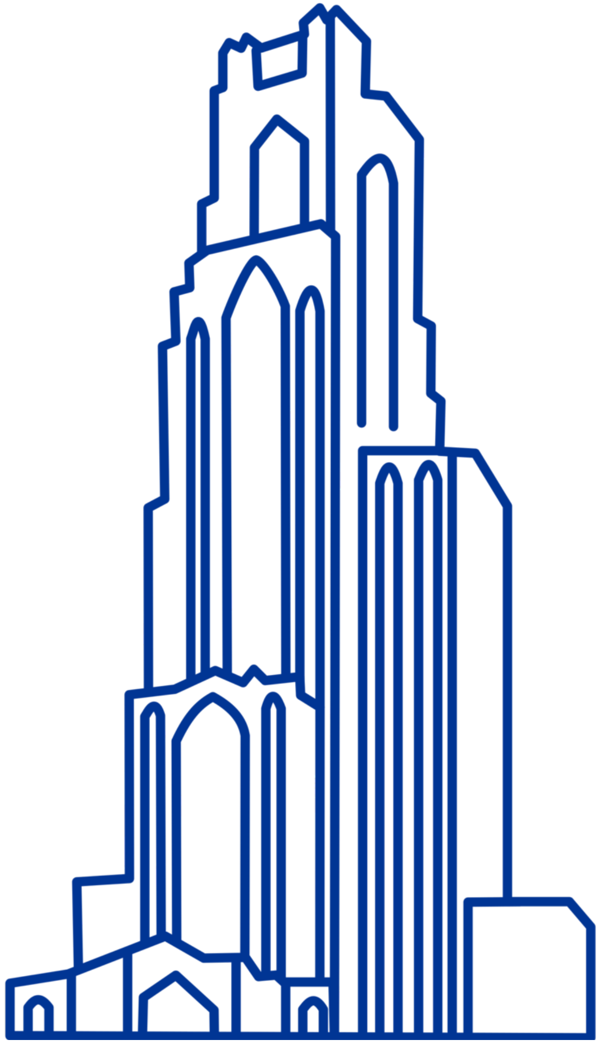
Computational Biology

(BIOSC 1540)



Lecture 13A

Cheminformatics

Foundations

Apr 8, 2025

University of Pittsburgh

Announcements

Assignments

- P03A is due tonight
- P04A will be due Apr 22

Quizzes

- Today is our last quiz

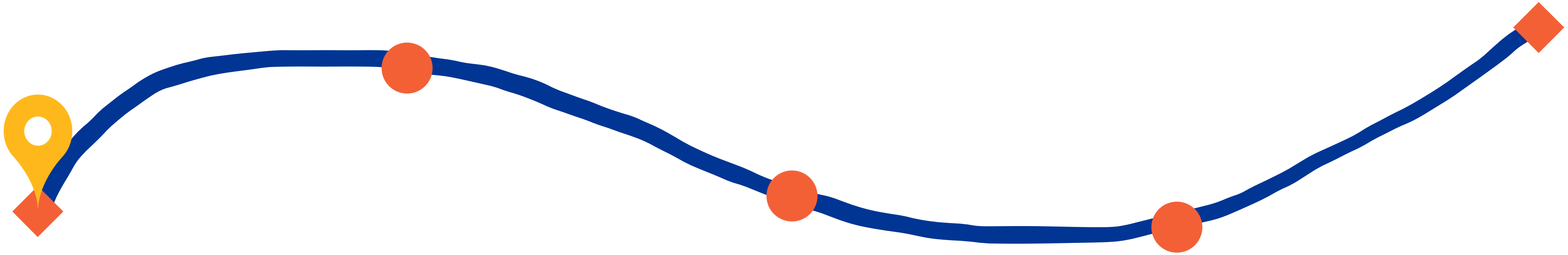
Final exam

- The final exam is on **Monday, Apr 28, at 4:00 pm in 244 Cathedral of Learning**

OMETs

- I will drop your lowest assignment if the response rate is 80% or higher.
- Current response rate: 63%

After today, you should have a better understanding of



Quiz 04

Please put away all materials as we distribute the quiz

Sit with an empty seat between you and your neighbors for the quiz

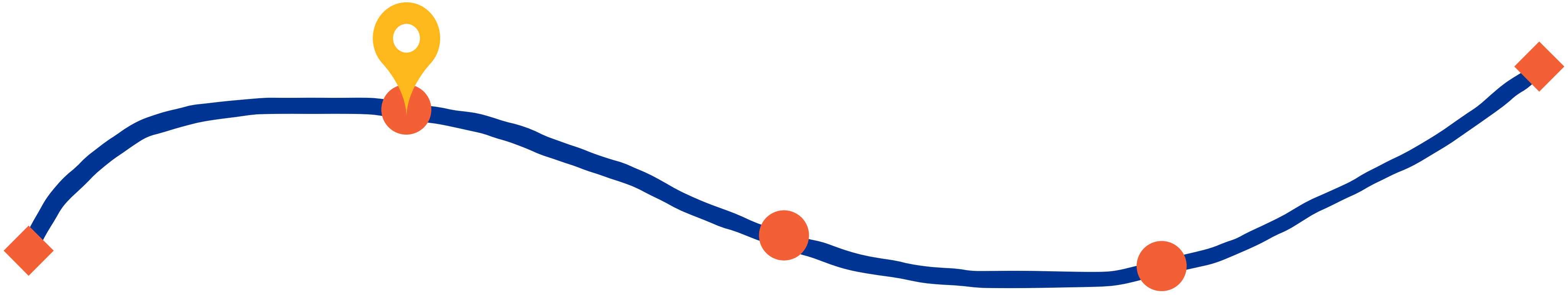
Fill out the cover page, and do not start yet

Quiz ends around **9:55 am**

https://www.clockfaceonline.co.uk/clocks/digital/

When you are finished, please hold on to your quiz and feel free to doodle or write anything on the last page

After today, you should have a better understanding of



Ligand-based drug design

Structural insight into a disease is a privilege

Phenotypic drug screening involves testing compounds on an organism level to identify potential leads

Example: Drug screening on an antibiotic-resistant bacterial strain to identify potential new leads

