1.3.1 ЗАДАНИЕ Создать 8x8 матрицу м заполнить ее в шахматном порядке нулями и единицами

In [8]:
```
import numpy as np
# arr = np.zeros(64).reshape(8,8)
# for i in range(0, 8, 2):
#     arr[0,i+1] = 1
#     arr[1,i] = 1
#     arr[2,i+1] = 1
#     arr[3,i] = 1
#     arr[4,i+1] = 1
#     arr[5,i] = 1
#     arr[6,i+1] = 1
#     arr[7,i] = 1
# print(arr)

arr = np.zeros(64).reshape(8,8)
for i in range(0, 8, 2):
    for j in range(1, 9, 2):
        arr[i, j] = 1
        arr[j, i] = 1
print(arr)
```

```
[[0. 1. 0. 1. 0. 1. 0. 1.]
 [1. 0. 1. 0. 1. 0. 1. 0.]
 [0. 1. 0. 1. 0. 1. 0. 1.]
 [1. 0. 1. 0. 1. 0. 1. 0.]
 [0. 1. 0. 1. 0. 1. 0. 1.]
 [1. 0. 1. 0. 1. 0. 1. 0.]
 [0. 1. 0. 1. 0. 1. 0. 1.]
 [1. 0. 1. 0. 1. 0. 1. 0.]]
```

1.3.2 ЗАДАНИЕ Создать матрицу 5x5 со значениями в строках от 0 до 4. Использовать arange

In [9]:
```
oclobystin = np.zeros(25).reshape(5, 5)
goida = np.arange(0 ,5)
for i in range(5):
    oclobystin[i] = goida
print(oclobystin)
```

```
[[0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]]
```

1.3.3 ЗАДАНИЕ Создать массив 3x3x3 со случайными значениями.

In [10]:
```
print(np.random.random((3, 3, 3)))
```

```
[[[0.6925084  0.02295902 0.69449925]
  [0.03662901 0.95344352 0.6469757 ]
  [0.044226   0.69124208 0.47524574]]

 [[0.49180467 0.91973779 0.25164349]
  [0.46218846 0.01640886 0.01080114]
  [0.12521841 0.14441493 0.89409107]]

 [[0.92280794 0.36319188 0.04069514]
  [0.8204002  0.11394873 0.56678833]
  [0.05710176 0.31973602 0.07271078]]]
```

1.3.4 ЗАДАНИЕ Создать матрицу с 0 внутри, и 1 на границах

In [11]:
```
arr = np.zeros(9).reshape(3, 3)
for i in range(0, 3):
    arr[0, i] = 1
    arr[2, i] = 1
    arr[i, 0] = 1
    arr[i, 2] = 1
print(arr)
```

```
[[1. 1. 1.]
 [1. 0. 1.]
 [1. 1. 1.]]
```

1.3.5 ЗАДАНИЕ Создайте массив и отсортируйте его по убыванию

In [12]:
```
arr = [1, 7, 4, 2, 6, 9, 3, 0]
print(list(reversed(sorted(arr))))
```

```
[9, 7, 6, 4, 3, 2, 1, 0]
```

1.3.6 ЗАДАНИЕ Создайте матрицу, выведите ее форму, размер и размерность

In [13]:
```
arr = np.array([[1, 2], [3, 4], [5, 6]])
print(arr.shape)
print(arr.size)
print(arr.ndim)
```

```
(3, 2)
6
2
```

БИБЛИОТЕКА PANDAS

2.2.1 - Создать Series из списка Python, словаря Python, массива Numpy (установить буквенные метки для последнего)

In [14]:
```
import pandas as pd
lst = [1, 2, 3, 4, 5]
d = {'a':1, 'b':2, 'c':3}
ndarr = np.array([1, 2, 3, 4, 5])

s1 = pd.Series(lst)
s2 = pd.Series(d)
s3 = pd.Series(ndarr, ['a', 'b', 'c', 'd', 'e'])

print(s1)
```

```
print(s2)
print(s3)
```

```
0    1
1    2
2    3
3    4
4    5
dtype: int64
a    1
b    2
c    3
dtype: int64
a    1
b    2
c    3
d    4
e    5
dtype: int32
```

2.2.2 - Дано два Series. Напечатать их первые элементы и все элементы после третьего (во втором фрейме)

In [15]:
```
s1 = pd.Series([1, 2, 3, 4, 5], ['a', 'b', 'c', 'd', 'e'])
s2 = pd.Series([5, 4, 3, 2, 1])
print(s1['a'])
print(s2[0])
print(s2[3:])
```

```
1
5
3    2
4    1
dtype: int64
```

