Активность лиганда. Молекулярные дескрипторы

Меры активности

- **EC50** - полумаксимальная эффективная концентрация, означает концентрацию лиганда, которая вызывает эффект, равный половине максимального возможного для данного лиганда
- **IC50** - является количественным индикатором, который показывает, сколько нужно лиганда—ингибитора для ингибирования биологического процесса на 50 %
- **Ki** - концентрация конкурентного лиганда, при которой он связывается с половиной мест связывания, имеющихся на реакционном субстрате, при условии отсутствия агониста

$$K_i = \frac{IC_{50}}{1 + \frac{[S]}{K_m}}$$

- Стандартные единицы измерения - **nM**

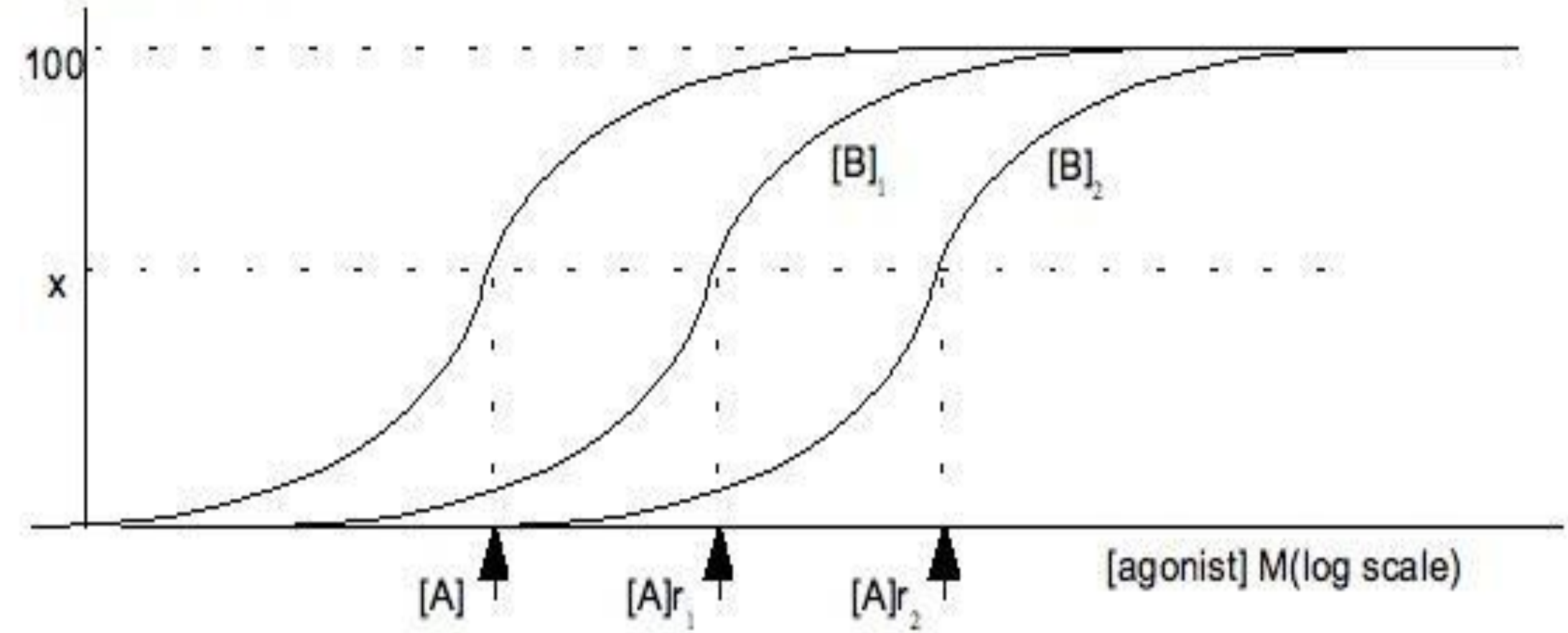
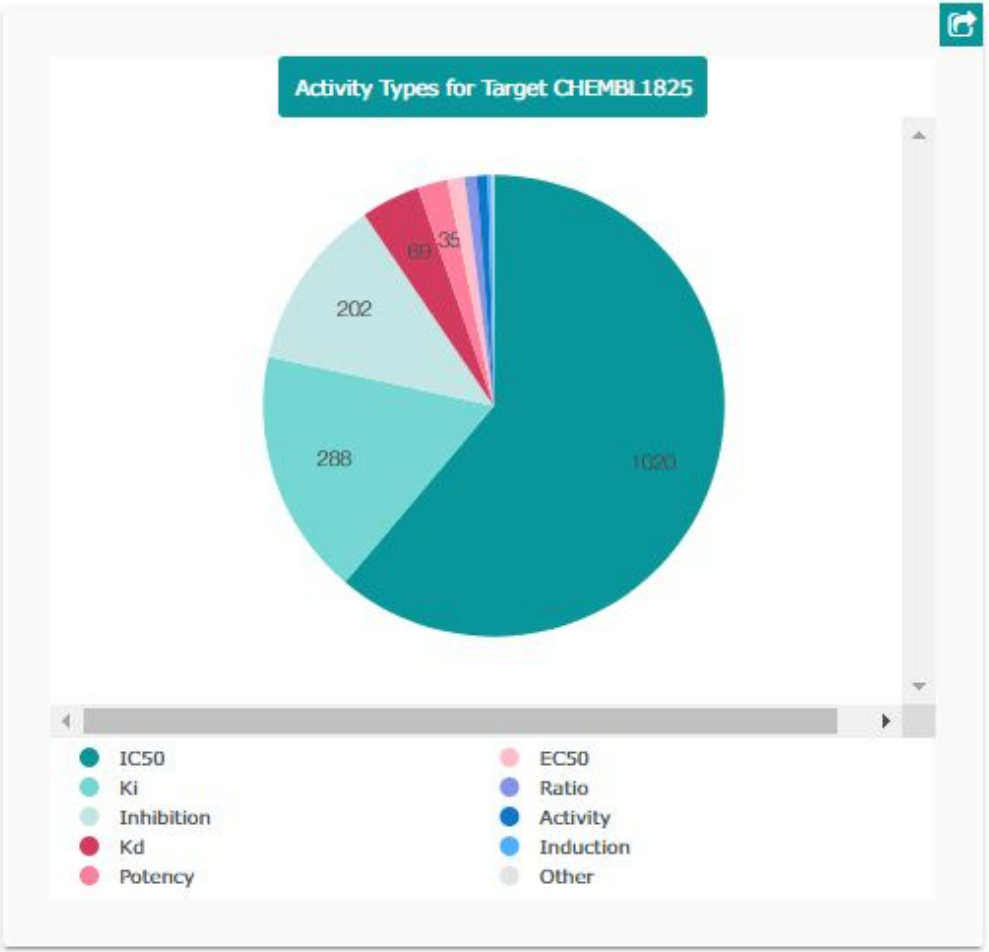


