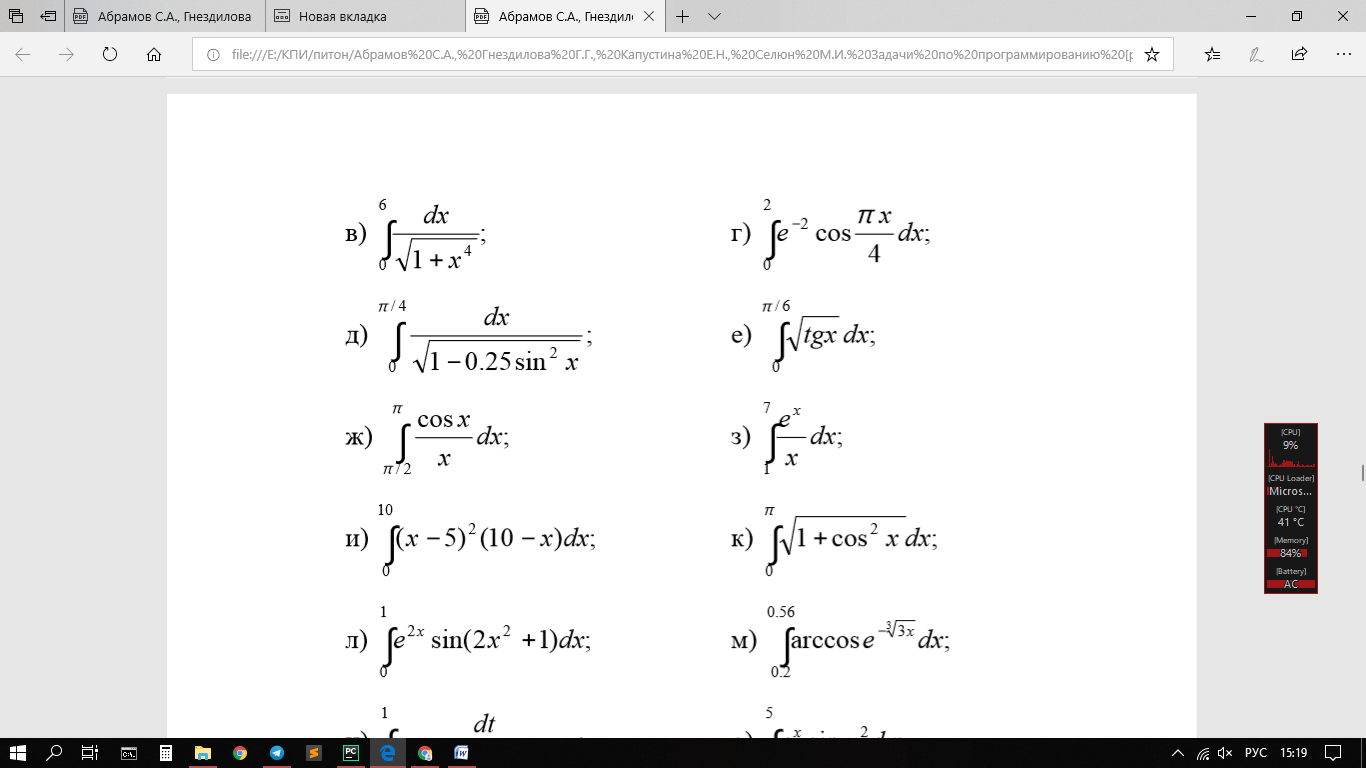
Головачук С. В.

Перевірив:

Сігайов А. О.

