ACCESS BIOINFORMATICS DATABASES WITH BIO-PYTHON

This project is aimed to deploy python-based programming pipelines and scripts to automate biological data retrieval and analysis.

5. KEGG

In this section, I deployed the KEGG module to procure genes and the pathways from the KEGG database using the enzyme commission (EC) number, which is assigned to every enzyme which has been studied or discovered so far.

Import Modules

In [57]:
```
from Bio.KEGG import REST, Enzyme
```

I created this variable named 'request' to fetch entries from the KEGG database. the 'kegg_get' function retrieves the KEGG molecule based on its EC number which is passed in as a string.

In [58]:
```
request = REST.kegg_get("ec:5.4.2.2")
open("ec_5.4.2.2.txt","w").write(request.read())
```

Out[58]:

264121

In this step, I fetched the enzyme classes present or associated with the EC number 5.4.2.2.

In [59]:
```
records = Enzyme.parse(open("ec_5.4.2.2.txt"))
record = list(records)[0]
record.classname
```

Out[59]:

```
['Isomerases;',
 'Intramolecular transferases;',
 'Phosphotransferases (phosphomutases)']
```

The pathways associated with the enzyme:

```
In [60]: record.pathway
```

Out[60]:

```
[('PATH', 'ec00010', 'Glycolysis / Gluconeogenesis'),
 ('PATH', 'ec00030', 'Pentose phosphate pathway'),
 ('PATH', 'ec00052', 'Galactose metabolism'),
 ('PATH', 'ec00230', 'Purine metabolism'),
 ('PATH', 'ec00500', 'Starch and sucrose metabolism'),
 ('PATH', 'ec00520', 'Amino sugar and nucleotide sugar metabolism'),
 ('PATH', 'ec00521', 'Streptomycin biosynthesis'),
 ('PATH', 'ec01100', 'Metabolic pathways'),
 ('PATH', 'ec01110', 'Biosynthesis of secondary metabolites'),
 ('PATH', 'ec01120', 'Microbial metabolism in diverse environments')]
```

The (first ten) genes associated with these pathways:

```
In [61]: record.genes[:10]
```

Out[61]:

```
[('HSA', ['5236', '55276']),
 ('PTR', ['456908', '461162']),
 ('PPS', ['100977295', '100993927']),
 ('GGO', ['101128874', '101131551']),
 ('PON', ['100190836', '100438793']),
 ('NLE', ['100596081', '100600656']),
 ('HMH', ['116456694', '116457795']),
 ('MCC', ['100424648', '699401']),
 ('MCF', ['101925921', '102130622']),
 ('MTHB', ['126935012', '126954887'])]
```

I fetched the genes involved by using a *for* loop, to eliminate the numbers involved:

```
In [62]: list_genes = []
         for x,y in record.genes:
    list_genes += x.split("\n")

print(list_genes[:10])
```

```
['HSA', 'PTR', 'PPS', 'GGO', 'PON', 'NLE', 'HMH', 'MCC', 'MCF', 'MTHB']
```