1.- Basic Tree operations

Relevant documentation:
 * http://etetoolkit.org/docs/latest/tutorial/index.html (main tutorial with examples)
 * http://etetoolkit.org/docs/latest/reference/index.html (reference guide)

Example Tree

"(((human:0.191029,chimp:0.997512)1.0:0.0900689,(mouse:0.752523,rat:0.91143)1.0:0.85928)0.75:0.131523,fish:0.738746,fungi:0.12271);"

1.1.- Read and Write trees

1.1.1 Create a tree with two leaves and visualize it

Relevant methods:
http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#id30

1.1.2 Read the example tree and visualize it

Relevant methods:
http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#reading-newick-trees

1.1.3 Is the tree rooted or unrooted?

Relevant information:
http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#id10

1.1.4 Read the example tree and print its newick representation without branch lengths and branch support

Relevant methods: Relevant methods:
http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#id7

1.1.5 Read the following newick containing internal node names and write it back removing branch length information

newick =

```
"((((H:0.222898,K:0.834177)D:0.231480,(F:0.516835,I:0.287718)G:0.4
97250)B:0.978909,E:0.318492)A:0.350620,
((L:0.111142,(N:0.975749,Q:0.228850)O:0.605134)J:0.672110,(P:0.763
545,S:0.245681)M:0.452808)C:0.034271);"
```

1.2 Finding nodes & Tree navigation

1.2.1 Read the example tree found at the top of the page and print a list of all leaf node names

Relevant: http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#id11

1.2.2 Read the example tree and extract the newick of the node grouping "human", "chimp" and "mouse"

Relevant methods: http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#id17

1.2.3 Read the following tree and print those partitions with a support value higher than 0.9

```
'((((Phy000EOOA_XENTR:0.328822,(((((((((Phy00085JI_HUMAN:0.003508,
(Phy000B82C_PANTR:0.021391,Phy0007XA1_HUMAN:0.024086)0.758000:0.00
7679)
0.726000:0.003461,Phy00085K5_HUMAN:0.066499)0.756000:0.005442,Phy0
008AT2_HUMAN:0.000001)0.986000:0.042419,Phy0003SBL_CANFA:0.079562)
0.399279:0.003292,Phy00026X9_BOVIN:0.019987)0.686000:0.015627,(Phy
000CU0N_RAT:0.018916,Phy000A05R_MOUSE:0.005641)0.983000:0.040844)
0.890000:0.026331,Phy00097LS_MONDO:0.099539)0.951000:0.051392,Phy0
0076B6_CHICK:0.114225)0.908000:0.042524,((Phy0006S1Q_TAKRU:0.03388
7,
Phy000DEYQ_TETNG:0.045116)0.992000:0.108875,Phy000665I_DANRE:0.113
457)0.996000:0.110827)0.129000:0.018783)0.230000:0.065105,
```

```
Phy0004FGG_CIOIN:0.798160)0.972000:0.462935,Phy00059F9_DICDI:1.428
615)0.629000:0.402703,Phy0005XY9_DROME:0.877980,Phy00005WQ_ANOGA:1
.290136);
```

Relevant attributes: http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#id9