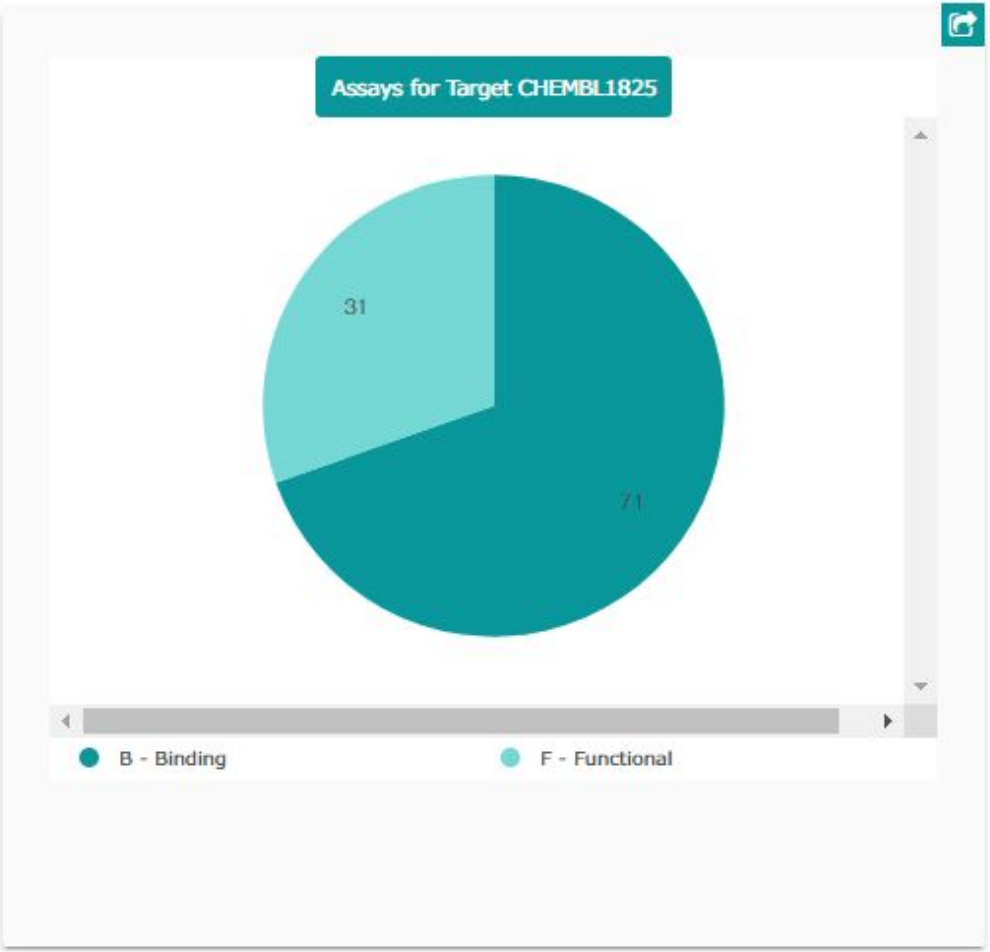
Figure 1 Diagrammatic representation of the dose response curves for two concentrations of reversibly competitive antagonist B on the effect of an agonist A. The value taken for response can be any value but typically 50% is used. The first curve is without antagonist, the second two are with increasing concentrations of antagonist