2.2.3 - Создайте новый фрейм данных

In [29]:
```
dataframe = pd.DataFrame()
dataframe['Имя'] = ['Джеки Джексон', 'Стивен Стивенсон', 'Travis Scott', 'Андрей
dataframe['Возраст'] = [38, 25, 1000, 52]
dataframe['Водитель'] = [True, False, False, True]
dataframe['Ведущий Программы "Субботний Вечер"'] = [False, False, False, True]
dataframe['МУЖЫК'] = [True, True, True, True]
dataframe
```

Out[29]:

	Имя	Возраст	Водитель	Ведущий Программы "Субботний Вечер"	МУЖЫК
0	Джеки Джексон	38	True	False	True
1	Стивен Стивенсон	25	False	False	True
2	Travis Scott	1000	False	False	True
3	Андрей Малахов	52	True	True	True

2.2.4 - Загрузите фрейм данных по ссылке

In [ ]:
```
# Создать URL-адрес
url = 'https://raw.githubusercontent.com/chrisalbon/simulated_datasets/master/ti
# Загрузить данные
dataframe = pd.read_csv(url)
# Показать пять строк
dataframe.head(5)
# ERROR 404((
```

2.2.5 - Проанализировать характеристики фрейма данных.

In [ ]:
```
dataframe.head(2)
dataframe.tail(3)
dataframe.shape
dataframe.describe()
```

2.2.6 - Выбор нескольких строк фрейма данных

In [ ]:
```
dataframe.iloc[1:4]
```

2.2.7 - Отобрат строки фрейма данных на основе некоторого условия

In [ ]:
```
dataframe[dataframe['PClass'] == '1st'].head(2)
```

2.3.1 - Найдите евклидово расстояние между двумя Series (точками) a и b, не используя встроенную формулу.

In [6]:
```
a = pd.Series([10, 18, 1488], ['x', 'y', 'z'])
b = pd.Series([0, 0, 0], ['x', 'y', 'z'])

L = ((a['x']-b['x'])**2 + (a['y']-b['y'])**2 + (a['z']-b['z'])**2)**0.5
print(L)
```

```
1488.14246629817
```

2.3.2 - Сформировать фрейм из csv файла

In [ ]:
```
# https://github.com/akmand/datasets/blob/main/arrhythmia.csv

url = 'https://github.com/akmand/datasets/blob/main/arrhythmia.csv'
dataframe = pd.read_csv(url)
dataframe
```

2.3.3 - то же, что и в примерах 2.2.5 - 2.2.7

In [ ]:
```
dataframe.head(2)
dataframe.tail(3)
dataframe.shape
dataframe.describe()

dataframe.iloc[1:4]

dataframe[dataframe['PClass'] == '1st'].head(2)
```

РАБОТА С ЧИСЛОВЫМИ ДАННЫМИ

3.2.1 - Прошкалируйте числовой признак в диапазон между двумя значениями.

In [16]:
```
# Загрузить из библиотеки
import numpy as np
from sklearn import preprocessing

# Создать признак
feature = np.array([[-500.5], [-100.1], [0], [100.1], [900.9]])
# Создать шкалировщик
minmax_scale = preprocessing.MinMaxScaler(feature_range = (0, 1))

# Прошкалировать признак
scaled_feature = minmax_scale.fit_transform(feature)

# Показать прошкалированный признак
scaled_feature
```

Out[16]:
```
array([[0.        ],
       [0.28571429],
       [0.35714286],
       [0.42857143],
       [1.        ]])
```

3.2.2 - Преобразуйте признак, чтобы он имел среднее значение 0 и стандартное отклонение 1.

In [27]:
```
x = np.array([[-1000.1], [-200.2], [500.5], [600.6], [9000.9]])
# Создать шкалировщик
scaler = preprocessing.StandardScaler()
# Преобразовать признак
standardized = scaler.fit_transform(x)
# Показать признак
standardized

# Мы можем увидеть эффект стандартизации, обратившись к среднему значению и стан
# print("Среднее: ", round(standardized.mean()))
# print("Стандартное отклонение: ", standardized.std())
```

Out[27]:
```
array([[-0.76058269],
       [-0.54177196],
       [-0.35009716],
       [-0.32271504],
       [ 1.97516685]])
```

3.2.3 - Дан фрейм данных. Необходимо масштабировать его числовые столбцы.

In [3]:
```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from sklearn import preprocessing

scaller = preprocessing.MinMaxScaler(feature_range=(0,1))
dfTest = pd.DataFrame({'A':[14.00,90.20,90.95,96.27,91.21],
                       'B':[103.02,107.26,110.35,114.23,114.68],
                       'C':['big','small','big','small','small']})

dfTest[['A', 'B']] = scaller.fit_transform(dfTest[['A', 'B']])
dfTest
```

Out[3]:

	A	B	C
0	0.000000	0.000000	big
1	0.926219	0.363636	small
2	0.935335	0.628645	big
3	1.000000	0.961407	small
4	0.938495	1.000000	small

3.3.2 - Загрузить фрейм данных по ссылке: https://raw.githubusercontent.com/akmand/datasets/master/iris.csv. Необходимо выполнить нормализацию первого числового признака (sepal_length_cm) с использованием минимаксного преобразования, а второго (sepal_width_cm) с задействованием z-масштабирования.