Київ-2018

**Хід виконання:**

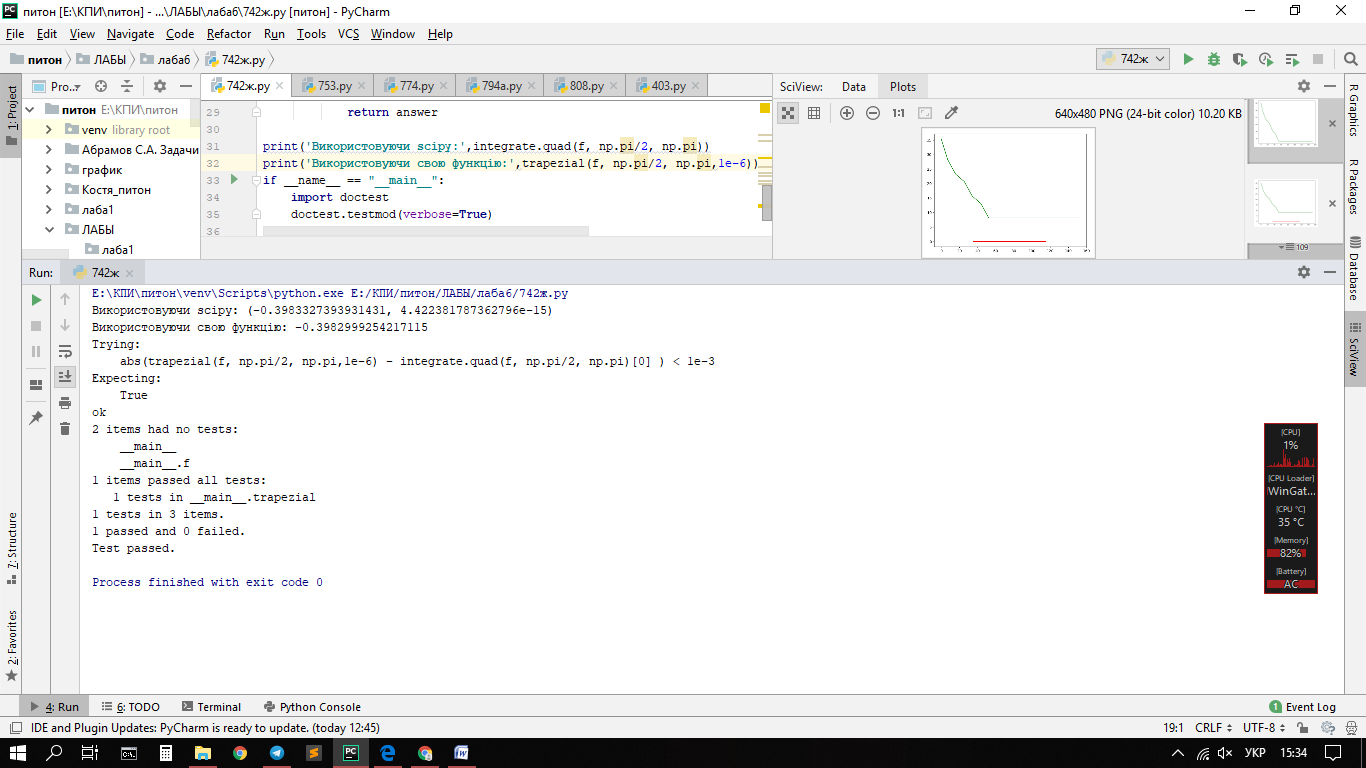


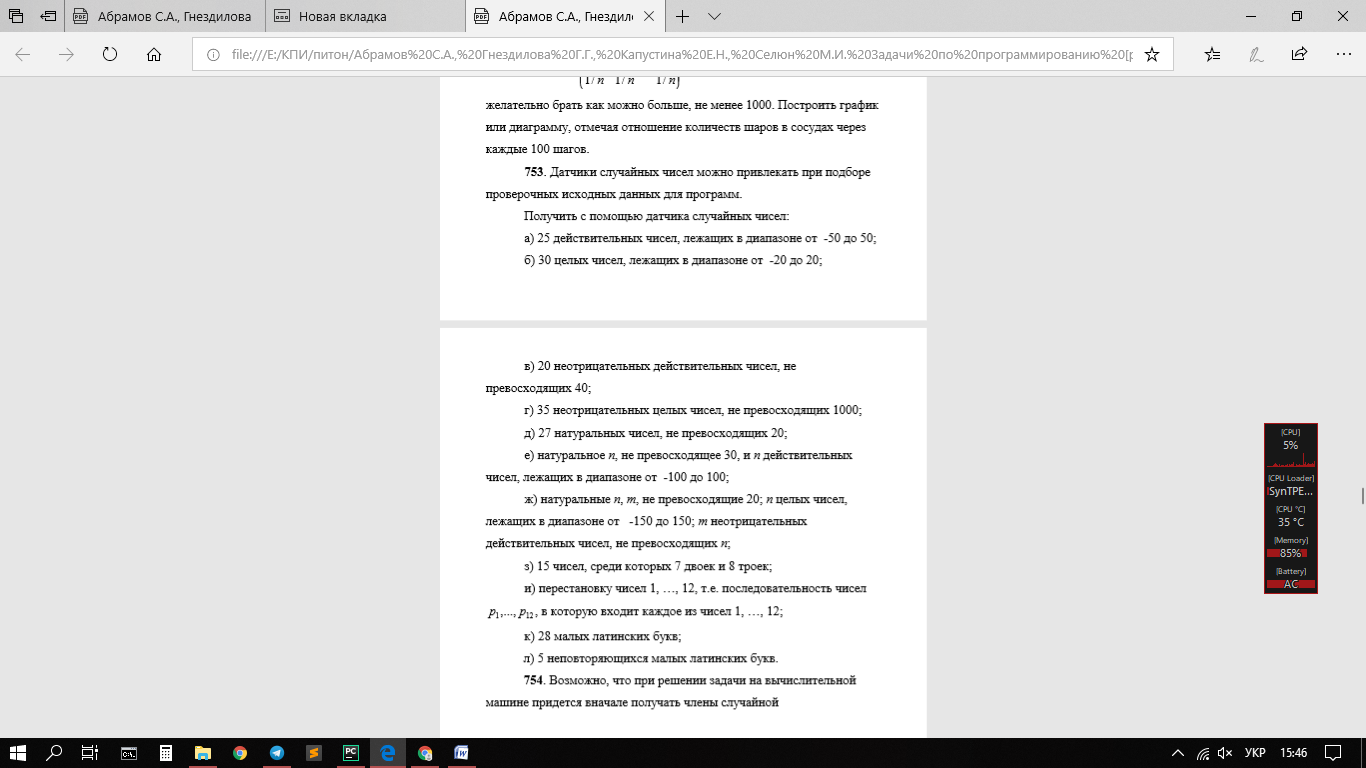
*"""  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*\*\* Лабораторна робота №6 \*\*\*  
\*\*\* з курсу "Основи програмування \*\*\*  
\*\*\* на скриптових мовах" \*\*\*  
\*\*\* Варіант №7 \*\*\*  
\*\*\* Виконав:студент Головачук С.В. Група ТІ-72 \*\*\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
"""*

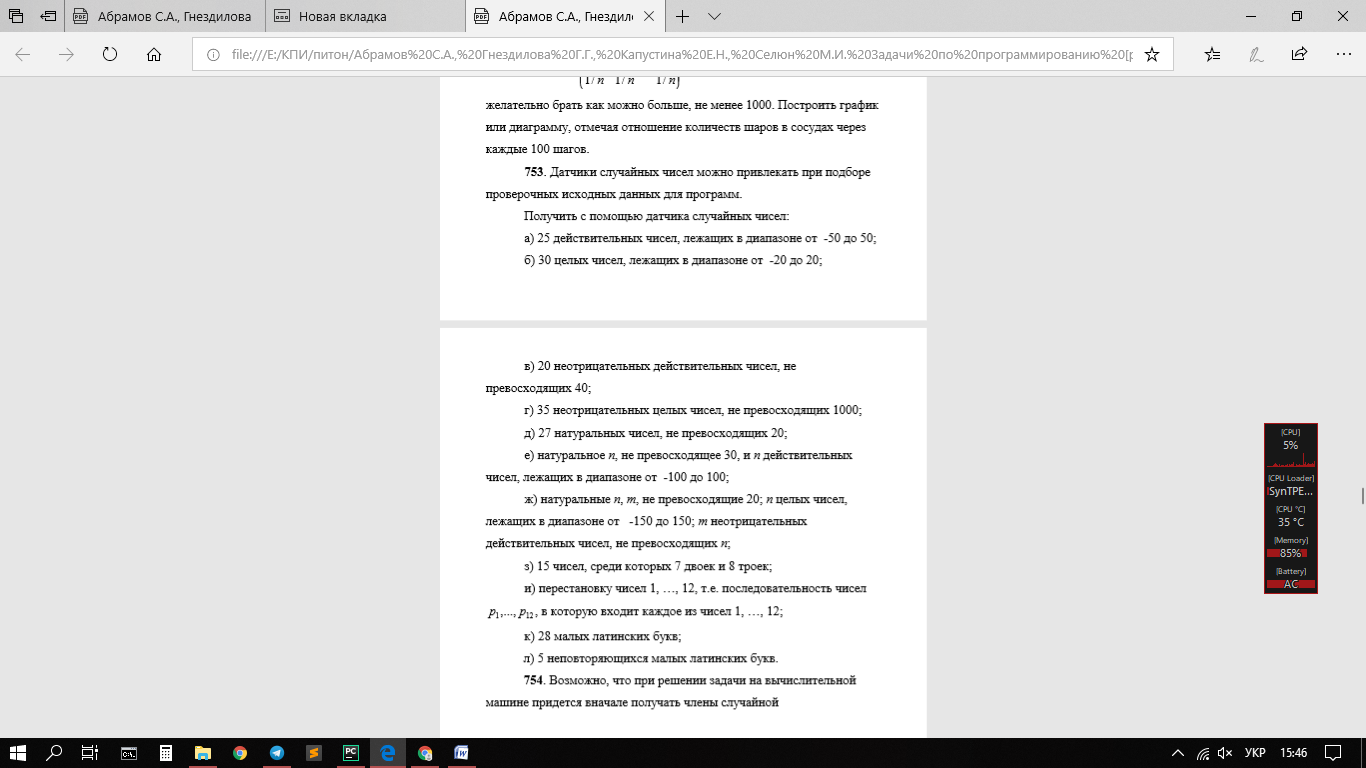
**import** sympy **as** sp  
**from** scipy **import** integrate  
**import** numpy **as** np  
**import** math  
x = sp.Symbol(**'x'**)  
**def** f(x):  
 **return** math.cos(x)/x  
**def** trapezial(f, a, b, eps):  
 *"""  
 Calculates definite integral by trapezoid method.* **:param** *f:function, a:lower limit, b:upper limit, eps:tolerance* **:return***: int:integral  
 :Tests:  
 >>> abs(trapezial(f, np.pi/2, np.pi,1e-6) - integrate.quad(f, np.pi/2,*

*np.pi)[0] ) < 1e-3  
 True  
 """* answer=(f(a) + f(b))/2 \* (b-a) *#рахуємо площу суцільної трапеції* n = 1  
 **while True**: *#уточнюємо до eps* n+=1  
 old = answer  
 h = (b - a) / n *#розділяємо на n трапецій* answer = (f(a) + f(b))/2 *#середня лінія трапеції* **for** i **in** range(1, n):  
 answer += f(a + i\*h)  
 answer \*= h  
 **if** abs(old - answer) < eps:  
 **return** answer