Выбираем свою меру активности

Associated Bioactivities



Associated Assays



In [1]:
```
import pandas as pd
```

In [2]:
```
data = pd.read_csv('Data.csv')
```

In [13]:
```
data.pivot_table(index = ['standard_type'], aggfunc='size')
```

Out[13]:
```
standard_type
Activity        3
IC50          463
Inhibition     67
Kd             59
Ki            288
dtype: int64
```

In [14]:
```
data = data.query("standard_type == 'IC50'")
```

In [17]:
```
data.head()
```

Out[17]:

	molecule_chembl_id	Smiles	standard_type	standard_value	standard_units	assay_type	assay_organism	ta
0	CHEMBL3640324	COc1ccc2c(c1F)C(=O)N(C[C@@]1(C#Cc3ccc(N4CCNCC4...	IC50	118.0	nM	B	Homo sapiens	
5	CHEMBL3640408	COc1ccc2c(c1)C(=O)N(C[C@@]1(C#Cc3ccc(NC(=O)C4C...	IC50	78.0	nM	B	Homo sapiens	

In [16]:
```
data.pivot_table(index = ['standard_units'], aggfunc='size')
```

Out[16]:
```
standard_units
nM    463
dtype: int64
```

Is it active ? Probably…

- IC50(compound 1) = 1000 nM
- IC50(compound 2) = 100000 nM

Как узнать активно соединение или нет?

- Посмотреть комментарий исследователя
- Приблизительно определить активность используя «стандартные» границы

Импорты

```
[1]: import pandas as pd
     from statistics import mean, mode, median
     import numpy as np
     import seaborn as sns
     from matplotlib import pyplot as plt
```

Давайте напишем функцию

```
def statDescribe (data):
    data_min = min(data)
    data_max = max(data)
    data_mean = round(mean(data), 2)
    data_mode = mode(data)
    data_median = median(data)
    print('Минимальное значение: {0} \nМаксимальное значение: {1} \nСреднее значение: {2} \nМода: {3} \nМедиана:{4}'\
          .format(data_min, data_max, data_mean, data_mode, data_median))
```

```
statDescribe(data['standard_value'])
```

```
Минимальное значение: 0.3
Максимальное значение: 20600000.0
Среднее значение: 122093.32
Мода: 20000.0
Медиана:678.0
```

pIC50 (pKi, pEC50)

pIC50 = - log(IC50)

Как это работает?

1. Переводим в моли(M): IC50 = 100,000 nM *10^-9= 10^-4
2. Находим отрицательный логарифм:
   - log(10^-4) = -(-4)log(10) = 4

3. Закономерность: чем выше pIC50, тем активнее средство.