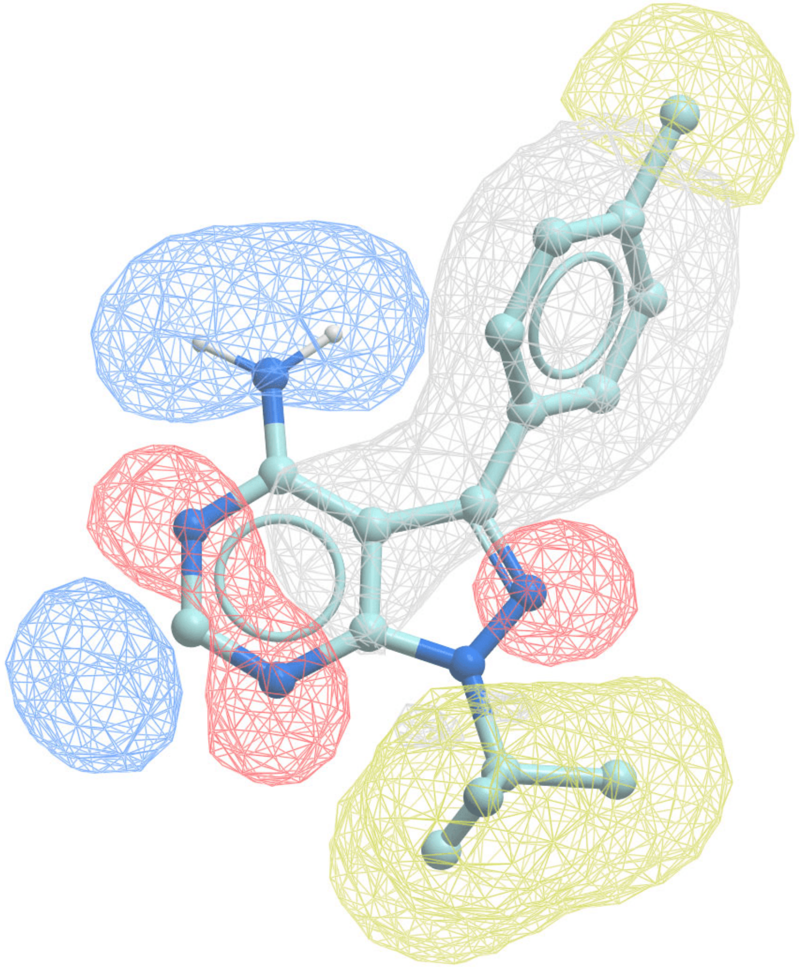
LBDD uses known compounds to guide drug discovery

Ligand-based drug design (LBDD) relies on the properties of known bioactive compounds

LBDD does not **require** the structure of the target protein, making it useful when this is unknown

Motivation: If we find compounds with little bioactivity, we can use LBDD to find compounds with similar chemical features to improve specific outcomes

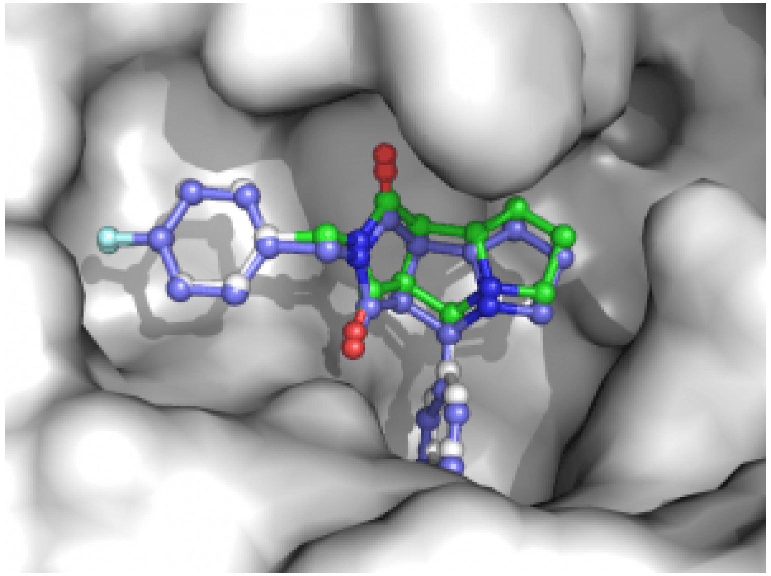
Assumption: Similar structures can lead to similar—hopefully improved—biological effects



Key differences between structure- and ligand-based drug design

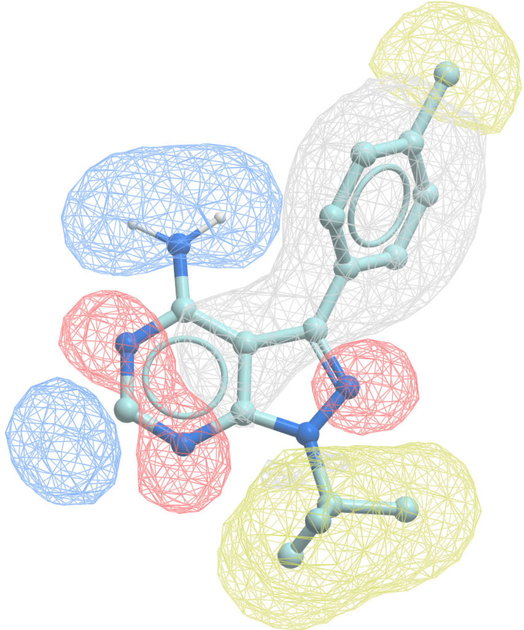
Structure-Based Drug Design:

- Requires 3D structure of the target protein.
- Uses the binding site structure to model potential interactions.
- Often employs docking and molecular simulations.



Ligand-Based Drug Design:

- Requires no structural information of the target.
- Uses the chemical structure and activity of known ligands as guides.
- Relies on molecular similarity rather than direct binding predictions.



Chemical space exploration is still challenging, and now we need to identify similar compounds

After today, you should have a better understanding of

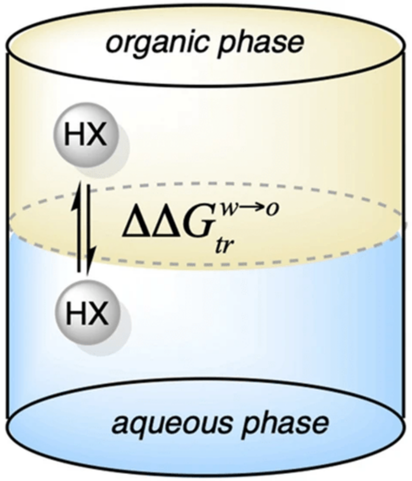
Molecular properties

Molecular properties are used to predict how a compound behaves in the body, before any biological testing

These properties help **prioritize molecules for synthesis and testing** by estimating solubility, permeability, bioavailability, and toxicity.

LogP quantifies lipophilicity, which affects absorption, distribution, and membrane permeability

LogP is the logarithm of a compound's **partition coefficient between octanol and water**.



High LogP values indicate **lipophilic (fat-loving) molecules** that may permeate membranes more easily, but also may have poor solubility and toxicity risks.

Low LogP values mean **hydrophilicity (water-loving)**, which helps with solubility but may hinder permeability.