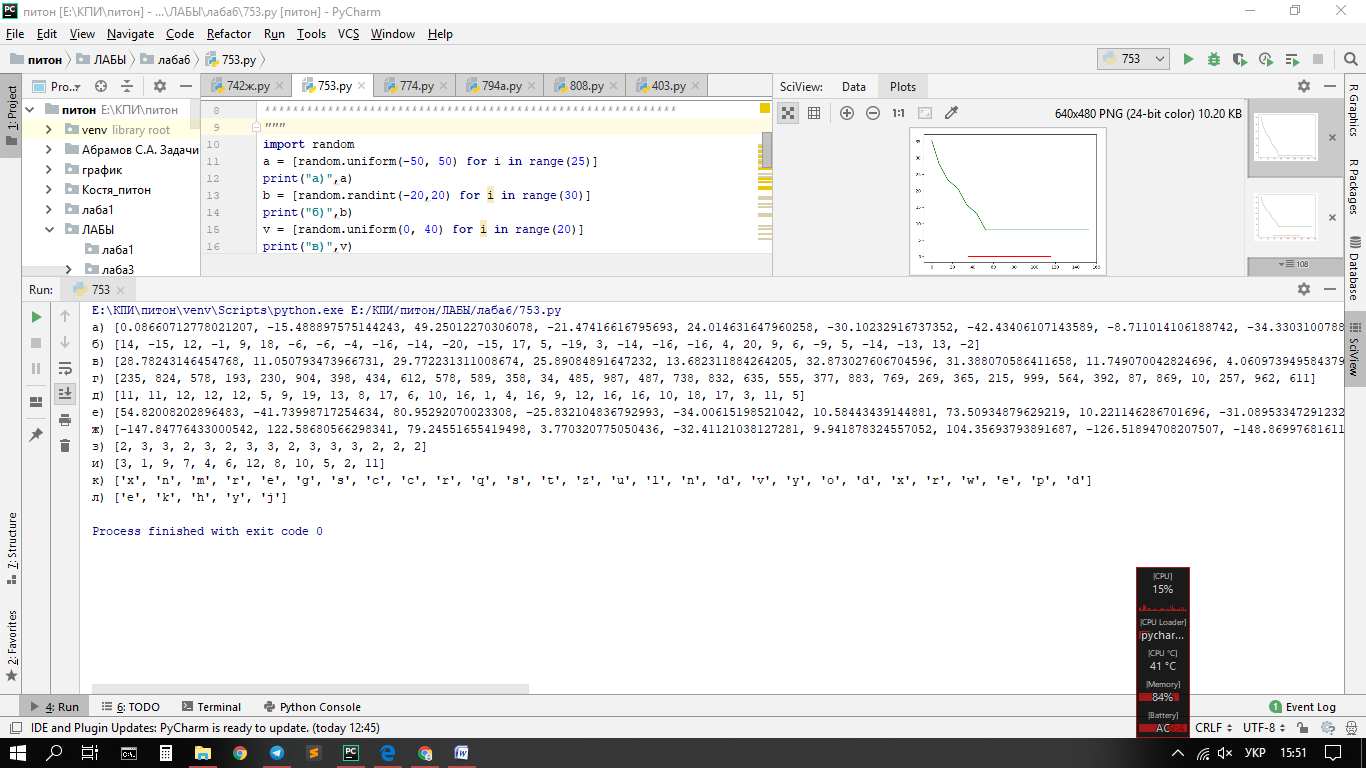
print(integrate.quad(f, np.pi/2, np.pi))  
print(trapezial(f, np.pi/2, np.pi,1e-6))  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 **import** doctest  
 doctest.testmod(verbose=**True**)