Давайте писать еще больше функций!!

```
def calcPic50(st_values):
    for i in range(0, len(st_values)):
        st_values[i] = st_values[i]*10**-9
        st_values[i] = -np.log10(st_values[i])
    return st_values
```

```
st_val = list(data['standard_value'])
```

```
pic50_vals = calcPic50(st_val)
```

Добавим pIC50 в наш датафрейм

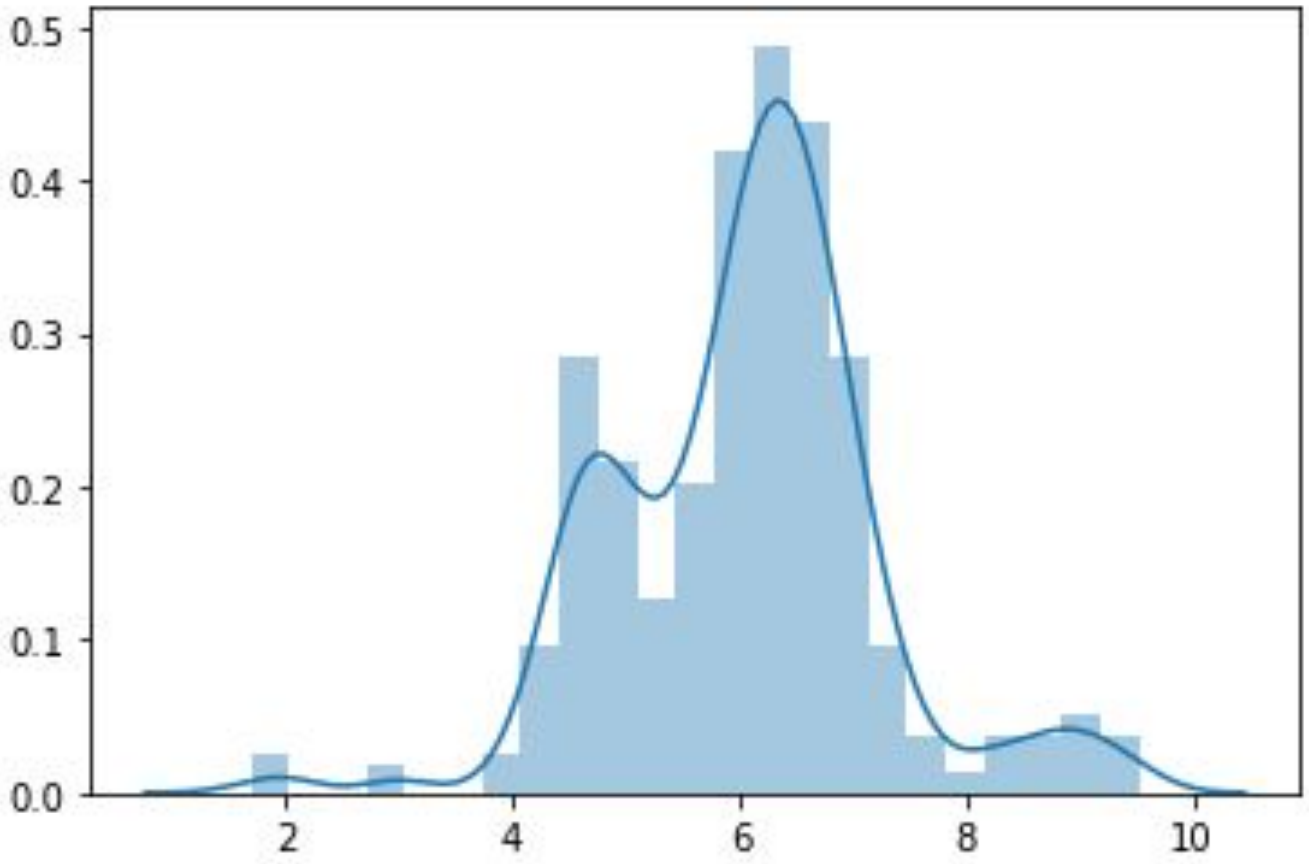
```
data['pIC50'] = pic50_vals
```

```
data.head()
```

Smiles	standard_type	standard_value	standard_units	assay_type	assay_organism	target_organism	pIC50
COc1ccc2c(c1F)C(=O)N(C[C@@]1(C#Cc3ccc(N4CCNCC4...	IC50	118.0	nM	B	Homo sapiens	Homo sapiens	6.928118
COc1ccc2c(c1)C(=O)N(C[C@@]1(C#Cc3ccc(NC(=O)C4C...	IC50	78.0	nM	B	Homo sapiens	Homo sapiens	7.107905
COc1ccc2c(c1)C(=O)N(CC1(C#Cc3ccc(/N=C(\NC#N)NC...	IC50	268.0	nM	B	Homo sapiens	Homo sapiens	6.571865
COc1ccc2c(c1)C(=O)N(C[C@@]1(C#Cc3ccc(C4(CN)NC(...	IC50	229.0	nM	B	Homo sapiens	Homo sapiens	6.640165
O=C1Nc2ccc(S(=O)(=O)Nc3ccc(O)c(Cl)c3)c3cccc1c23	IC50	45700.0	nM	B	Homo sapiens	Homo sapiens	4.340084