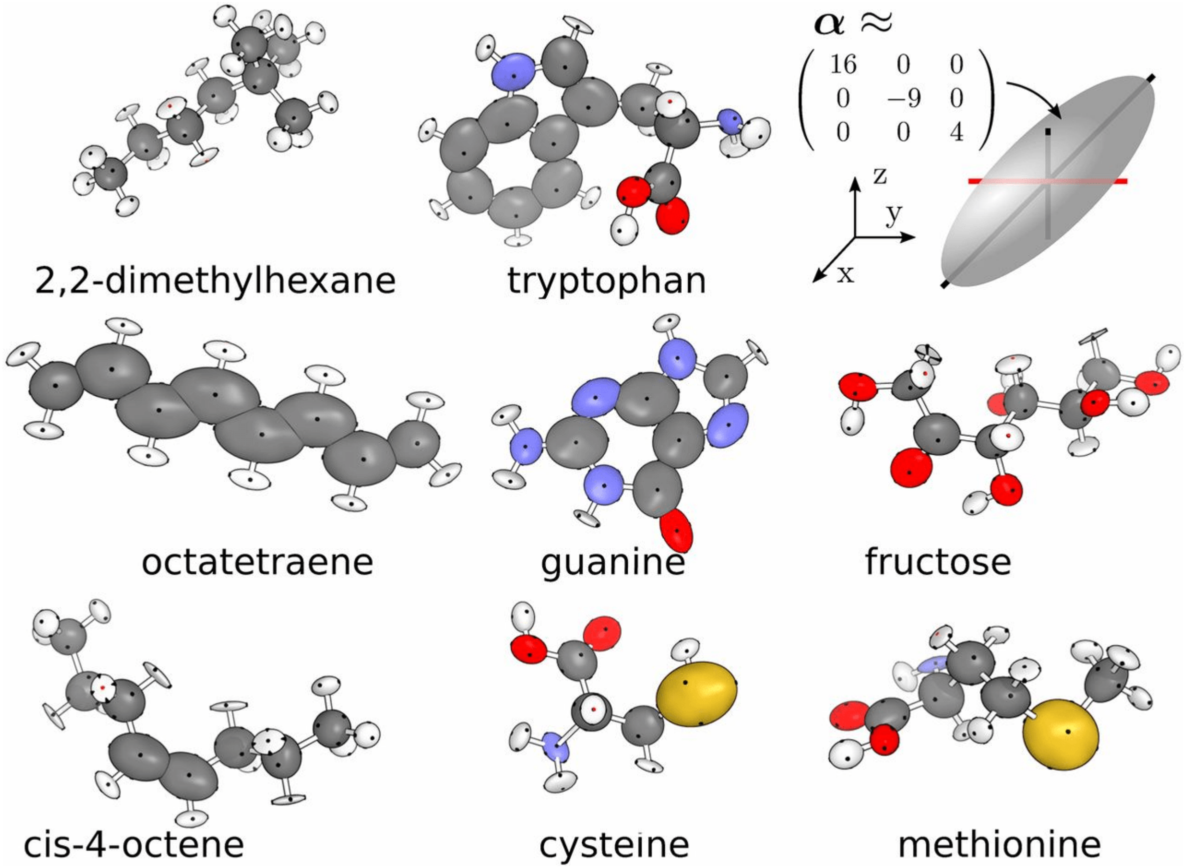
$$\log P = \log_{10}\left(\frac{[\text{solute}]_{\text{octanol}}}{[\text{solute}]_{\text{water}}}\right)$$

Molar refractivity (MR) measures polarizability and molecular volume

MR depends on molecular size and the type of atoms present.

Higher MR suggests **greater polarizability**, which can enhance binding via dispersion forces.

It is also used as a **proxy for molecular volume**—important in steric compatibility with binding pockets.

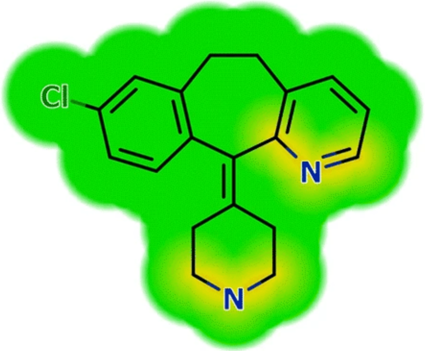


Topological Polar Surface Area (TPSA) predicts membrane permeability

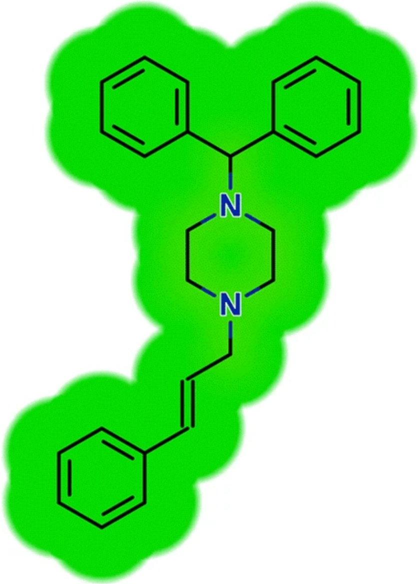
It is calculated from the **surface area of oxygen and nitrogen atoms** (and their attached hydrogens).

Molecules with **TPSA >140 Å² typically show poor oral bioavailability**.

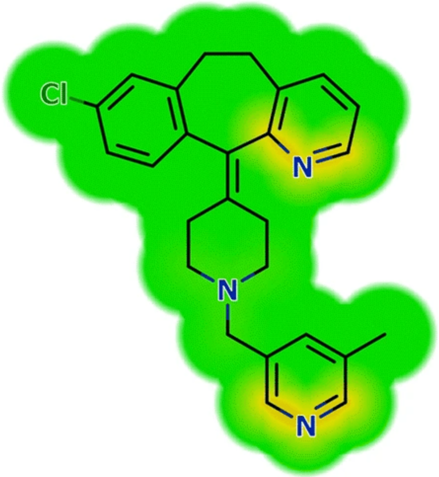
Lower **TPSA values (<90 Å²)** suggest good potential for **crossing the blood-brain barrier (BBB)**.



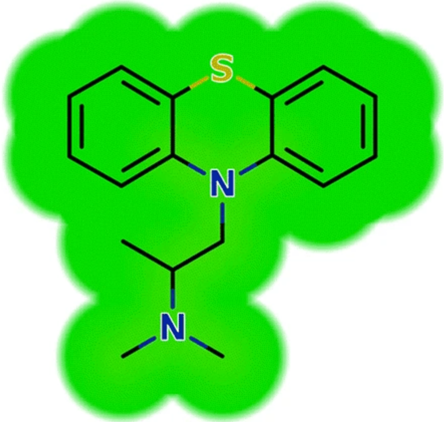
TPSA = 24.92



TPSA = 6.48



TPSA = 29.02

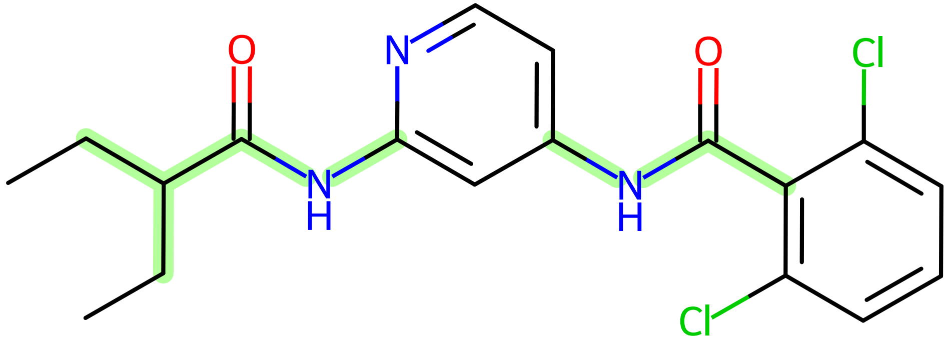


TPSA = 6.48

Rotatable bonds contribute to molecular flexibility

Fewer rotatable bonds generally mean better oral bioavailability and metabolic stability.

