

a-piege.py

```
import numpy as np

np.random.seed(2329392) # figer le hasard

t = np.random.randint(100, 200, (4, 10))

derniere = t[ -1 , -1 ]
print(t)
print(derniere)

print("---")
piege = t[1 : 4, 6 ] # piege
aide = t[1:4 , 6 ]
paspiege = t[1:4, 6]
print("Les valeurs des lignes de 1 à 3, de la colonne 6")
print(piege)
```

corr-td3.py

```
#!/usr/bin/env python
# coding: utf-8

# In[34]:

import numpy as np
np.set_printoptions(precision=3, threshold=0, edgeitems=5)
import numpy_html

# # Q1

# In[35]:

np.array([4, 5])

# In[36]:

np.array([4, 10, 5])

# In[37]:

np.array((4, 5))

# # Q2/3/4/5

# In[38]:

np.zeros((4, 5))

# In[39]:

np.zeros( (4, 5) )

# In[40]:

sh = (4, 5)
np.zeros(sh)

# In[41]:

np.ones((5, 4))

# In[42]:
```

```
np.random.uniform(2, 3, (4, 5))
```

```
# In[43]:
```

```
np.random.normal(2, 3, (4, 5))
```

```
# In[99]:
```

```
tab1000 = np.random.normal(1000, 0.01, (4, 5))  
tab1000
```

```
# In[ ]:
```

```
# In[ ]:
```

```
# In[45]:
```

```
print(np.mean(tab1000))
```

```
# In[46]:
```

```
print(np.std(tab1000))
```

```
# # Q6/7
```

```
# In[47]:
```

```
np.linspace(3, 4, 5)
```

```
# In[48]:
```

```
np.arange(100, 1000, 10)
```

```
# In[49]:
```

```
# Q9  
b = np.arange(100, 1000, 10)  
print(b.shape)  
print(np.shape(b))
```

```
# # Q8
```

```
# In[50]:
```

```
np.linspace(100, 200, 5)
```

```
# In[51]:
```

```
np.arange(100, 201, 25)
```

```
# In[52]:
```

```
l = [100, 125, 150, 175, 200]
```

```
np.array(1)
# ou directement np.array([100, 125, 150, 175, 200])

# In[53]:

np.arange(5) * 25 + 100

# # Q10....

# In[54]:

print(np.shape( np.array([4,5]) ))

# In[58]:

# comment creer tableau avec 1 ligne, 2 colonnes
#np.zeros((1,2))
np.array( [ [4, 5] ] )

# In[62]:

# np.zeros etc acceptent des entiers à la place de shape
np.testing.assert_allclose(
    np.zeros(100),
    np.zeros((100,))
)
print("OK")

# In[64]:

print(np.zeros(5))
print("----")
print(np.zeros((1,5)))
print("----")
print(np.zeros((2,5)))

# In[66]:

print(np.shape( np.zeros((4, 5)) ))

# In[68]:

print(np.shape( np.ones((5, 4)) ))

# In[69]:

print(np.shape( np.random.uniform(2, 3, (4, 5)) ))

# In[70]:

print(np.shape( np.random.normal(2, 3, (4, 5)) ))

# In[72]:

print(np.shape( np.linspace(10, 20, 30) ))

# In[73]:

print(np.shape( np.arange(100, 1000, 10) ))
```

```
# In[76]:
```

```
print(np.shape( np.ones((100,      )) ))
```

```
# # Q18
```

```
# In[84]:
```

```
a = np.zeros((5,))
print(a)
print(np.shape(a))
a
```

```
# In[85]:
```

```
a = np.zeros((5, 1))
print(a)
print(np.shape(a))
a
```

```
# In[86]:
```

```
a = np.zeros((1, 5))
print(a)
print(np.shape(a))
a
```

```
# # Q19
```

```
# In[87]:
```

```
print(np.shape( np.ones((100)).reshape(( 5, 20))      ))
```

```
# In[88]:
```

```
print(np.shape( np.ones((100)).reshape(( -1, 20))      ))
```

```
# In[90]:
```

```
print(np.shape( np.ones((100)).reshape(( 5, -1))      ))
```

```
# In[91]:
```

```
print(np.shape( np.ones((100)).reshape(( -1, 5))      ))
```

```
# In[93]:
```

```
np.linspace(2, 5, 16)
```

```
# In[94]:
```

```
np.linspace(2, 5, 16).reshape((2, 8))
```

```
# In[95]:
```

```
np.linspace(2, 5, 16).reshape((-1, 4))
```

```
# In[ ]:
```

```
# In[ ]:
```

```
# In[ ]:
```