Распределение

```
plot = sns.distplot(pic50_vals)
```




```
statDescribe(pic50_vals)
```

```
Минимальное значение: 1.686
Максимальное значение: 9.523
Среднее значение: 6.06
Мода: 4.699
Медиана:6.169
```

«Стандартные» значения для IC50

- IC50 < 1000 nM – активный компонент (6.0 и больше)
- IC50 > 10000 nM – неактивный компонент (5.0 и меньше)

```
data = data.query("pIC50 >= 6.0")
data.head()
```

_id	Smiles	standard_type	standard_value	standard_units	assay_type	assay_organism	target_organism	pIC50
324	COc1ccc2c(c1F)C(=O)N(C[C@@]1(C#Cc3ccc(N4CCNCC4...	IC50	118.0	nM	B	Homo sapiens	Homo sapiens	6.928
408	COc1ccc2c(c1)C(=O)N(C[C@@]1(C#Cc3ccc(NC(=O)C4C...	IC50	78.0	nM	B	Homo sapiens	Homo sapiens	7.108
557	COc1ccc2c(c1)C(=O)N(CC1(C#Cc3ccc(/N=C(\NC#N)NC...	IC50	268.0	nM	B	Homo sapiens	Homo sapiens	6.572
442	COc1ccc2c(c1)C(=O)N(C[C@@]1(C#Cc3ccc(C4(CN)NC(...	IC50	229.0	nM	B	Homo sapiens	Homo sapiens	6.640
760	COc1cccc2c1ncc1c(=O)n(-c3cccc(Cl)c3)c(=O)n(C3C...	IC50	0.4	nM	B	Homo sapiens	Homo sapiens	9.398

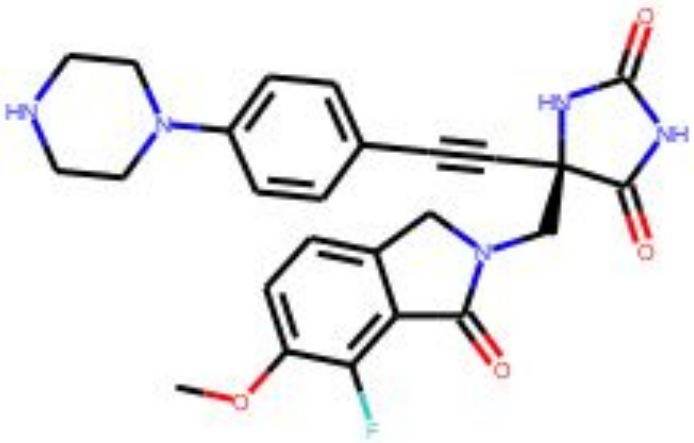
Введение в RDkit
Молекулярные дескрипторы

```
from rdkit import Chem
from rdkit.Chem import AllChem, Draw
```

```
smile = data['Smiles'][0]
smile
```

```
'COc1ccc2c(c1F)C(=O)N(C[C@@]1(C#Cc3ccc(N4CCNCC4)cc3)NC(=O)NC1=O)C2'
```

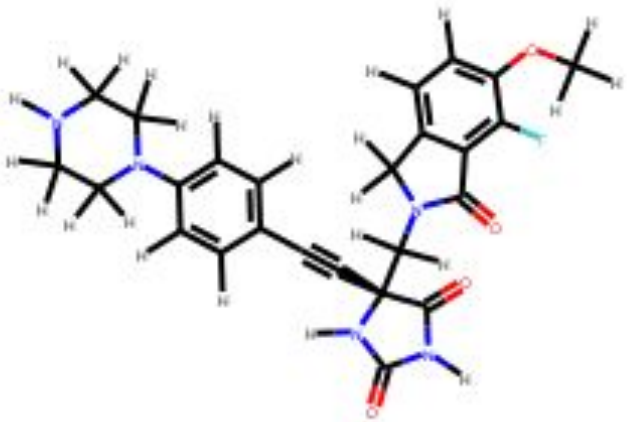
```
mol = Chem.MolFromSmiles(smile)
mol
```