Highly flexible molecules may pay a greater entropic cost upon binding, reducing affinity.



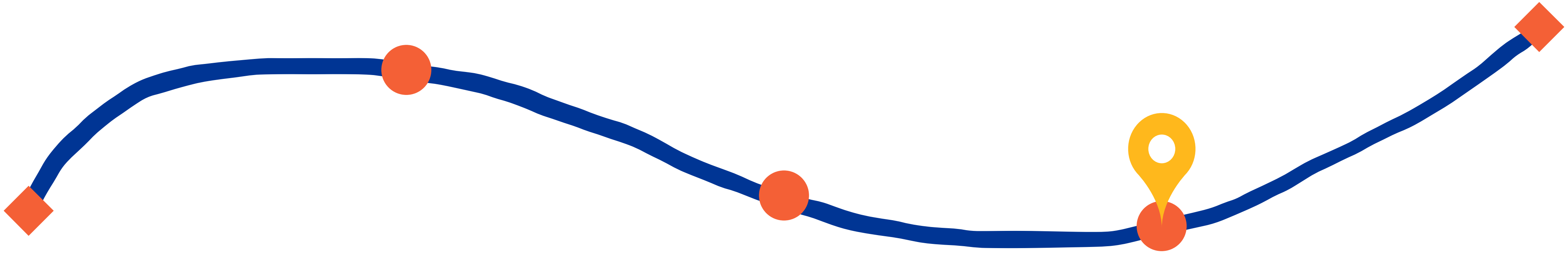
Drug-like molecules often have **fewer than 10 rotatable bonds**.

While molecular properties provide crucial insight, they do not fully describe a molecule’s structure or function

Two compounds can have similar LogP, TPSA, and molecular weights—but behave very differently due to subtle structural variations (e.g., isomers or stereochemistry).

Properties are global summaries, but molecular similarity often depends on local structural features like functional groups, ring systems, or atom connectivity.

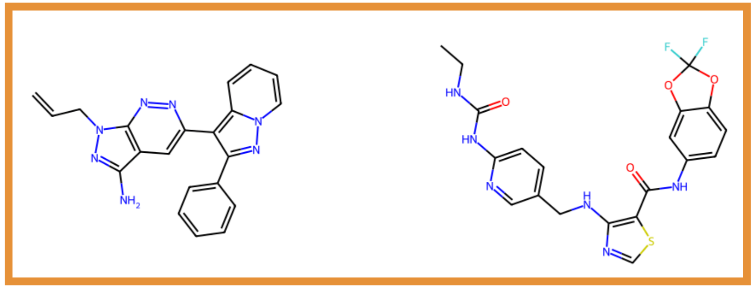
After today, you should have a better understanding of