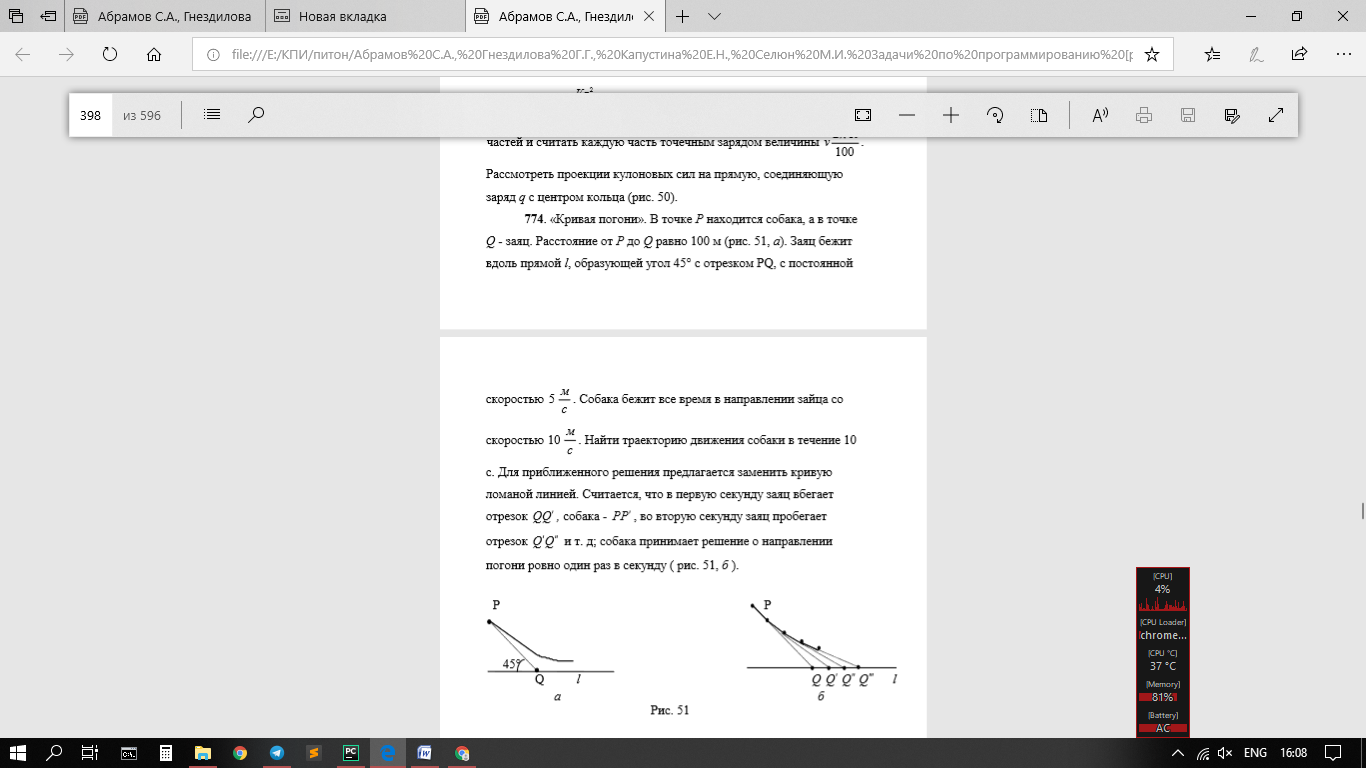
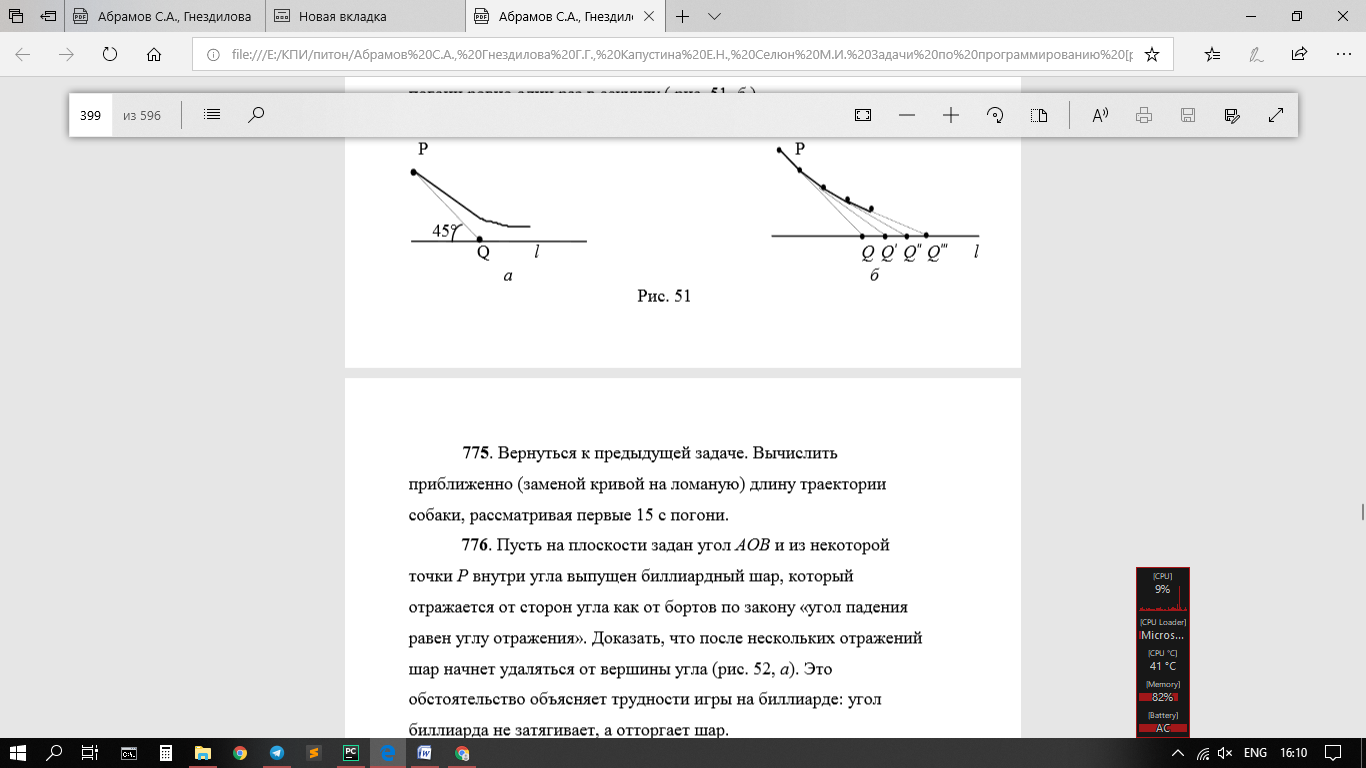
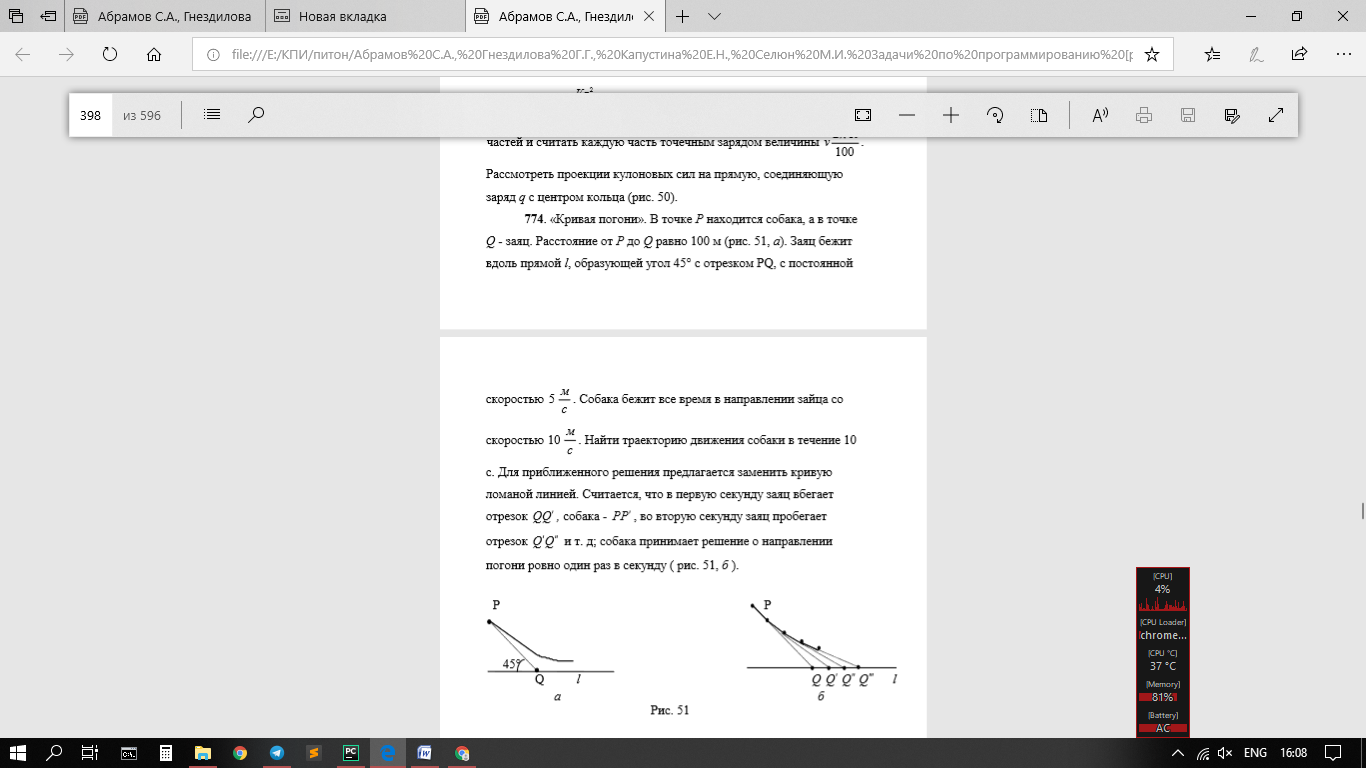




