MINISTRY OF EDUCATION OF THE REPUBLIC OF BELARUS

EDUCATIONAL INSTITUTION

«BREST STATE TECHNICAL UNIVERSITY»

Department of IIT

**Laboratory work №2**

**For the third semester**

**Topic: «Python basic»**

Completed by the 2st year student of

Faculty of Electronic Information Systems

the group AC-57f Chernookiy I.V.

Checked by Khatskevich M.V.

Brest 2020

**Laboratory work №2**

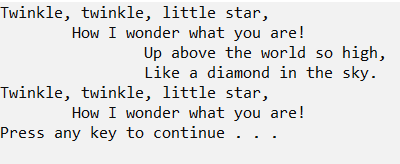
**Topic: «Python basic»**

**Variant 1 (1-10)**

**Goal:** to learn the main principles of Python and how to write programs on it.

**Task 1. (Python basic)**

print("Twinkle, twinkle, little star, \n\tHow I wonder what you are! \n\t\tUp above the world so high, \n\t\tLike a diamond in the sky. \nTwinkle, twinkle, little star, \n\tHow I wonder what you are!")

****

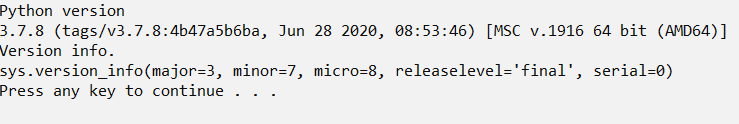
import sys

print("Python version")

print (sys.version)

print("Version info.")

print (sys.version\_info)

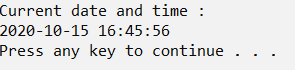


import datetime

now = datetime.datetime.now()

print ("Current date and time : ")

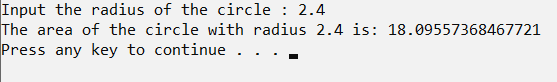
print (now.strftime("%Y-%m-%d %H:%M:%S"))



from math import pi

r = float(input ("Input the radius of the circle : "))

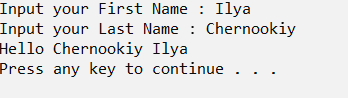
print ("The area of the circle with radius " + str(r) + " is: " + str(pi \* r\*\*2))



fname = input("Input your First Name : ")

lname = input("Input your Last Name : ")

print ("Hello " + lname + " " + fname)



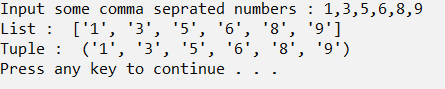
values = input("Input some comma seprated numbers : ")

list = values.split(",")

tuple = tuple(list)

print('List : ',list)

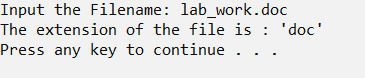
print('Tuple : ',tuple)



filename = input("Input the Filename: ")

f\_extns = filename.split(".")

print ("The extension of the file is : " + repr(f\_extns[-1]))



color\_list = ["Red","Green","White" ,"Black"]

print( "%s %s"%(color\_list[0],color\_list[-1]))



exam\_st\_date = (11,12,2014)

print( "The examination will start from : %i / %i / %i"%exam\_st\_date)



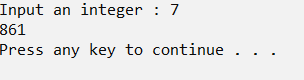
a = int(input("Input an integer : "))

n1 = int( "%s" % a )

n2 = int( "%s%s" % (a,a) )

n3 = int( "%s%s%s" % (a,a,a) )

print (n1+n2+n3)