Основной объект - Mol

[20]:
```
mol  = Chem.AddHs(mol)
mol
```

t[20]:



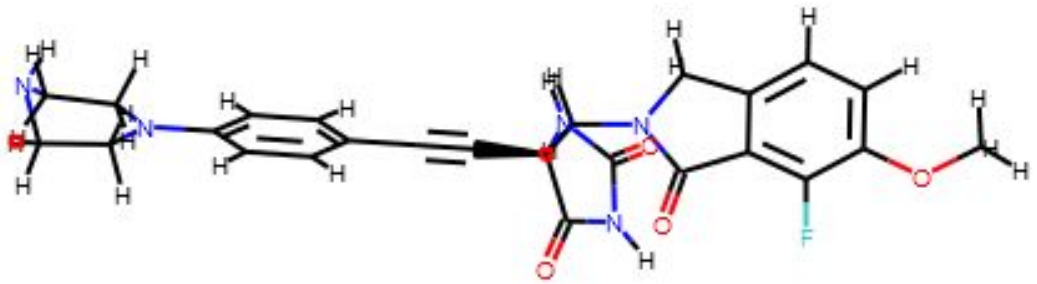
[21]:
```
print(Chem.MolToMolBlock(mol))
```

```
     RDKit          2D

 59 63  0  0  0  0  0  0  0  0999 V2000
    1.3299    5.6152    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -0.0967    5.1517    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
   -0.4086    3.6844    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.8351    3.2209    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -2.1470    1.7537    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.0323    0.7500    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    0.3943    1.2135    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
```

In [27]:
```
AllChem.EmbedMolecule(mol)
mol
```

Out[27]:



In [23]:
```
print(Chem.MolToMolBlock(mol))
```

```
     RDKit          3D

 59 63  0  0  0  0  0  0  0  0999 V2000
   10.5979    0.5390    0.0705 C   0  0  0  0  0  0  0  0  0  0  0  0
    9.3965    0.6764    0.8012 O   0  0  0  0  0  0  0  0  0  0  0  0
    8.1748    0.6172    0.1254 C   0  0  0  0  0  0  0  0  0  0  0  0
    8.1053    0.4275   -1.2397 C   0  0  0  0  0  0  0  0  0  0  0  0
    6.8657    0.3747   -1.8732 C   0  0  0  0  0  0  0  0  0  0  0  0
    5.6843    0.5052   -1.1900 C   0  0  0  0  0  0  0  0  0  0  0  0
    5.7487    0.6936    0.1657 C   0  0  0  0  0  0  0  0  0  0  0  0
    6.9823    0.7477    0.8071 C   0  0  0  0  0  0  0  0  0  0  0  0
    7.0508    0.9338    2.1473 F   0  0  0  0  0  0  0  0  0  0  0  0
```

Идея:

1. Объединить схожие молекулы в кластеры.
2. Найти центральные молекулы каждого кластера.
3. Построить модель фармакофора

Молекулярные дескрипторы

- **Одномерные (1D): Растворимость, logP, молекулярная масса**

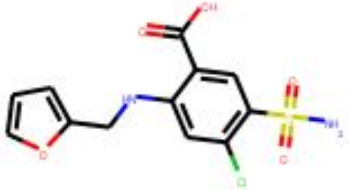
- Глобальный дескриптор: одно значение представляет все соединение
- Обычно недостаточны для применения в машинном обучении (ML)
- Может быть добавлен к 2D-дескрипторам для улучшения молекулярного кодирования в ML.