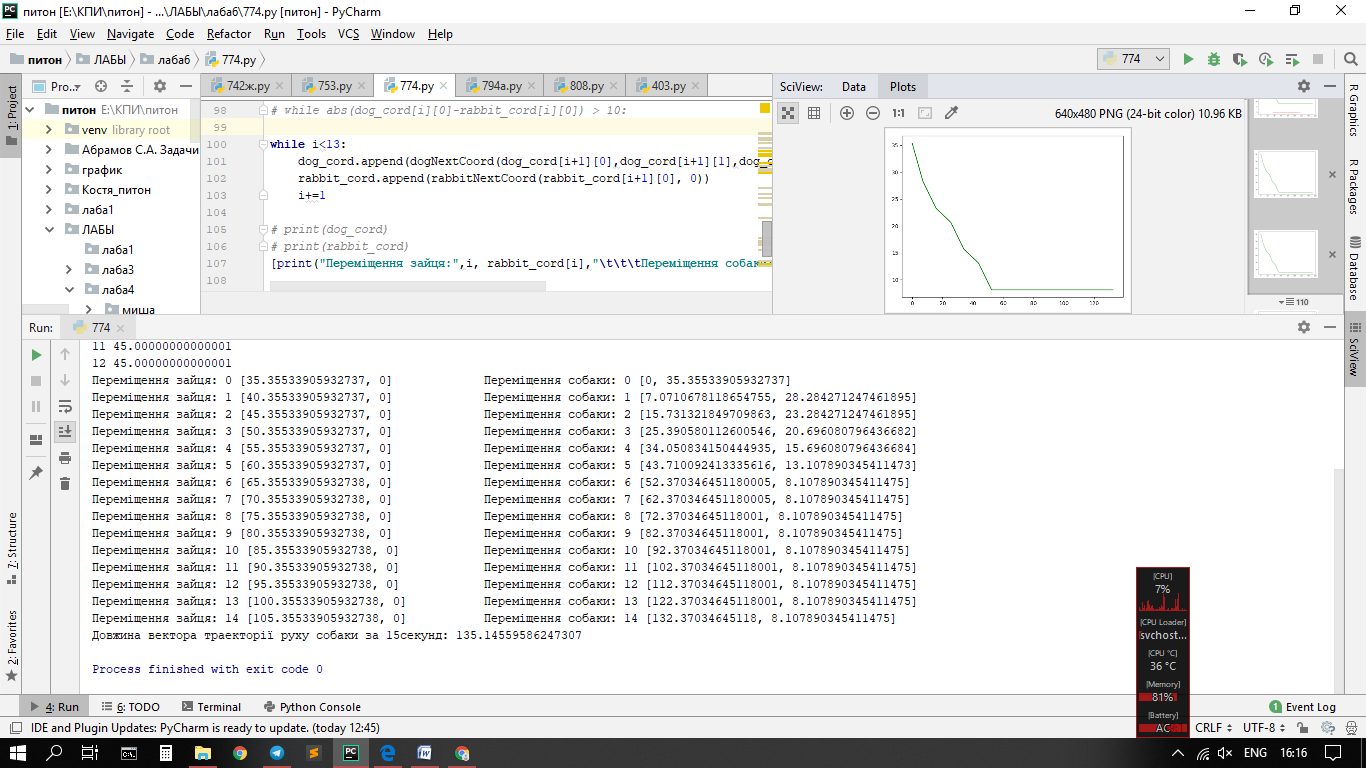


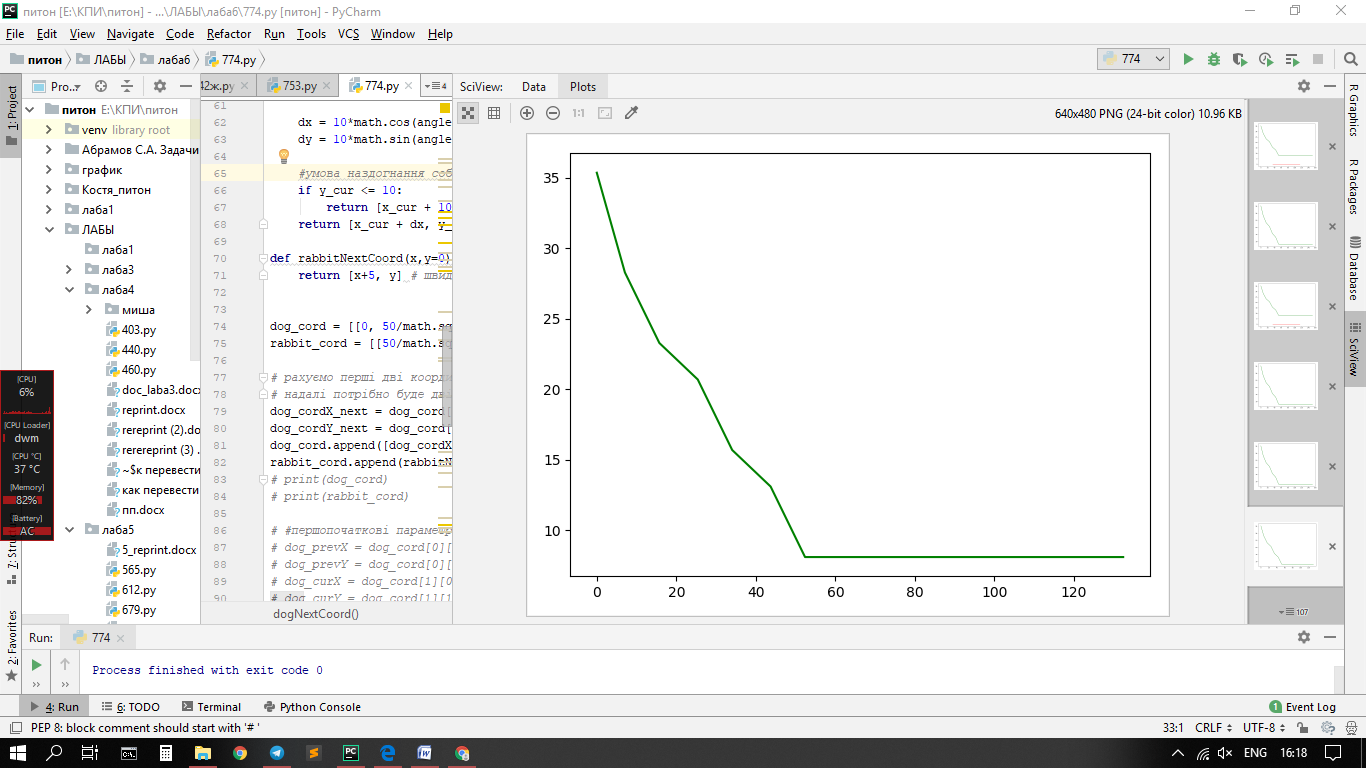
**import** random  
a = [random.uniform(-50, 50) **for** i **in** range(25)]  
print(**"a)"**,a)  
b = [random.randint(-20,20) **for** i **in** range(30)]  
print(**"б)"**,b)  
v = [random.uniform(0, 40) **for** i **in** range(20)]  
print(**"в)"**,v)  
g = [random.randint(0,1000) **for** i **in** range(35)]  
print(**"г)"**,g)  
d = [random.randint(1,20) **for** i **in** range(27)]  
print(**"д)"**,d)  
n = random.randint(1,30)  
e = [random.uniform(-100, 100) **for** i **in** range(n)]  
print(**"e)"**, e)  
n = random.randint(1,20)  
m = random.randint(1,20)  
zhe1 = [random.uniform(-150, 150) **for** i **in** range(n)]  
zhe2 = [[random.uniform(0, n) **for** i **in** range(m)]]  
print(**"ж)"**, zhe1, zhe2)  
two = [2 **for** i **in** range(7)]  
three = [3 **for** i **in** range(8)]  
z = two+three  
random.shuffle(z)  
print(**"з)"**,z)  
I = [i **for** i **in** range(1,13)]  
random.shuffle(I)  
print(**"и)"**,I)  
**import** string  
Alphabet = string.ascii\_lowercase  
k = [random.choice(Alphabet) **for** i **in** range(28)]  
print(**"к)"**,k)  
**def** notIn(lst, key):  
 **if** key **not in** lst:  
 **return True  
 return False**l = []  
**while**(len(l)<5):  
 r = random.choice(Alphabet)  
 **if** notIn(l,r):  
 l.append(r)  
print(**"л)"**,l)