**Task 2. (Data types)**

def string\_length(str1):

count = 0

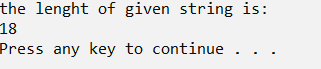
for char in str1:

count += 1

return count

print("the lenght of given string is: ")

print(string\_length('i hate programming'))



def sum\_list(items):

sum\_numbers = 0

for x in items:

sum\_numbers += x

return sum\_numbers

print(sum\_list([24,107,-34]))



def multiply\_list(items):

tot = 1

for x in items:

tot \*= x

return tot

print(multiply\_list([24,107,-34]))



def max\_num\_in\_list( list ):

max = list[ 0 ]

for a in list:

if a > max:

max = a

return max

print(max\_num\_in\_list([1, 2, -8, 0]))



def smallest\_num\_in\_list( list ):

min = list[ 0 ]

for a in list:

if a < min:

min = a

return min

print(smallest\_num\_in\_list([1, 2, -8, 0]))



def char\_frequency(str1):

dict = {}

for n in str1:

keys = dict.keys()

if n in keys:

dict[n] += 1

else:

dict[n] = 1

return dict

print(char\_frequency('google.com'))



def match\_words(words):

ctr = 0

for word in words:

if len(word) > 1 and word[0] == word[-1]:

ctr += 1

return ctr

print(match\_words(['abc', 'xyz', 'aba', '1221']))



def last(n): return n[-1]

def sort\_list\_last(tuples):

return sorted(tuples, key=last)

print(sort\_list\_last([(2, 5), (1, 2), (4, 4), (2, 3), (2, 1)]))



def string\_both\_ends(str):

if len(str) < 2:

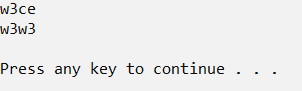
return ''

return str[0:2] + str[-2:]

print(string\_both\_ends('w3resource'))

print(string\_both\_ends('w3'))

print(string\_both\_ends('w'))



**Task 3. (Python strings)**

def string\_length(str1):

count = 0

for char in str1:

count += 1

return count

print(string\_length('do\_not\_give\_up'))



def change\_char(str1):

char = str1[0]

str1 = str1.replace(char, '$')

str1 = char + str1[1:]

return str1

print(change\_char('restart'))



def chars\_mix\_up(a, b):

new\_a = b[:2] + a[2:]

new\_b = a[:2] + b[2:]

return new\_a + ' ' + new\_b

print(chars\_mix\_up('abc', 'xyz'))



def add\_string(str1):

length = len(str1)

if length > 2:

if str1[-3:] == 'ing':

str1 += 'ly'

else:

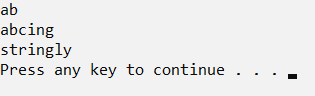
str1 += 'ing'

return str1

print(add\_string('ab'))

print(add\_string('abc'))

print(add\_string('string'))



def not\_poor(str1):

snot = str1.find('not')

spoor = str1.find('poor')

if spoor > snot and snot>0 and spoor>0:

str1 = str1.replace(str1[snot:(spoor+4)], 'good')

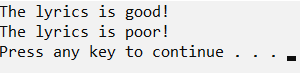
return str1

else:

return str1

print(not\_poor('The lyrics is not that poor!'))

print(not\_poor('The lyrics is poor!'))



def find\_longest\_word(words\_list):

word\_len = []

for n in words\_list:

word\_len.append((len(n), n))

word\_len.sort()

return word\_len[-1][1]

print(find\_longest\_word(["Big", "Huge", "Enormous"]))



def remove\_char(str, n):

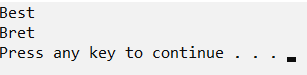
first\_part = str[:n]

last\_part = str[n+1:]

return first\_part + last\_part

print(remove\_char('Brest', 1))

print(remove\_char('Brest', 3))

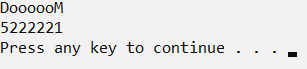


def change\_sring(str1):

return str1[-1:] + str1[1:-1] + str1[:1]

print(change\_sring('MoooooD'))

print(change\_sring('1222225'))



**Task 4. (Python dictionary)**

import operator

d = {1: 2, 3: 4, 4: 3, 2: 1, 0: 0}

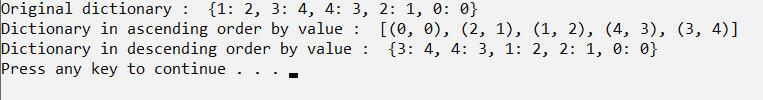
print('Original dictionary : ',d)

sorted\_d = sorted(d.items(), key=operator.itemgetter(1))

print('Dictionary in ascending order by value : ',sorted\_d)

sorted\_d = dict( sorted(d.items(), key=operator.itemgetter(1),reverse=True))

print('Dictionary in descending order by value : ',sorted\_d)