Doxycycline: 444.44 Da

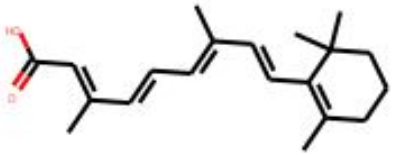
Tetracycline: 444.44 Da

Glycol dilaurate: 426.68 Da

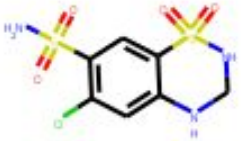
Amoxicilline: 365.41 Da



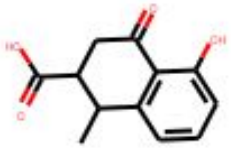
Furosemide: 330.75 Da



Isotretinoine: 300.44 Da



Hydrochlorothiazide: 297.75 Da



Hemi-cycline D: 220.22 Da

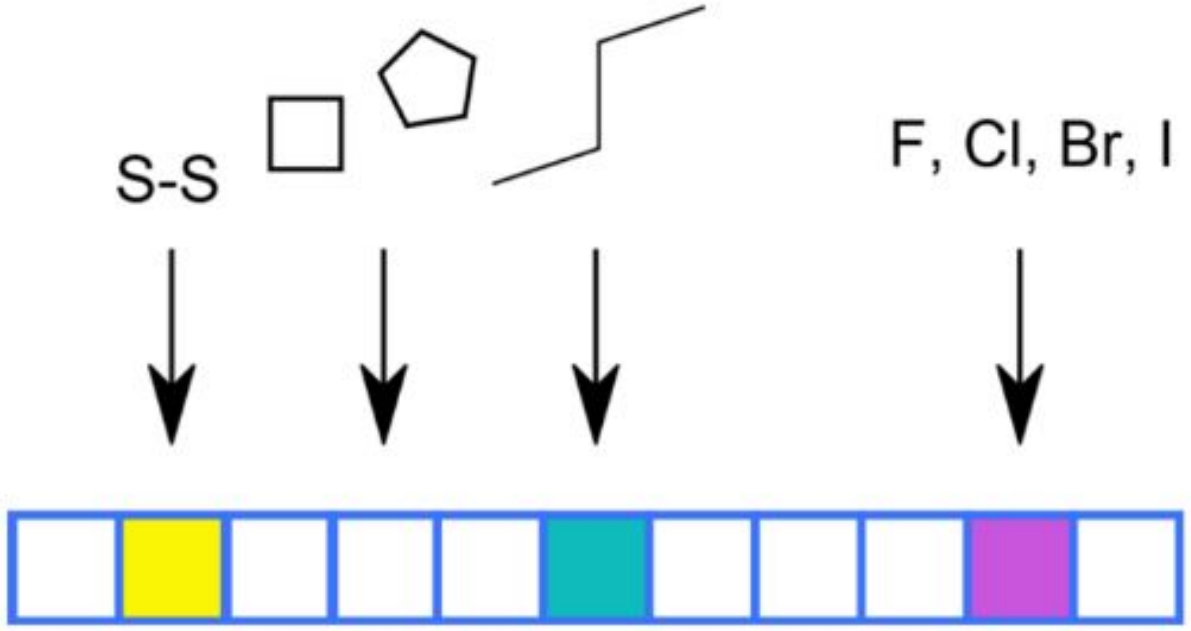
Молекулярные дескрипторы

- **Двухмерные (2D): молекулярные графы, фрагменты молекул, окружения атомов.**

- Подробное представление отдельных частей молекулы
- Обнаружение так называемых отпечатков пальцев (фингерпринтов)
- Очень часто используется в поиске сходства между молекулами и машинном обучении

Молекулярные дескрипторы

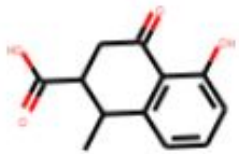
- Основная идея – представить молекулу в виде бит-строки, где каждый бит (1 или 0) соответствует наличию/отсутствию определенной структурной части молекулы (MACCS fingerp



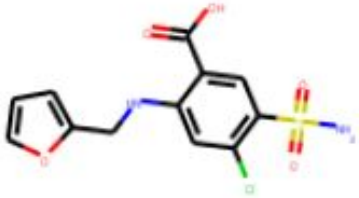
Query: Doxycycline (1.0)

#1: Tetracycline (0.93)

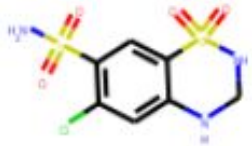
#2: Amoxicilline (0.59)



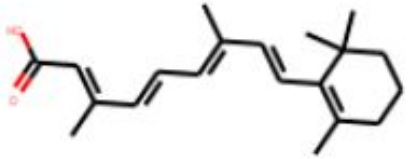
#3: Hemi-cycline D (0.4)



#4: Furosemide (0.32)



#5: Hydrochlorothiazide (0.31)

