Laboratorium **Programowanie w języku Python 2** Wydział Elektrotechniki Automatyki I Informatyki Politechnika Świętokrzyska	
Studia: **Stacionarne I stopnia**	Kierunek: **Informatyka**
Data wykonania: **18.03.2021**	Grupa: **3ID16B**
Imię I nazwisko: **Arkadiusz Więcław**	Temat ćwiczenia: **Iteratory i generatory**

Zad 1 - 4:

```
from matplotlib import pylab
from pylab import *

def p1():
    """
    Przyklad na poczatku tworzy macierz a 10x10
    potem tworzy na jej podstawie macierz b wypelniona zerami
    nastepnie podczas dzialanie petli sa  przypisywane
    wartosc macierzy a do macierzy b
    """
    a = eye(10)
    b = zeros(a.shape)
    for w in list(range(a.shape[0])):
        for k in list(range(a.shape[1])):
            print(w,k)
            b[w,k] = a[w,k]

def p2():
    """
    Przyklad na poczatku tworzy macierz 10x8 ktora zapisuje do zminnej
a
    nastepnie elementa  od indeksie 3 i 5 w calej macierzy sa
przypisywane wartosc 1
    na podstawie macierzy a tworzona jest tworzona macierz b ktore
jest zerowana
    na koncu sa petla ktora ma za zadanie wypelnij macierz b  tak samo
jak macierz a
    """
    a = zeros((10,8))
    a[:,(3,5)] = 1
    b = zeros(a.shape)
    for w in range(a.shape[1]):
        for k in range(a.shape[1]):
            print(w,k)
            b[w,k] = a[w,k]
    imshow(b, interpolation='none',cmap='Blues')

def p3():
    """
```

```
    Przyklad na poczatku tworzy macierz 11x9 i zapisuje ja do zmiennej
a
    """
    a = zeros((11,9))
    a[5::3,0:8:3] = 1
    b = zeros(a.shape)
    for (y,x) in [(y,x) for y in range(a.shape[0]) for x in
    range(a.shape[1]) if a[y,x]>0.5]:
        print(y,x)
        b[y,x]=a[y,x]
    imshow(b,interpolation='none',cmap=cm.gray)

def p4():
    """
    Przyklad na poczatku tworzy macierz 11x9 i zapisuje ja do zmiennej
    w rzedach 5 i  8 , w kolumnach 0 , 3, 6 macierzy   są wstawiane
liczby 1
    nastepnie na bazie utworzonej macierzy jest tworzona nowa macierz
    """
    a=zeros((11,9))
    a[5::3,0:8:3]=1
    b=zeros(a.shape)
    for y,x in ndindex(a.shape):
        print (y,x)
        b[y,x]=a[y,x]
    imshow(b,interpolation='bilinear',cmap=cm.hot)

def p5():
    """
    Przyklad tworzy macierz 11x9
    nastepnie przypisuje wartosc 1 co drugiemu rzedowi i kolumnie
    potem na podstawie macierzy a tworzy macierz b  ktore jest
wyzerowana.
     Iterator ndumerate w pentli przypisuje wartosc z macierz a do
macierzy b.
    """
    a=zeros((11,9))
    a[::2,::2]=1
    b=zeros(a.shape)
    for (y,x),i in ndenumerate(a):
        print(y,x,i)
        b[y,x]=a[y,x]
```

```
    imshow(b,interpolation='none',cmap=cm.autumn)

def p6():
    """
    Przyklad tworzy tuple i iterator ktory bedzie iterowal po tupli
    na koncu wyswietla nastepne elementy tupli
    """
    mytuple = ("apple", "banana", "cherry")
    myit = iter(mytuple)
    print(next(myit))
    print(next(myit))
    print(next(myit))

def p7():
    """
    przyklad na poczatku tworzy klase MyNumbers ktore bedzie  naszym
nowym iteratorem
    klasa zawiera dwie metody __iter__() i __next__()
    iter zwraca iterator i tworzy wlasciwosc  o nazwie a
    next sluzy do iterowanie w naszym przypadku podnosi  atrybut a o 2
przy kazdym wywolanie
    tworzymy obiekt myclass nastepnie  metoda iter tworzymy iterator
obiektu myclass
    """
    class MyNumbers:
        def __iter__(self):
            self.a = 1
            return self
        def __next__(self):
            x = self.a
            self.a += 2
            return x
    myclass = MyNumbers()
    myiter = iter(myclass)
    print(next(myiter))

def p8():
    """
    Przyklad robi to samo co przyklad  wyzej ale wewnatrz metody
__next__ jest instrukcja warunkowa
    wartosc atrybut a nie moze byc wieksza niz 20 w przeciwynym
wypadku zglosi wyjatek StopIteration.
```

```
    """
    class MyNumbers:
        def __iter__(self):
            self.a = 1
            return self
        def __next__(self):
            if self.a <= 20:
                x = self.a
                self.a += 2
                return x
            else:
                raise StopIteration
    myclass = MyNumbers()
    myiter = iter(myclass)
    for x in myiter:
        print(x)

def p9():
    """
    Przyklad liczy  koljene wyrazy ciagu fibonaiego
    zapomoca iteratora. Maxymalna  liczba do ktororej program  ma
liczyc  ciag jest podawany przez uzytkownika.