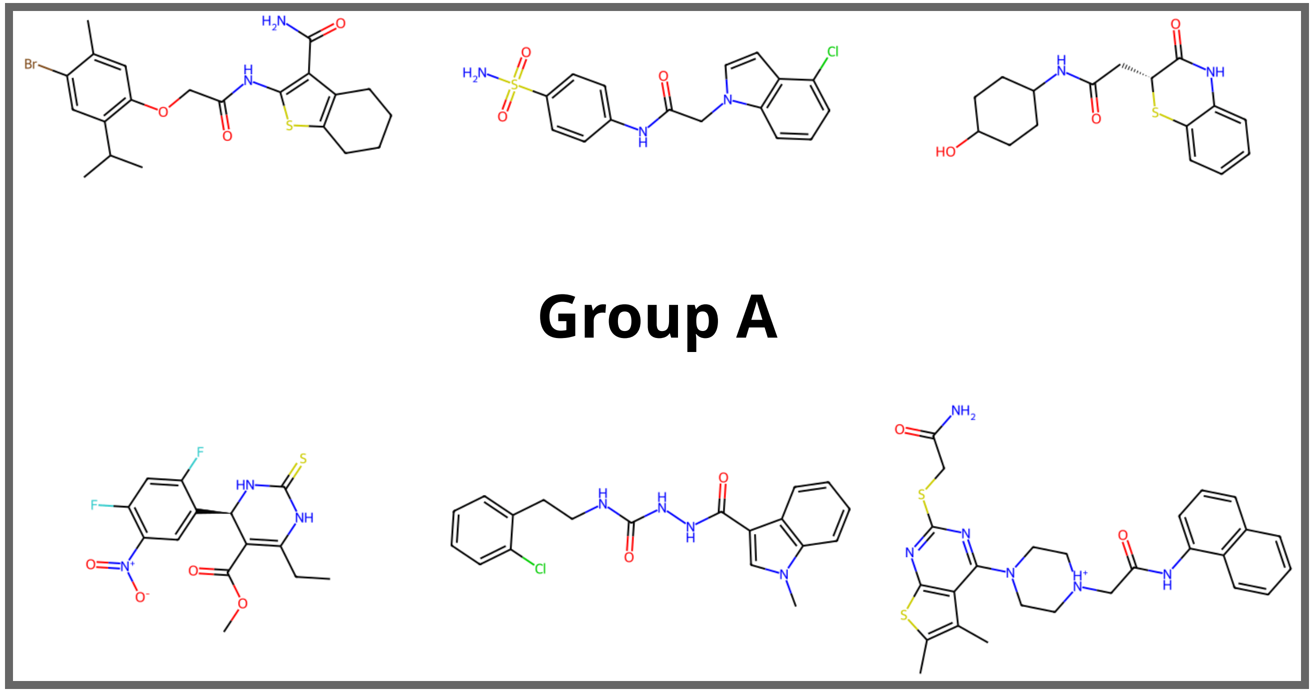
Molecular similarity

Quantifying molecular similarity is challenging

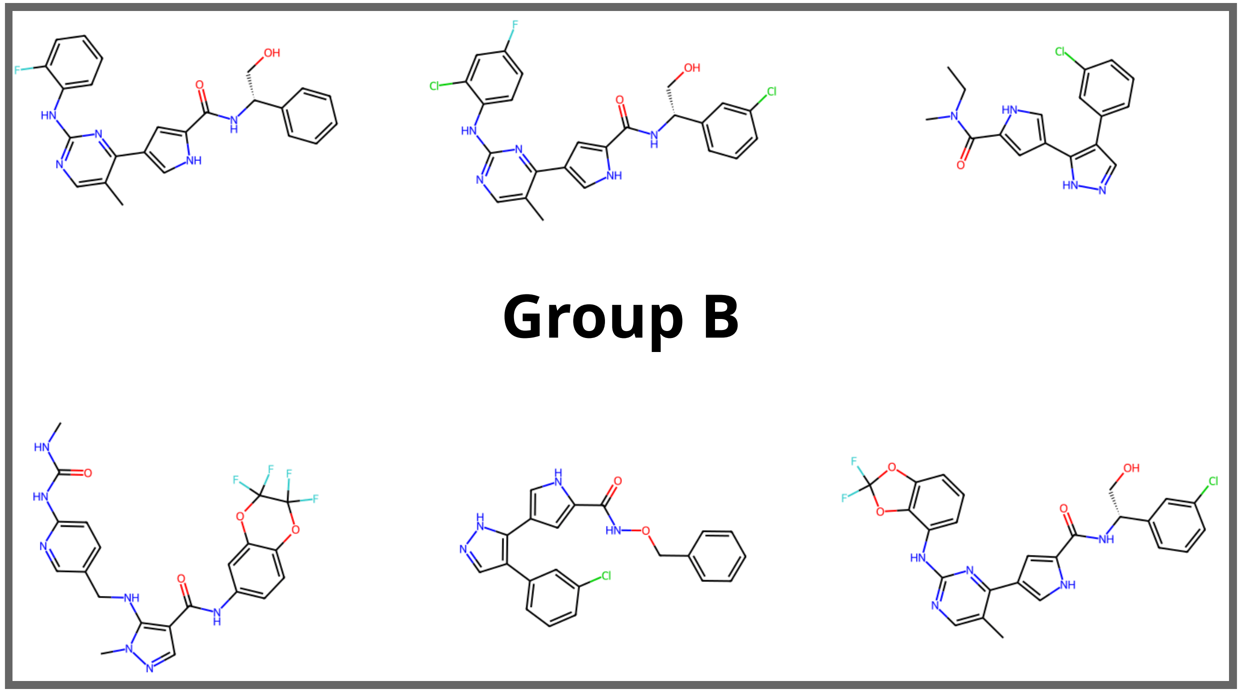
Suppose we performed an experimental high-throughput screen and identified these **potential leads**



Which group of molecules should we pursue for increased bioafinity?



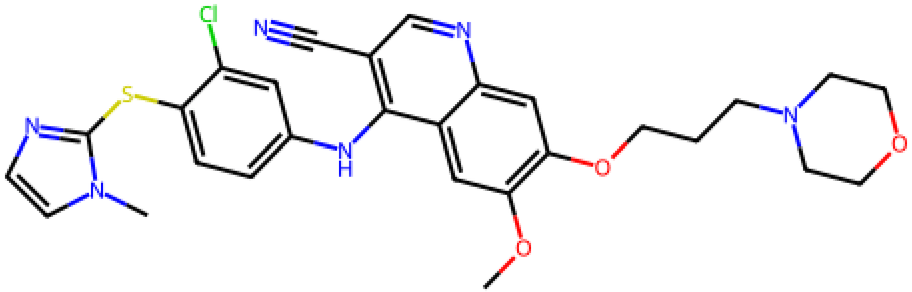
Group A

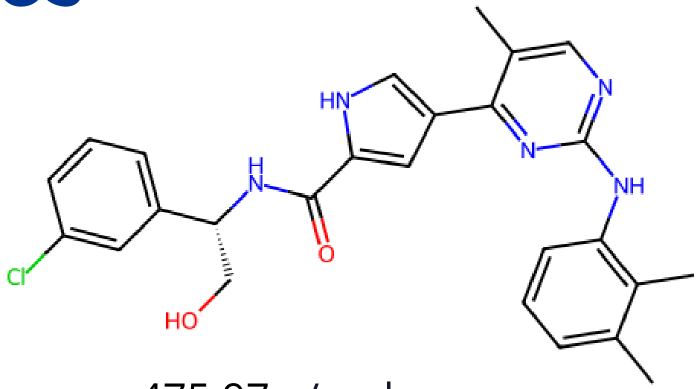


Group B

With your neighbors, determine how you would choose the group of molecules to pursue.

Molecular descriptors numerically encode chemical properties





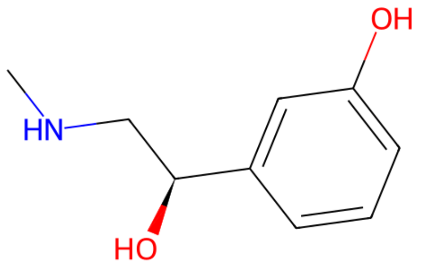
Molecular weight	565.09 g/mol	475.97 g/mol
Indicates the overall size of the molecule, impacting drug distribution and elimination rates in the body.		
LogP	4.08	4.30
Measures lipophilicity, which influences a molecule's ability to cross cell membranes and affects absorption and bioavailability.		
Molar Refractivity	156.23	134.72
Relates to polarizability and electron cloud distribution, affecting intermolecular interactions and binding affinity.		
TPSA	122.76 Å²	102.93 Å²
Estimates the molecule’s ability to form hydrogen bonds, impacting solubility and permeability across biological membranes.		
Num. rotatable bonds	10	8
Reflects molecular flexibility, which can influence binding affinity and oral bioavailability.		

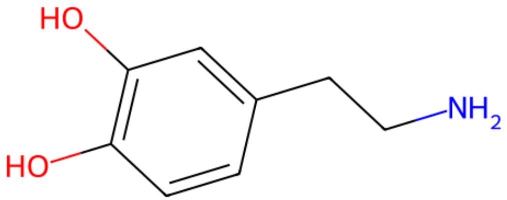
Computed with SwissADME

Molecules can have similar properties, with slight structural differences causing widely different functions

Phenylephrine is a synthetic compound that acts as a vasoconstrictor by stimulating alpha-adrenergic receptors

Dopamine is a naturally occurring neurotransmitter in the brain and interacts with dopamine receptors





	Phenylephrine	Dopamine
Molecular weight	167.21 g/mol	153.18 g/mol
LogP	0.65	0.46
Molar Refractivity	47.01	42.97
TPSA	52.49 Å^2	66.48 Å^2
Num. rotatable bonds	3	2
SMILES	`CNC[C@@H](C1=CC(=CC=C1)O)O`	`C1=CC(=C(C=C1CCN)O)O`

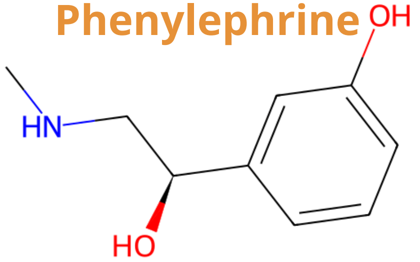
Simple descriptor comparisons are not sufficient for computing molecular similarity

Computed with SwissADME

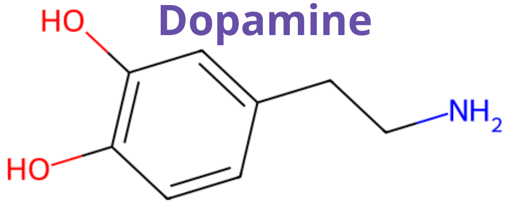
Molecular fingerprints encode structural information

Extended Connectivity Fingerprints (ECFPs) encode structural features into numerical representations