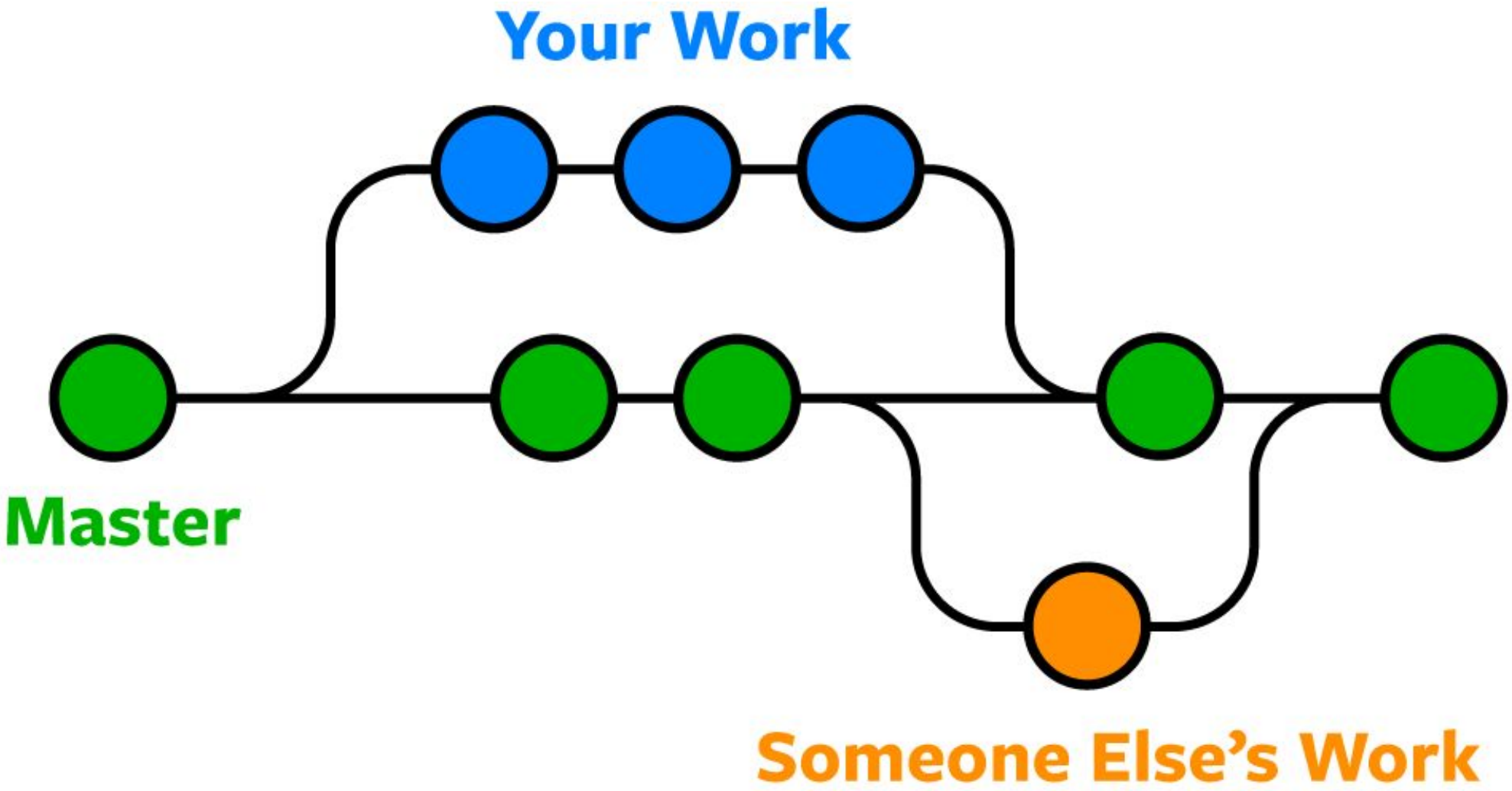


Последняя функция((

```
def describe_mol(mol):
    mw = round(Descriptors.MolWt(mol), 3)
    logP = round(Descriptors.MolLogP(mol), 3)
    des_obj = rdMolDescriptors.GetMACCSKeysFingerprint(mol)
    des = des_obj.ToBitString()
    print(f'Молекулярная масса: {mw} \nLogP: {logP} \nMACCSKeys: {des}')
```

```
describe_mol(mol)
```

Git



Git

- 1. Регистрируемся на GitHub
- 2. Создаем репозиторий
- 3. Скачиваем Git на PC
- 4. Логинимся
- 5. Клонируем репозиторий (git clone)
- 6. Копируем файлы проекта в репозиторий на PC
- 7. Проверяем статус (git status)
- 8. Добавляем файлы (подготовка коммита) (git add .)
- 9. Создаем коммит (git commit –m ‘initial commit’)
- 10. git push

Спасибо за внимание!!