    """
    class Fib:
        def __init__(self, max):
            self.max = max
        def __iter__(self):
            self.a = 0
            self.b = 1
            return self

        def __next__(self):
            fib = self.a
            if fib > self.max:
                raise StopIteration
            self.a, self.b = self.b, self.a + self.b
            return fib
    liczba = int(input("Podaj liczbe "))
    fib = Fib(liczba)
    iterator  = iter(fib)
    for x in iterator:
        print(x)
```

```
def p10():
    """
    Przyklad ma za zadanie odworcic slowo w tym celu jest utworzony
generator .
    Generator zwraca kolejne znaki w odwroconej kolejnosc
wykorzystuje  yield
    """
    def reverse(data):
        for index in range(len(data)-1, -1, -1):
            yield data[index]
    for char in reverse('golf'):
        print(char)
#zad2
def nindex():
    """
    przyklad tworzy macierz  10x10 a nastepnie dzieki iteratorowi
ndindex iteruje i przypisuje mu
    jedenki jesli x jest nieparzyste a jesli jest parzyste to 2
    """
    matrx = zeros((10,10))
    for x , y in ndindex(matrx.shape):
        if x & 2 :
            matrx[x,y] = 1
        else:
            matrx[x,y] = 2
    print(matrx)
def range_p():
    """
    przyklad liczyc sume elementow dla kazdej wewnetrznej  listy i
zapisuje je w nowej tablicy
    """
    list2 = 0
    list1 = []
    w = [ [1,2,3,4,5] , [7,6,3,2,1] ]
    for s in  range(len(w)):
        for z in w[s]:
            list2 += z
        list1.append(list2)
        list2 = 0
    print("Suma obu list = ", list1)

def comprehension():
```

```
    """
    Przyklad tworzy macierz 5x10 ktory ,nastepnie utworzonej macierzy
przypisuje wartosc rzeda od 2 do 6
    .Na podstawie macierzy a tworzona jest macierz b . Macierz a jest
literowane za pomoca list comprahetnych
    i petli for  . Wartosc elementow macierz a  ktore sa mnozeno przez
wartosc y
    sa przypisywane  do macierzy b
    """
    a = zeros((5,10))
    a[::1,2:6] = 2
    b = zeros(a.shape)
    for (y,x) in [(y,x) for y in range(a.shape[0]) for x in
range(a.shape[1])]:
        b[y,x] = a[y,x] + y
    print(b)

def ndenumerate():
    """
    Przyklad tworzy macierz 6x10 a cnastpenie przypisuje jej 3 co
cztery kolumny i rzedy
    .Potem za pomoca iteratora ndenumerate iterujemy po kolejnych
elementach macierz przypisujac je
    do innej macierzy .
    """
    a = zeros((6,10))
    a[::2,::2]= 3
    b = zeros(a.shape)
    for (y,x),i in ndenumerate(a):
        b[y,x] = a[y,x]
    print(b)

#zad3
def iter_p1():
    """
    Program przyjmuje od uzytkownika do jakiej liczby ma potegowac
utworzony iterator
    """

    class Potenga:
        def __init__(self, max_w):
            self.max_w = max_w
```

```
        def __iter__(self):
            self.x = 0
            self.w = 1
            return self

        def __next__(self):
            x = self.x
            w = self.w
            if w <= self.max_w + 1:
                self.x = self.w * self.w
                self.w += 1
                return x
            else:
                raise StopIteration

    pote = Potenga(int(input("Maxymalna liczba  potegowania ")))
    iter_pot = iter(pote)
    for w in iter_pot:
        print(w)

#zad4
def generator_p():
    """
    Przyklad z pomoca utworzonego generatora podnosi liczbe  do potegi
wprowadzonej przez uzytkownika
    """
    def potenga(n):
        pow = 1
        for i in range(n):
            yield pow
            pow *= 5
    for u in potenga(int(input(" Do jakiej potegi podnies liczbe 5
"))):
        print(u)
if __name__ == '__main__':
    print("Zadanie 1 = ")

    print("Przyklad 1 = ")
    p1()
    print("Przyklad 2 = ")
    p2()
    print("Przyklad 3 = ")
```

```
    p3()
    print("Przyklad 4 = ")
    p4()
    print("Przyklad 5 = ")
    p5()
    print("Przyklad 6 = ")
    p6()
    print("Przyklad 7 = ")
    p7()
    print("Przyklad 8 = ")
    p8()
    print("Przyklad 9 = ")
    p9()
    print("Przyklad 10 = ")
    p10()
    print("Zadanie 2 = ")

    print("Przyklad 1 iterowanie po tablicach za pomoca nindex = ")
    nindex()
    print("Przyklad 2 iterowanie po tablicach za pomoca range = ")
    range_p()
    print("Przyklad 3 iterowanie po tablicach za pomoca comprahetnych
list ")
    comprehension()
    print("przyklad 5 iterowanie po tablicach za pomoca ndenumerate
")
    ndenumerate()

    print("Zadanie 3 = ")

    print("przyklad 1 program wypisuje znaki ciagu")
    iter_p1()

    print("Zadanie 4 = ")

    print("Pzyklad 1")
    generator_p()
```