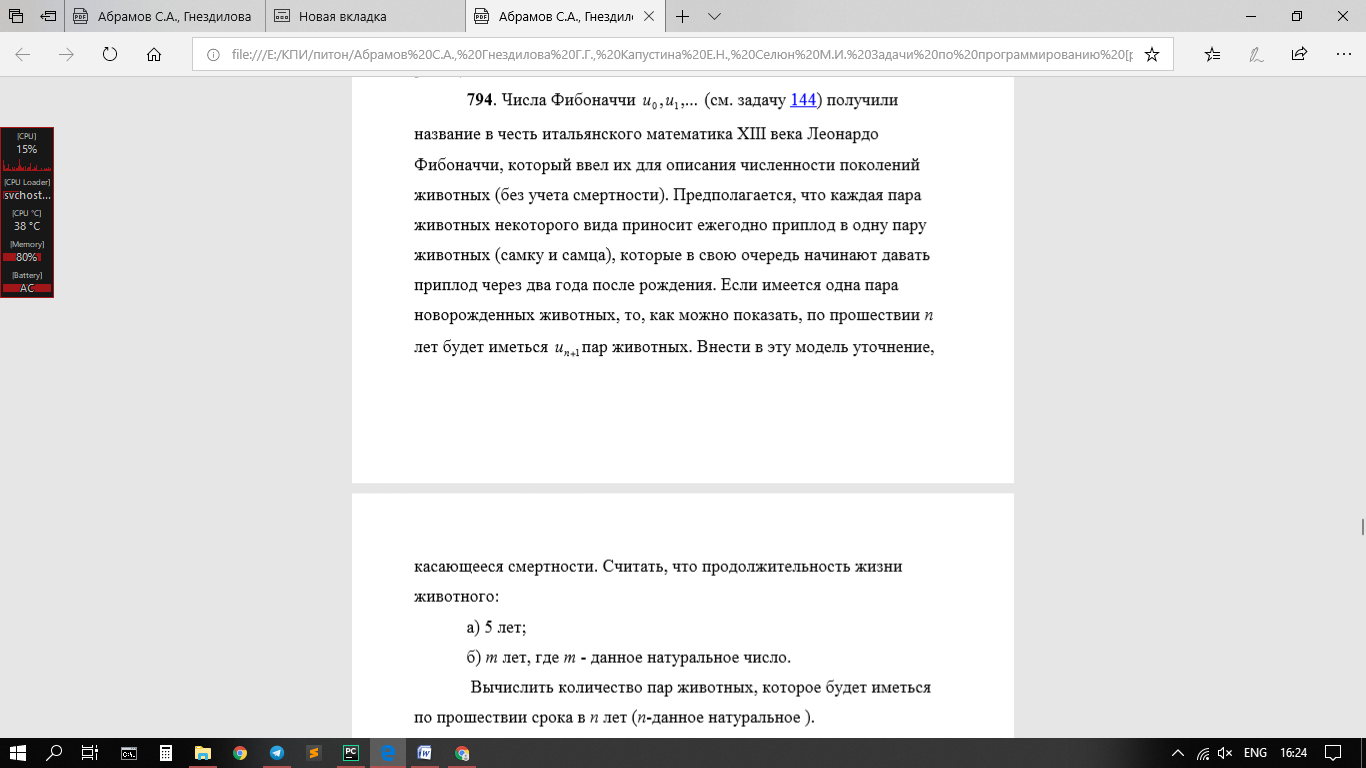


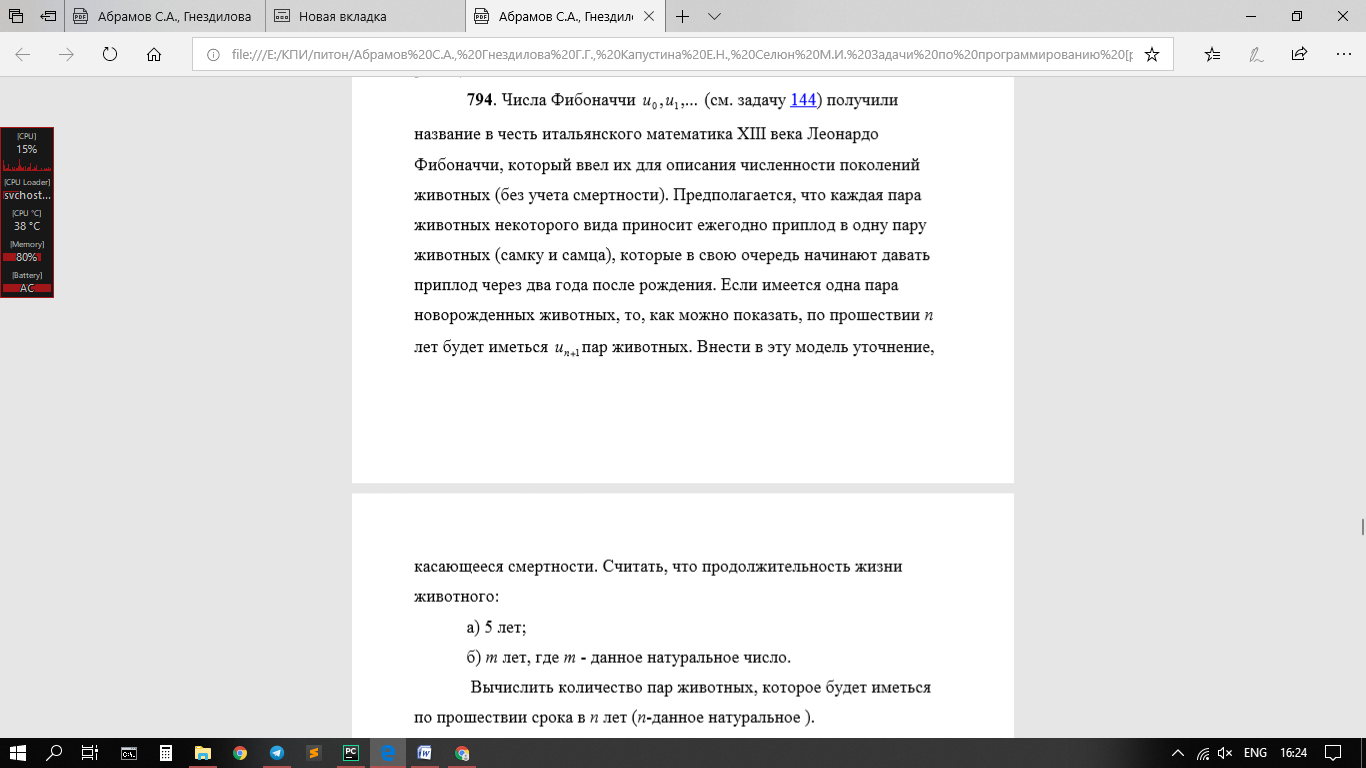


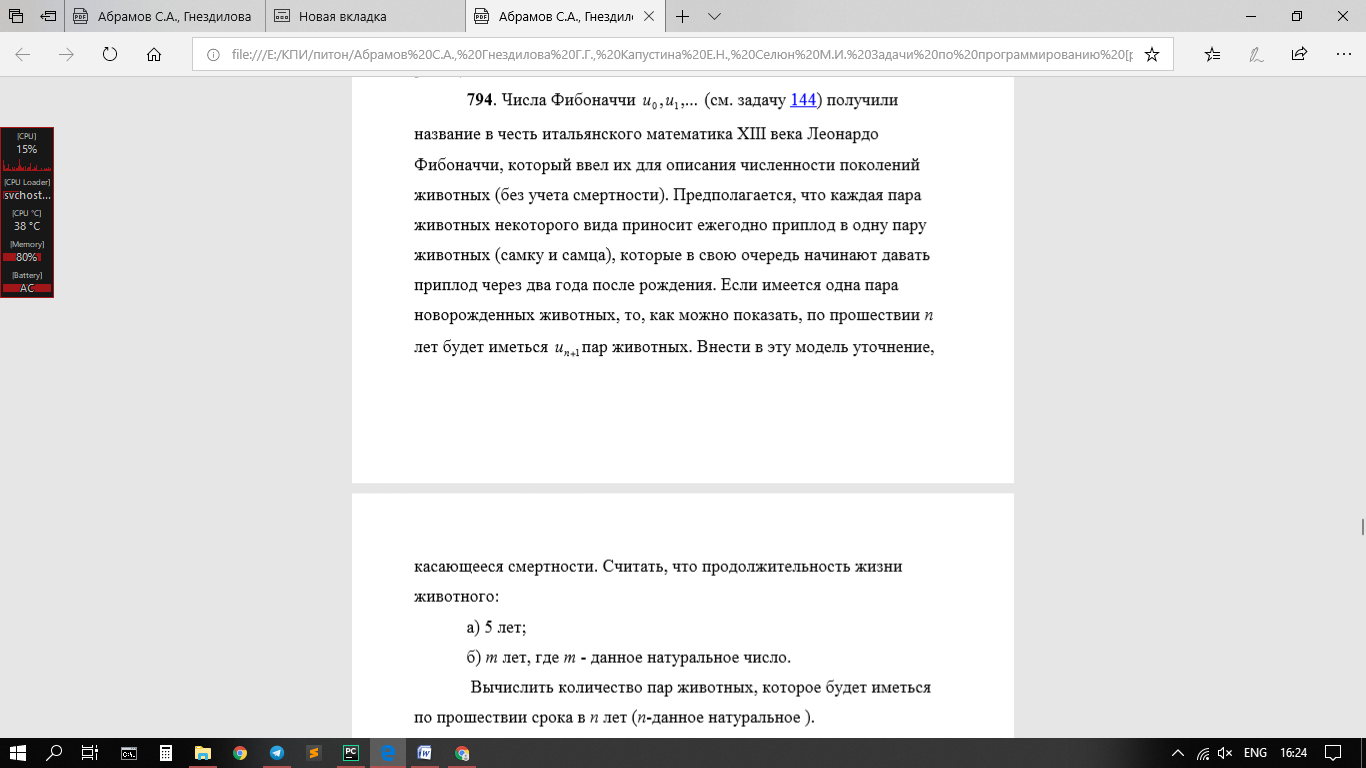
**import** math  
**import** numpy **as** np  
  