In [33]:
```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import StandardScaler
from sklearn import preprocessing

url = 'https://raw.githubusercontent.com/akmand/datasets/master/iris.csv'
dataframe = pd.read_csv(url)
minmax_scale = preprocessing.MinMaxScaler(feature_range = (0, 1))
standard_scaler = preprocessing.StandardScaler()

dataframe['sepal_length_cm'] = minmax_scale.fit_transform(dataframe)

dataframe['sepal_width_cm'] = standard_scaler.fit_transform(dataframe)
dataframe
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_10284\568058663.py in ?()
      7 dataframe = pd.read_csv(url)
      8 minmax_scale = preprocessing.MinMaxScaler(feature_range = (0, 1))
      9 standard_scaler = preprocessing.StandardScaler()
     10
---> 11 dataframe['sepal_length_cm'] = minmax_scale.fit_transform(dataframe)
     12
     13 dataframe['sepal_width_cm'] = standard_scaler.fit_transform(dataframe)
     14 dataframe

~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\sklearn\utils\_set_output.py in ?(self, X, *args, **kwargs)
    314     @wraps(f)
    315     def wrapped(self, X, *args, **kwargs):
--> 316         data_to_wrap = f(self, X, *args, **kwargs)
    317         if isinstance(data_to_wrap, tuple):
    318             # only wrap the first output for cross decomposition
    319             return_tuple = (

~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\sklearn\base.py in ?(self, X, y, **fit_params)
   1094                 )
   1095
   1096         if y is None:
   1097             # fit method of arity 1 (unsupervised transformation)
-> 1098             return self.fit(X, **fit_params).transform(X)
   1099         else:
   1100             # fit method of arity 2 (supervised transformation)
   1101             return self.fit(X, y, **fit_params).transform(X)

~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\sklearn\preprocessing\_data.py in ?(self, X, y)
    446             Fitted scaler.
    447         """
    448         # Reset internal state before fitting
    449         self._reset()
--> 450         return self.partial_fit(X, y)

~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\sklearn\base.py in ?(estimator, *args, **kwargs)
   1469                 skip_parameter_validation=(
   1470                     prefer_skip_nested_validation or global_skip_validation
   1471                 )
   1472             ):
-> 1473                 return fit_method(estimator, *args, **kwargs)

~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\sklearn\preprocessing\_data.py in ?(self, X, y)
    486
    487         xp, _ = get_namespace(X)
    488
    489         first_pass = not hasattr(self, "n_samples_seen_")
```

```
--> 490         X = self._validate_data(
    491             X,
    492             reset=first_pass,
    493             dtype=_array_api.supported_float_dtypes(xp),

~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\sklearn\base.py in ?(self, X, y, reset, validate_separately, cast_to_ndarray, **check_params)
    629                 out = y
    630             else:
    631                 out = X, y
    632         elif not no_val_X and no_val_y:
--> 633             out = check_array(X, input_name="X", **check_params)
    634         elif no_val_X and not no_val_y:
    635             out = _check_y(y, **check_params)
    636         else:

~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\sklearn\utils\validation.py in ?(array, accept_sparse, accept_large_sparse, dtype, order, copy, force_writeable, force_all_finite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, estimator, input_name)
   1009                         )
   1010                     array = xp.astype(array, dtype, copy=False)
   1011                 else:
   1012                     array = _asarray_with_order(array, order=order, dtype=dtype, xp=xp)
-> 1013             except ComplexWarning as complex_warning:
   1014                 raise ValueError(
   1015                     "Complex data not supported\n{}\n".format(array)
   1016                 ) from complex_warning

~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\sklearn\utils\_array_api.py in ?(array, dtype, order, copy, xp, device)
    741         # Use NumPy API to support order
    742         if copy is True:
    743             array = numpy.array(array, order=order, dtype=dtype)
    744         else:
--> 745             array = numpy.asarray(array, order=order, dtype=dtype)
    746
    747         # At this point array is a NumPy ndarray. We convert it to an array
    748         # container that is consistent with the input's namespace.

~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\pandas\core\generic.py in ?(self, dtype)
   2082     def __array__(self, dtype: npt.DTypeLike | None = None) -> np.ndarray:
   2083         values = self._values
-> 2084         arr = np.asarray(values, dtype=dtype)
   2085         if (
   2086             astype_is_view(values.dtype, arr.dtype)
   2087             and using_copy_on_write()

ValueError: could not convert string to float: 'setosa'
```