Phenylephrine

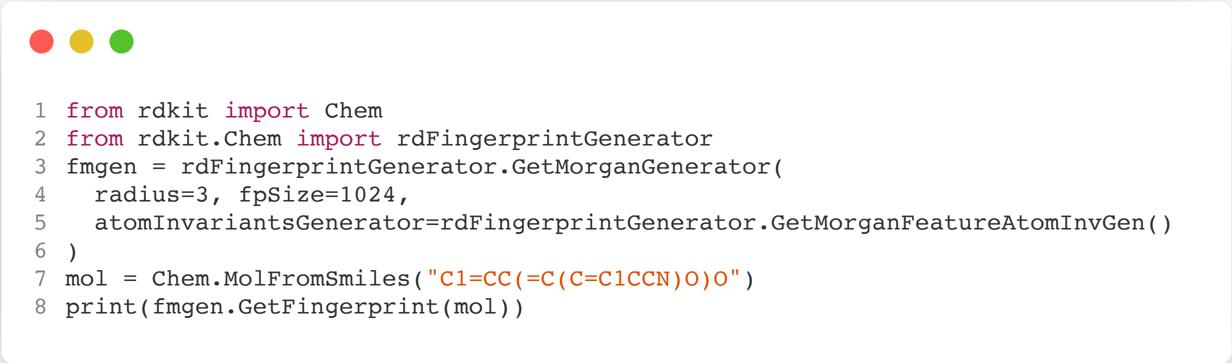


```
10011000000000000001000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000001000001000000000000000000001000000
00000000000000000000000000000000000000000000000000001000000000001000000000000000000000000000000000000000000000000010000000000000
00000000000000000000000000100000000000000000001000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000100000000000000000000000000000000000000000000000000001000000000000100000000000000000001000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000001000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000001000000000000100100000000000000000000000000001000000001000000100000000000000000000000000
00000100000000000000001000000000000000000000000000000000000000000000000000000000000000000000000001000000000000000001001001000000
```



Dopamine

```
10011000000000000001000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000010000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000100000000000000000000000000000000000000010000000100000010000000000000000000000000010
00000000000000000000100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000001000000000000000000000000010000000000000000000000
00000000000000000000000000000000000010001000000000100101000000000000000000000000000000000100000000000000000000000000000000000000
00000100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000001000000000000000001001000000000
```



```python
from rdkit import Chem
from rdkit.Chem import rdFingerprintGenerator
fmgen = rdFingerprintGenerator.GetMorganGenerator(
  radius=3, fpSize=1024,
  atomInvariantsGenerator=rdFingerprintGenerator.GetMorganFeatureAtomInvGen()
)
mol = Chem.MolFromSmiles("C1=CC(=C(C=C1CCN)O)O")
print(fmgen.GetFingerprint(mol))
```

How do we compute this?

Hash functions are used to encode chemical information

"Encoding" is a computational term for transforming information in a numerical format for computers

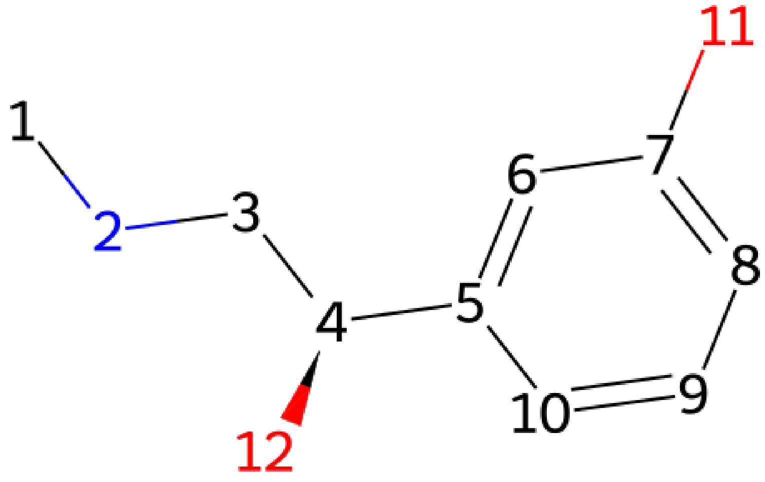
For each heavy atom (i.e., not H), hash atom-specific properties

$$ID_0 = \text{hash}(Z_i, V_i, C_i, R_i, \ldots)$$

ID_0

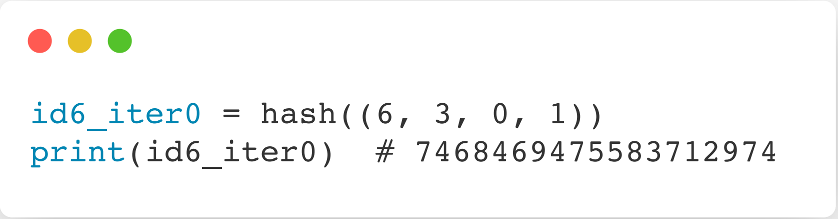
Iteration 0 identifier

Z	Atomic number
V	Valence
C	Formal charge
R	Ring membership

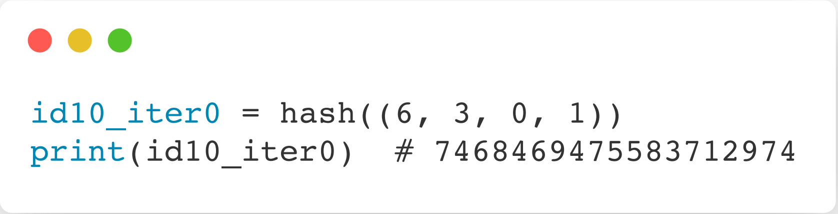


Let's look at carbons 6 and 10

Because of the same element and connectivity, they have the same ID_0



```
id6_iter0 = hash((6, 3, 0, 1))
print(id6_iter0)  # 7468469475583712974
```

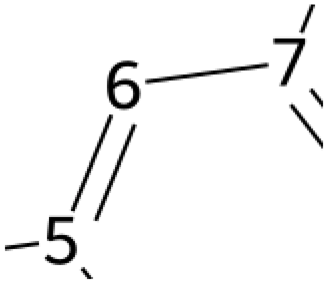


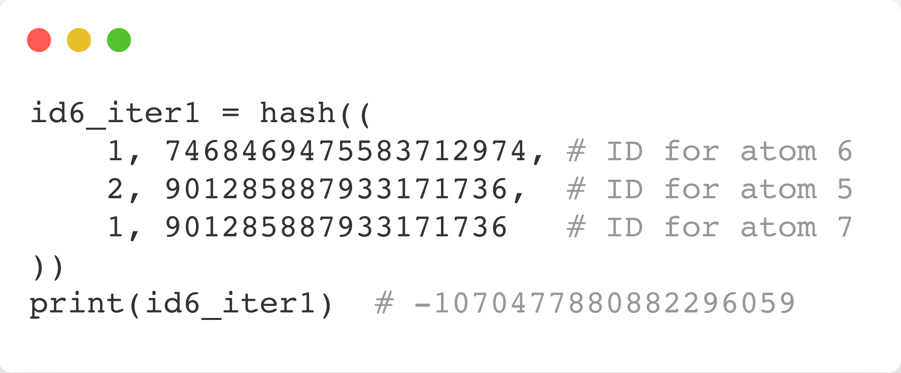
```
id10_iter0 = hash((6, 3, 0, 1))
print(id10_iter0)  # 7468469475583712974
```