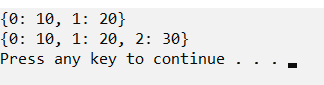


d = {0:10, 1:20}

print(d)

d.update({2:30})

print(d)



dic1={1:10, 2:20}

dic2={3:30, 4:40}

dic3={5:50,6:60}

dic4 = {}

for d in (dic1, dic2, dic3): dic4.update(d)

print(dic4)



d = {"key1": 10, "key2": 23}

if "key1" in d:

print("this will execute")

if "nonexistent key" in d:

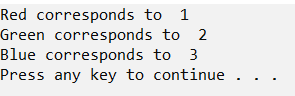
print("this will not")



d = {'Red': 1, 'Green': 2, 'Blue': 3}

for color\_key, value in d.items():

print(color\_key, 'corresponds to ', d[color\_key])



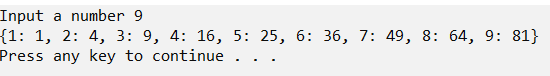
n=int(input("Input a number "))

d = dict()

for x in range(1,n+1):

d[x]=x\*x

print(d)



d=dict()

for x in range(1,11):

d[x]=x\*\*2

print(d)



d1 = {'a': 100, 'b': 200}

d2 = {'x': 300, 'y': 200}

d = d1.copy()

d.update(d2)

print(d)



my\_dict = {'data1':100,'data2':-54,'data3':247}

print(sum(my\_dict.values()))



**Task 5. (Python tuple)**

#Create an empty tuple

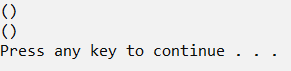
x = ()

print(x)

#Create an empty tuple with tuple() function built-in Python

tuplex = tuple()

print(tuplex)



#Create a tuple with different data types

tuplex = ("tuple", False, 3.2, 1)

print(tuplex)



#Create a tuple with numbers

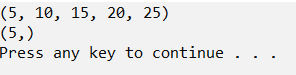
tuplex = 5, 10, 15, 20, 25

print(tuplex)

#Create a tuple of one item

tuplex = 5,

print(tuplex)



#create a tuple

tuplex = 4, 8, 3

print(tuplex)

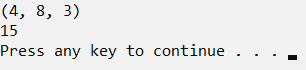
n1, n2, n3 = tuplex

#unpack a tuple in variables

print(n1 + n2 + n3)

#the number of variables must be equal to the number of items of the tuple

n1, n2, n3 = tuplex



#create a tuple

tuplex = (4, 6, 2, 8, 3, 1)

print(tuplex)

#tuples are immutable, so you can not add new elements

#using merge of tuples with the + operator you can add an element and it will create a new tuple

tuplex = tuplex + (9,)

print(tuplex)

#adding items in a specific index

tuplex = tuplex[:5] + (15, 20, 25) + tuplex[:5]

print(tuplex)

#converting the tuple to list

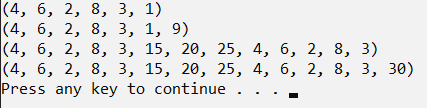
listx = list(tuplex)

#use different ways to add items in list

listx.append(30)

tuplex = tuple(listx)

print(tuplex)



tup = ('e', 'x', 'e', 'r', 'c', 'i', 's', 'e', 's')

str = ''.join(tup)

print(str)



#Get an item of the tuple

tuplex = ("a", "b", "e", "h", "o", "m", "p", "f", "u")

print(tuplex)

#Get item (4th element)of the tuple by index

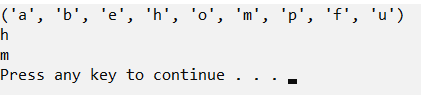
item = tuplex[3]

print(item)

#Get item (4th element from last)by index negative

item1 = tuplex[-4]

print(item1)

****

from copy import deepcopy

#create a tuple

tuplex = ("HELLO", 5, [], True)

print(tuplex)

