МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Кафедра вычислительной математики и программирования

Дисциплина: «Разработка ПО для высокопроизводительных систем»

Отчет по лабораторной работе №3 по NumPy

Выполнил: Павлов М.М.
Группа: М8О-103М-20
Преподаватель: Поповкин А.В.

```
import numpy as np

def run():
  #1
  print("-------------------------1")
  v = np.array([1, 2, 3, 4, 5])
  z = np.array(v)
  for i in range(4):
      z = np.insert(z, 1+i*4, [0,0,0], axis=0)
  print(z)

  #2
  print("-------------------------2")
  m2 = np.random.rand(5,5)
  print(m2)
  m2[[1,2]] = m2[[2,1]]
  print(m2)

  #3
  print("-------------------------3")
  triangles = np.random.randint(0,10,(10,3))
  print(triangles)
  temp = np.roll(triangles.repeat(2,axis=1),-1,axis=1)
  print(temp)
  pairs = temp.reshape(len(temp)*3,2)
  pairs = np.sort(pairs,axis=1)
  pairs = pairs.view( dtype=[('vertex1',pairs.dtype),('vertex2',pairs.dtype)])
  print(pairs)
  print(np.unique(pairs))

  #4
  print("-------------------------4")
  v = np.random.randint(10, size=10)
  print(v)
  o = np.repeat(np.arange(v.size), v)
  print(o)
  print(np.bincount(o))

  #5
  print("-------------------------5")
  n = 4
  a = np.random.randint(0, 20, 15)
  print(a)
  ma = np.cumsum(a)
  print(ma)
  ma[n:] -= ma[:-n]
  print(ma[n - 1:]/n)

  #6
  print("-------------------------6")
  Z = np.arange(13)
```

```
  width = 3
  arr6 = np.lib.stride_tricks.as_strided(Z, shape=(Z.size - width + 1, width),
strides=(Z.itemsize, Z.itemsize))
  print(arr6)

  #7
  print("--------------------------7")
  a = np.random.randint(0,2,10).astype(bool)
  print(a)
  print(np.logical_not(a).astype(bool))

  #8
  print("--------------------------8")
  a = np.random.randint(10, size=(5,5))
  print(a)
  print(np.linalg.matrix_rank(a))

  #9
  print("--------------------------9")
  a = np.random.randint(0,10,10)
  print(a)
  print(np.bincount(a).argmax())

  #10
  print("--------------------------10")
  a = np.random.randint(0, 10, (10, 10))
  print(a)
  n = 3
  print(a.shape[0] - n + 1, a.shape[1] - n + 1, n, n)
  print(a.strides+a.strides)

  subBlocks = np.lib.stride_tricks.as_strided(
      a,
      shape=(a.shape[0] - n + 1, a.shape[1] - n + 1, n, n),
      strides=a.strides + a.strides)
  print(subBlocks)

  #11
  print("--------------------------11")
  p, n = 5, 10
  matricesArray = np.ones((p,n,n))
  vectorsArray = np.ones((p,n))

  print(matricesArray)
  print(vectorsArray)
  prodArray = np.einsum('ijk,ik->k', matricesArray, vectorsArray)
  print(prodArray)

  #12
  print("--------------------------12")
  a = np.random.randint(0, 3, (16, 16))
```

```
print(a)
n = 4
subBlocks = np.lib.stride_tricks.as_strided(
    a,
    shape=(a.shape[0] - n + 1, a.shape[1] - n + 1, n, n),
    strides=a.strides + a.strides
)
print(np.sum(subBlocks, axis = (-2, -1)))

#13
print("--------------------------13")
n = 5
a = np.random.randint(0,100,10)
print(a)
print (a[np.argpartition(a, n)][len(a) - n:])

#14
print("--------------------------14")
a = [[np.random.randint(0, 5, 3)] for i in range(2)]
print(a)
print(np.array(np.meshgrid(a[0], a[1])).T.reshape(-1, 2))


#15
print("--------------------------15")
a = np.random.randint(0,5,(8,3))
b = np.random.randint(0,5,(2,2))
print(a)
print(b)
c = (a[..., np.newaxis, np.newaxis] == b)
print(c)
input()
rows = np.where(c.any((1,3)).all(1))[0]
print(a[rows])

#16
print("--------------------------16")
a = np.random.randint(0, 3, (10, 3))
print(a)
print(a[~np.logical_and.reduce(a[:,1:] == a[:,:-1], axis=1)])

#17
print("--------------------------17")
a = np.random.randint(0, 64, 5, dtype=np.uint8)
print(a)
print(np.unpackbits(a).reshape(-1, 8))

#18
print("--------------------------18")
a = np.random.randint(0, 2, (10, 3))
print(a)
```

```
print(np.unique(a, axis = 0))

#19
print("--------------------------19")
a, b = np.arange(5), np.arange(5) + 5
print(a)
print(b)
print("inner:")
print(np.einsum("i,i->", a, b))
print("outer:")
print(np.einsum("i,j->ij", a, b))
print("sum:")
print(np.einsum("i->", a), np.einsum("i->", b))
print("mul:")
print(np.einsum("i,i->i", a, b))
```