For each additional iteration of *n*, incorporate the hashes of connected atoms that are *n* bonds away

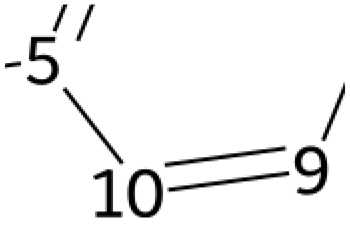
Next, encode the atom IDs that are exactly one bond away

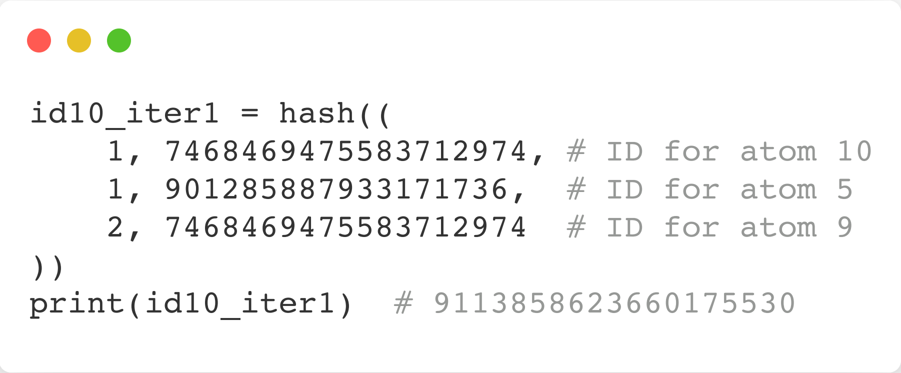
Format: `(IterationNumber, AtomID, BondOrder1, AtomID1, BondOrder2, AtomID2, ...)`





```
id6_iter1 = hash((
    1, 7468469475583712974, # ID for atom 6
    2, 9012858879331717336,  # ID for atom 5
    1, 9012858879331717336   # ID for atom 7
))
print(id6_iter1)  # -1070477880882296059
```





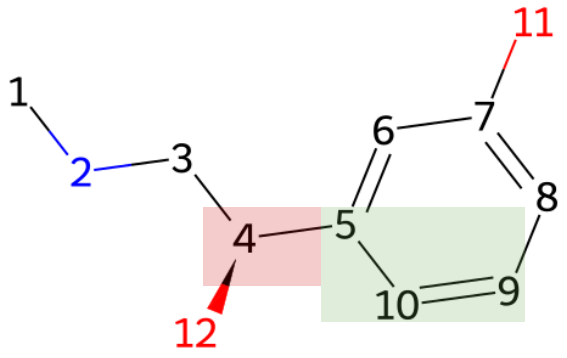
```
id10_iter1 = hash((
    1, 7468469475583712974, # ID for atom 10
    1, 9012858879331717336,  # ID for atom 5
    2, 7468469475583712974  # ID for atom 9
))
print(id10_iter1)  # 9113858623660175530
```

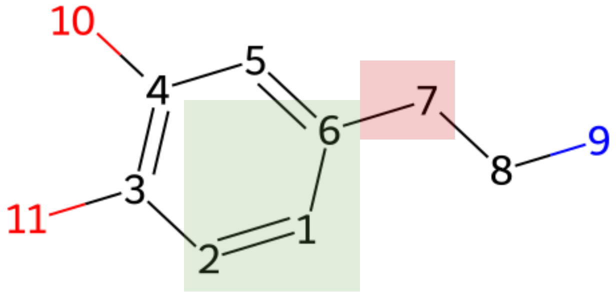
Repeat for all atoms while hashing *n - 1* IDs

Each iteration encodes local chemical information into each atom's ID

We can repeat the process for larger *n*, which captures more chemical information at a (small) computational cost

We keep track of atom IDs at each iteration to encode multiple "levels" of chemical information





```
# Iteration 0
[-96873481, -5237400, -608624, -40896092, 13106358, 39304191,
13106358, 39304191, 39304191, 39304191, 18495798, 18495798]

# Iteration 1
[-12887828, 34836456, -82428984, -76182021, 57441373, 18535308,
36698099, -16062189, -71082609, -16062189, -13803757, -35226747]

# Iteration 2
[-30242937, -22342045, -3701095, -83323106, -81401022, -79585126,
259777, -18164777, -83853893, -9624634, -63890015, -86218719]

# Iteration 3
[24482285, -67056973, -1049934, 58183281, 9686245, 65319696,
-89546467, 90525418, -96278682, -31838946, -41820336, -42202112]
```

```
# Iteration 0
[39304191, 39304191, 13106358, 13106358, 39304191, 13106358,
-608624, -608624, -2248911, 18495798, 18495798]

# Iteration 1
[-16062189, -16062189, -54942758, -54942758, 18535308, 80518135,
-46276084, 85303560, -4225841, -13803757, -13803757]

# Iteration 2
[45202524, -32527659, 91315393, -86313403, 74663225, 43056615,
-92441264, 61456743, 35268850, -86729888, -86729888]

# Iteration 3
[17051553, -83857497, -10864101, 42020134, 84228020, 88509243,
53634925, 58427327, 85169475, -62345869, -23012595]
```

Similar structural features will share atom IDs

until our iteration starts incorporating different structural features

Atom IDs are encoded into a bit array

We can get a collection of atom IDs, but how would we rapidly compare molecules with different number of atoms?

We use **bit arrays**, which are fixed-length collections of ones and zeros

```
10101100     11011010
```

This allows efficient operations

```
    10101100
AND 11011010
    --------
    10001000
```

Features that are in **both molecules**

```
    10101100
OR  11011010
    --------
    11111110
```

Features that are in **either molecules**

Converting atom IDs to bit arrays

Decide on length of bit array, for example, 1024 and fill with zeros

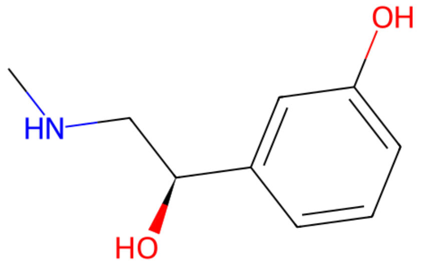
```
ecfp = [0, 0, 0, 0, ..., 0, 0, 0]
```

Divide each atom ID by the length of the array and determine the remainder

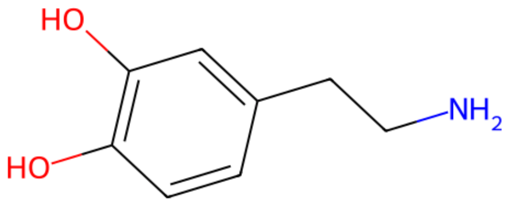
```
-1070477880882296059 % 1024 = 908
```