*#для розрахунку довжини вектора траекторії***def** get\_vector\_len(cord1, cord2):  
 **return** math.sqrt(cord1\*\*2+cord2\*\*2)  
  
**def** get\_vector(x1,y1, x2,y2):  
 *""" Щоб знайти вектор треба від координат кінця відняти коорд. початку """* **return** (x2-x1, y2-y1)  
  
**def** unit\_vector(vector):  
 *""" Returns the unit vector of the vector. """* **return** vector / np.linalg.norm(vector)  
  
**def** angle\_between(v1, v2):  
 *"""  
 Returns the angle in radians between vectors 'v1' and 'v2':  
 >>> angle\_between((1, 0), (0, 1))  
 1.5707963267948966  
 >>> angle\_between((1, 0), (1, 0))  
 0.0  
 >>> angle\_between((1, 0), (-1, 0))  
 3.141592653589793  
 """* v1\_u = unit\_vector(v1)  
 v2\_u = unit\_vector(v2)  
 **return** np.arccos(np.clip(np.dot(v1\_u, v2\_u), -1.0, 1.0))  
  
  
**def** dogNextCoord(x\_cur, y\_cur, x\_prev, y\_prev):  
 *# цей вектор використовується для обчислення кута між собакою та паралеллю ОХ* vector\_parallel\_OX = (x\_cur - x\_prev, x\_prev - x\_cur)vector\_DOG\_direction = (x\_cur - x\_prev, y\_cur - y\_prev)  
 angle = angle\_between(vector\_parallel\_OX, vector\_DOG\_direction)**if** math.degrees(angle) <= 0:  
 *#angle = int(angle)* angle += math.radians(30)  
 dx = 10\*math.cos(angle) *#швидкість = 10* dy = 10\*math.sin(angle)  
 *#умова наздогнання собакою зайця -> починають рухатись паралельно* **if** y\_cur <= 10:  
 **return** [x\_cur + 10, y\_cur]  
 **return** [x\_cur + dx, y\_cur - dy]  
  
**def** rabbitNextCoord(x,y=0):  
 **return** [x+5, y] *# швидкість=5*dog\_cord = [[0, 50/math.sqrt(2)]]  
rabbit\_cord = [[50/math.sqrt(2), 0]]  
  
*# рахуємо перші дві координати, оскільки за умовою задачі нам даний першопочатковий кут і відстань  
# надалі потрібно буде дві останні координати собаки у функцію для визначення кута її повороту*dog\_cordX\_next = dog\_cord[0][0] + 10\*math.cos(math.radians(45))  
dog\_cordY\_next = dog\_cord[0][1] - 10\*math.sin(math.radians(45))  
dog\_cord.append([dog\_cordX\_next,dog\_cordY\_next])  
rabbit\_cord.append(rabbitNextCoord(rabbit\_cord[0][0]))  
i = 0  
*# поставимо, що одна ітерація = одній секунді погоні***while** i<15:  
 dog\_cord.append(dogNextCoord(dog\_cord[i+1][0],dog\_cord[i+1][1],dog\_cord[i][0],dog\_cord[i][1]))  
 rabbit\_cord.append(rabbitNextCoord(rabbit\_cord[i+1][0], 0))  
 i+=1  
[print(**"Переміщення зайця:"**, rabbit\_cord[i],**"\t\t\tПереміщення собаки:"**, dog\_cord[i]) **for** i **in** range(len(dog\_cord))]  
  
*# вектор ( початок - точка початку рух, кінець - точка руху на 15 секунді)*traject = [dog\_cord[-1][0] - dog\_cord[0][0], dog\_cord[-1][1] - dog\_cord[0][1]]  
print(**'Довжина вектора траекторії руху собаки за 15секунд:'**, get\_vector\_len(traject[0], traject[1]))  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 **import** doctest  
 doctest.testmod(verbose=**False**)  
**import** matplotlib.pyplot **as** plt  
dog = np.array(dog\_cord).ravel()  
dogX = [dog[i] **for** i **in** range(len(dog)) **if** i%2==0]  
dogY = [dog[i] **for** i **in** range(len(dog)) **if** i%2!=0]  
plt.plot(dogX, dogY, **"g"**)  
plt.show()