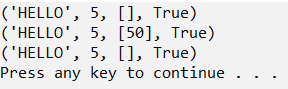
#make a copy of a tuple using deepcopy() function

tuplex\_colon = deepcopy(tuplex)

tuplex\_colon[2].append(50)

print(tuplex\_colon)

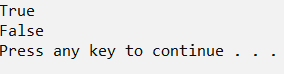
print(tuplex)



tuplex = ("a", "b", "e", "h", "o", "m", "p", "f", "u")

print("p" in tuplex)

print(5 in tuplex)



**Task 6. (Python sets)**

#Create a new empty set

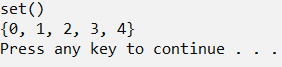
x = set()

print(x)

#Create a non empty set

n = set([0, 1, 2, 3, 4])

print(n)

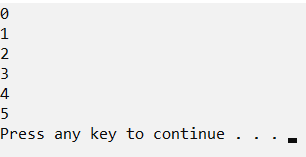


#Create a set

num\_set = set([0, 1, 2, 3, 4, 5])

for n in num\_set:

print(n)



#A new empty set

color\_set = set()

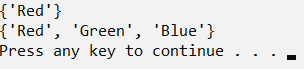
color\_set.add("Red")

print(color\_set)

#Add multiple items

color\_set.update(["Blue", "Green"])

print(color\_set)



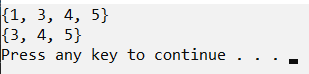
num\_set = set([0, 1, 3, 4, 5])

num\_set.pop()

print(num\_set)

num\_set.pop()

print(num\_set)



#Create a new set

num\_set = set([0, 1, 2, 3, 4, 5])

#Discard number 4

num\_set.discard(4)

print(num\_set)



#Intersection

setx = set(["green", "blue"])

sety = set(["blue", "yellow"])

setz = setx & sety

print(setz)



#Union

setx = set(["green", "blue"])

sety = set(["blue", "yellow"])

seta = setx | sety

print(seta)



setx = set(["apple", "mango"])

sety = set(["mango", "orange"])

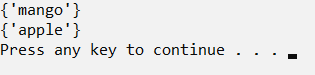
setz = setx & sety

print(setz)

#Set difference

setb = setx – setz

print(setb)



setx = set(["apple", "mango"])

sety = set(["mango", "orange"])

#Symmetric difference

setc = setx ^ sety

print(setc)



print("Check if a set is a subset of another set, using comparison operators and issubset():\n")

setx = set(["apple", "mango"])

sety = set(["mango", "orange"])

setz = set(["mango"])

print("x: ",setx)

print("y: ",sety)

print("z: ",setz,"\n")

print("If x is subset of y")

print(setx <= sety)

print(setx.issubset(sety))

print("If y is subset of x")

print(sety <= setx)

print(sety.issubset(setx))

print("\nIf y is subset of z")

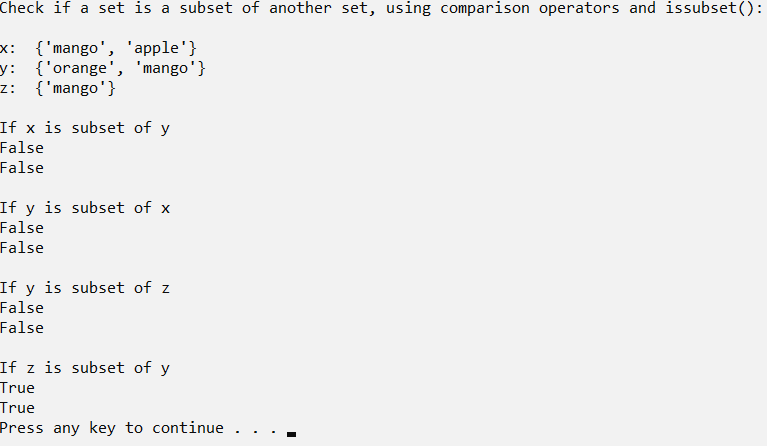
print(sety <= setz)

print(sety.issubset(setz))

print("If z is subset of y")

print(setz <= sety)

print(setz.issubset(sety))



**Conclusion:** learn how to work with different types of python syntax and how to make various mathematical transformations with tuple, sets and dictionary.