Set the value of the bit array at that index to 1

```
ecfp[908] = 1
```



```
10011000000000000001000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000001000001000000000000000000001000000
00000000000000000000000000000000000000000000000000001000000000001000000000000000000000000000000000000000000000000010000000000000
00000000000000000000000000100000000000000000100000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000100000000000000000000000000000000000000000000000000001000000000000100000000000000000001000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000001000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000001000000000000100100000000000000000000000000001000000001000000100000000000000000000000000
00000100000000000000001000000000000000000000000000000000000000000000000000000000000000000000000001000000000000000001001001000000
```



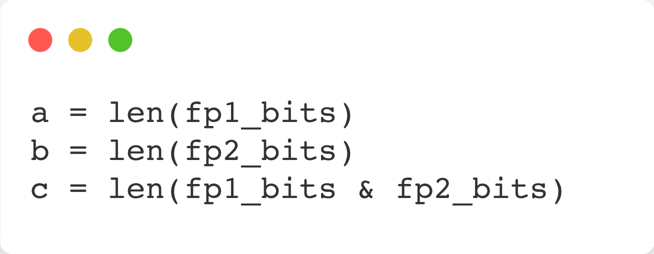
```
10011000000000000001000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000010000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000100000000000000000000000000000000000000010000000100000010000000000000000000000000010
00000000000000000000100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000001000000000000000000000000010000000000000000000000
00000000000000000000000000000000000010001000000000100101000000000000000000000000000000000100000000000000000000000000000000000000
00000100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000001000000000000000001001000000000
```

Tanimoto similarity compares the ECFPs between two molecules

Molecular similarity: The concept that similar molecules often show similar biological effects.

Using bit operations, we can compute similarity using Tanimoto

$$\text{Tanimoto similarity} = \frac{c}{a + b - c}$$



```
a = len(fp1_bits)
b = len(fp2_bits)
c = len(fp1_bits & fp2_bits)
```

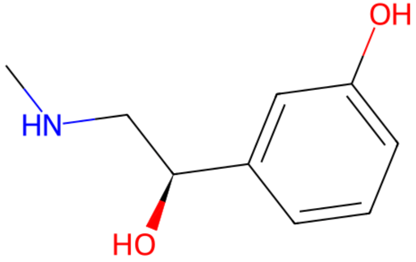
- a is the number of bits set to 1 in vector **A**.
- b is the number of bits set to 1 in vector **B**.
- c is the number of bits set to 1 in both vectors **A** and **B** (the intersection).

This formula measures the ratio of the shared features to the total number of unique features between two molecules.

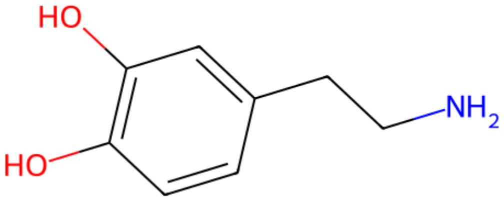
Tanimoto similarity ranges

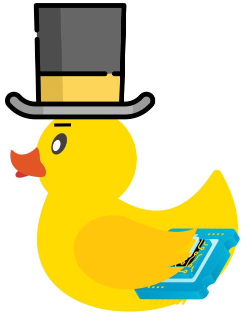
How similar does ECFPs and Tanimoto say these molecules are?

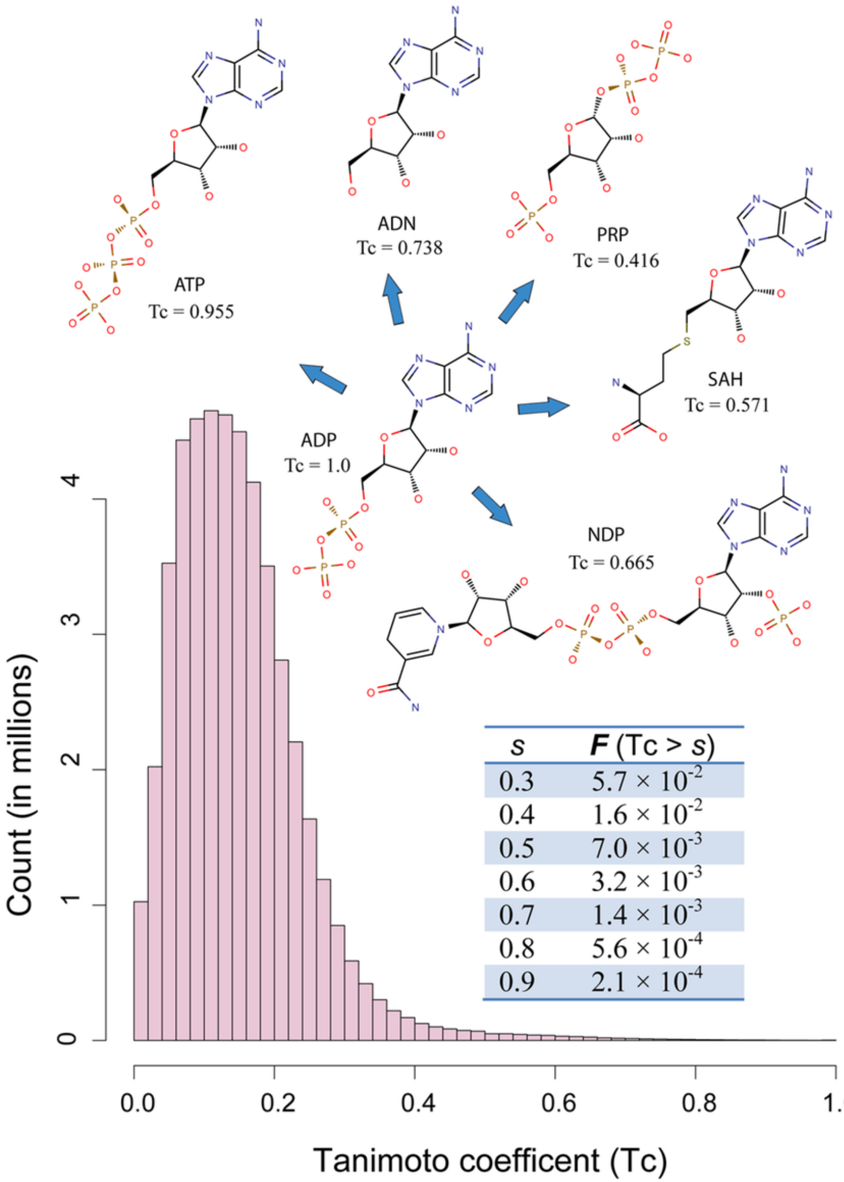
Phenylephrine



Dopamine







s	***F*** (Tc > *s*)
0.3	5.7×10^{-2}
0.4	1.6×10^{-2}
0.5	7.0×10^{-3}
0.6	3.2×10^{-3}
0.7	1.4×10^{-3}
0.8	5.6×10^{-4}
0.9	2.1×10^{-4}

Before the next class, you should

Lecture 13A:

Cheminformatics - Foundations

Today

Lecture 13B:

Cheminformatics - Methodology

Thursday

- Submit P03A
- Fill out your OMETs