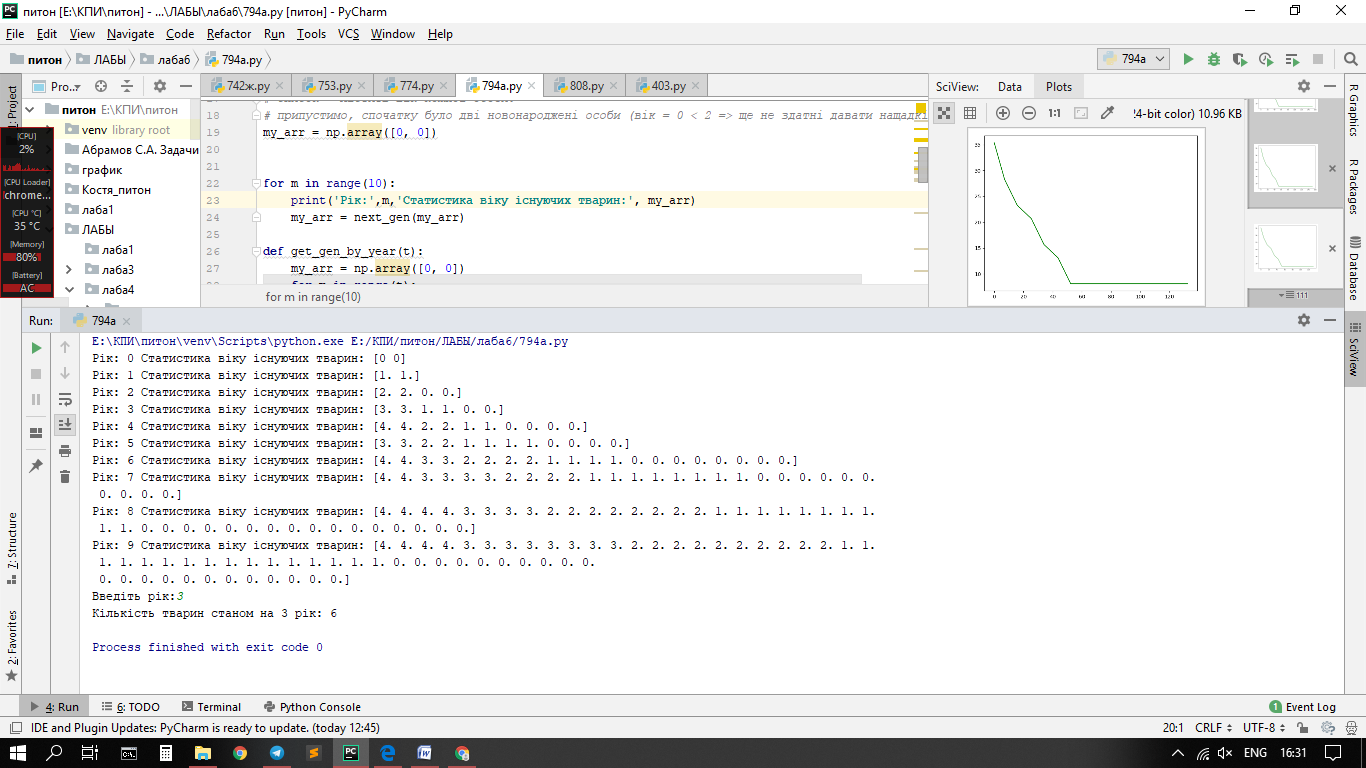


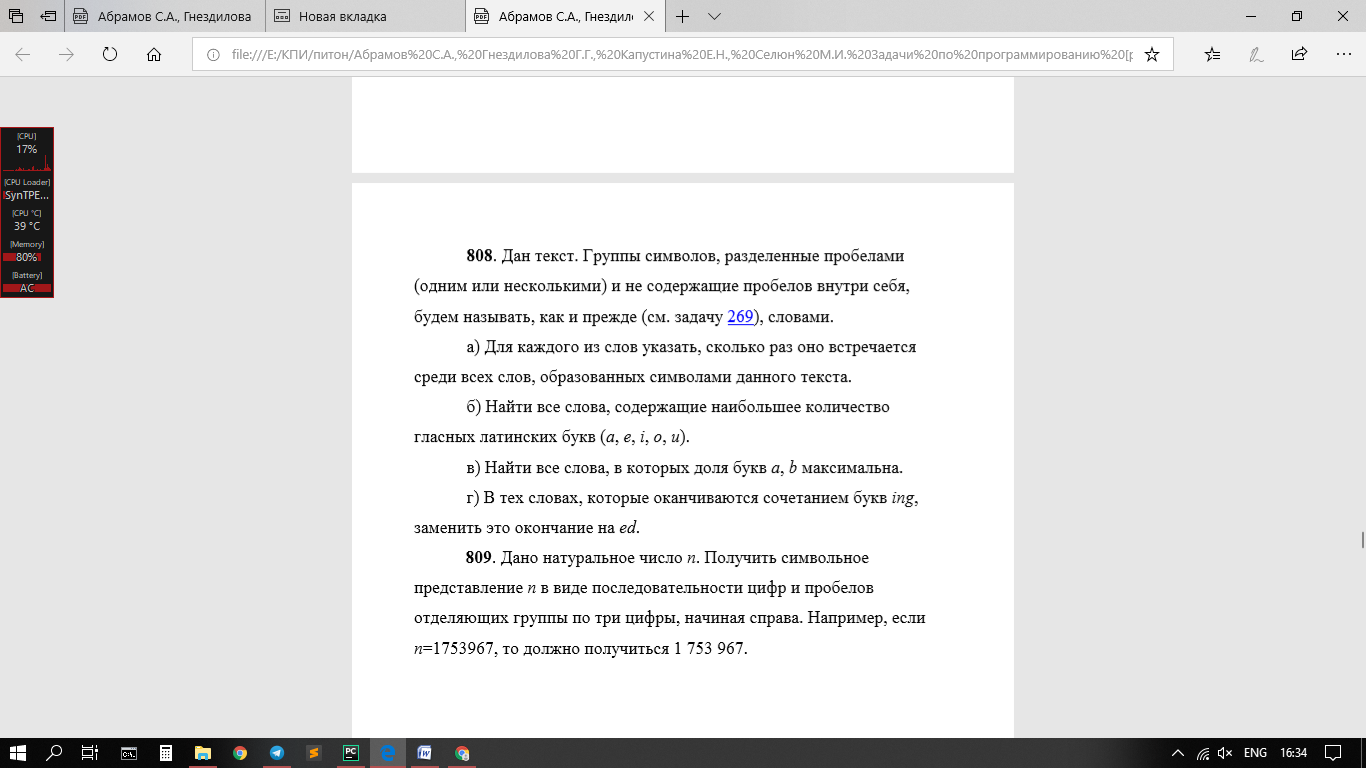






**import** numpy **as** np  
  
**def** next\_gen(arr):  
 *# кому 2 і більше - народжують інших* arr1 = arr + 1 *# всі виростають на 1 рік* to\_death = np.arange(len(arr1))[arr1 >= 5] *# живуть 5 років* arr2 = np.delete(arr1, to\_death)  
 parents = arr2 >= 2 *# статевозрілі(особи які вже здатні давати нащадків)* n\_p = len(arr2[parents])  
 new\_born = np.zeros(n\_p) *# поставимо, що народилося стільки ж скільки було статевозрілих* arr3 = np.append(arr2, new\_born)  
 **return** arr3  
  
*# список - містить вік кожної особи.  
# припустимо, спочатку було дві новонароджені особи (вік = 0 < 2 => ще не здатні давати нащадків)*my\_arr = np.array([0, 0])  
**for** m **in** range(10):  
 print(m, my\_arr)  
 my\_arr = next\_gen(my\_arr)  
  
**def** get\_gen\_by\_year(t):  
 my\_arr = np.array([0, 0])  
 **for** m **in** range(t):  
 my\_arr = next\_gen(my\_arr)  
 **return** my\_arr  
  
n = int(input(**'Введіть рік:'**))  
print(**'Кількість тварин станом на'**,n,**'рік:'**,len(get\_gen\_by\_year(n)))





Text = str(input(**"Введіть текст для обробки:"**))  
words = Text.split()  
L = len(words)

q = [ words.count(item) **for** item **in** words]  
[print(**'а) Слово '**,words[i],**' зустрічається '**,q[i],**'раз'**) **for** i **in** range(L)]  
print(**"\n"**)  
**def** calc\_quantity(words, searched\_letters):  
 *"""  
 Calculates quantity of defined letters in word.* **:param** *words:list of words, searched\_letters:list of defined letters* **:return***: countable\_list: list of entires searched letters in every*

*word  
 :Tests:  
 >>> calc\_quantity(['aeiouAEIOU'],vowels)[0]==10  
 True  
 """* countable\_list=[]  
 **for** word **in** words:  
 countable\_list.append(sum(x **in** searched\_letters **for** x **in** word))  
 **return** countable\_list  
  
vowels = list(**"aeiouAEIOU"**)  
number\_of\_vowels=calc\_quantity(words,vowels)  
[print(**'б) В слові '**, words[i],**' голосних букв налічується '**, number\_of\_vowels[i]) **for** i **in** range(L)]  
print(**"\n"**)  
a\_b = list(**"abAB"**)  
number\_of\_ab = calc\_quantity(words,a\_b)  
[print(**'в) В слові '**, words[i],**' букв a,b налічується '**, number\_of\_ab[i]) **for** i **in** range(L)]  
print(**"\n"**)  
**import** re  
print(**'г) Текст, в словах якого закінчення слів на -ing замінено на -ed: '**,re.sub(**r'ing\b'**, **'ed'**, Text))  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 **import** doctest  
 doctest.testmod(verbose=**False**)