1.3 Working with branch length distances

1.3.1 Read the example tree and detect the most distant leaf to root

Relevant methods: http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#id37

1.3.2 Read the example tree and list all leaf nodes sorted by their distance to the parent node

1.3.3 Given the following tree, What is the most distant sequence to "Phy000EOOA_XENTR"?

```
((((Phy000EOOA_XENTR:0.328822,(((((((((Phy00085JI_HUMAN:0.003508,(
Phy000B82C_PANTR:0.021391,Phy0007XA1_HUMAN:0.024086)0.758000:0.007
679)
0.726000:0.003461,Phy00085K5_HUMAN:0.066499)0.756000:0.005442,Phy0
008AT2_HUMAN:0.000001)0.986000:0.042419,Phy0003SBL_CANFA:0.079562)
0.399279:0.003292,Phy00026X9_BOVIN:0.019987)0.686000:0.015627,(Phy
000CU0N_RAT:0.018916,Phy000A05R_MOUSE:0.005641)0.983000:0.040844)
0.890000:0.026331,Phy00097LS_MONDO:0.099539)0.951000:0.051392,Phy0
0076B6_CHICK:0.114225)0.908000:0.042524,((Phy0006S1Q_TAKRU:0.03388
7,
Phy000DEYQ_TETNG:0.045116)0.992000:0.108875,Phy000665I_DANRE:0.113
457)0.996000:0.110827)0.129000:0.018783)0.230000:0.065105,
Phy0004FGG_CIOIN:0.798160)0.972000:0.462935,Phy00059F9_DICDI:1.428
615)0.629000:0.402703,Phy0005XY9_DROME:0.877980,Phy00005WQ_ANOGA:1
.290136);
```

1.3.4 Given the following newick, find the node that separates the tree into two branch-length balanced partitions (midpoint)

```
tree = '
((((Phy000EOOA_XENTR:0.328822,(((((((((Phy00085JI_HUMAN:0.003508,(Phy000B82C_P
ANTR:0.021391,Phy0007XA1_HUMAN:0.024086)0.758000:0.007679)
0.726000:0.003461,Phy00085K5_HUMAN:0.066499)0.756000:0.005442,Phy0008AT2_HU
MAN:0.000001)0.986000:0.042419,Phy0003SBL_CANFA:0.079562)
0.399279:0.003292,Phy00026X9_BOVIN:0.019987)0.686000:0.015627,(Phy000CU0N_RAT
:0.018916,Phy000A05R_MOUSE:0.005641)0.983000:0.040844)
0.890000:0.026331,Phy00097LS_MONDO:0.099539)0.951000:0.051392,Phy00076B6_CHI
CK:0.114225)0.908000:0.042524,((Phy0006S1Q_TAKRU:0.033887,
Phy000DEYQ_TETNG:0.045116)0.992000:0.108875,Phy000665I_DANRE:0.113457)0.996
000:0.110827)0.129000:0.018783)0.230000:0.065105,
Phy0004FGG_CIOIN:0.798160)0.972000:0.462935,Phy00059F9_DICDI:1.428615)0.629000
:0.402703,Phy0005XY9_DROME:0.877980,Phy00005WQ_ANOGA:1.290136);'
'
```

Relevant methods: http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#id39

1.4 Modifying trees

1.4.1 Read the example tree and root it to the "fungi" node

1.4.2 Read the example tree and change the name of leaf nodes for their corresponding codes

```
name2code = {"human": "Hsa", "fungi": "Sce", "mouse": "Mms", "chimp": "Ptr", "rat": "Rno",
"fish": "Tru"}
```

1.4.3 Given the following reference tree, prune it to get only the relationships of the species included in the example tree

```
reference_tree =
'(((((((human,chimp),(mouse,rat)),(cow,horse)),fish),bird),((fly,mosquito),worm)),fungi);'
```

Relevant methods: http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#id32

1.4.4 Read the example tree and separate the node containing mouse and rat from the rest of the tree

Relevant methods: http://etetoolkit.org/docs/latest/tutorial/tutorial_trees.html#id31

1.4.5 Read the example tree and create a multifurcation between human, chimp, mouse and rat

2 Tree Annotation

2.1 Adding features to nodes

2.1.1 Given the example tree, add a feature to every leaf node about whether it is a mammal species or no

Relevant methods: iter_leaves, add_feature

2.1.2 Using the result of the previous exercise, list the branch length of all non-mammal species

Relevant methods: "search_nodes" or "iter_leaves"

2.1.3 Given the following gene tree, prune it to maintain only the relationships of mammal species. Extract the newick of the new tree.

```
tree =
"(((((human1,chimp1),rat1),fish1),(((mouse2,rat2),human2),fungi2))
,(((human3,chimp3),(mouse3,rat3)),fish3));"
```

Relevant methods: "prune", "add_feature", "iter_leaves"

3. Phylogenetic trees

Example tree:

```
((Dmelanogaster_001,Agambiae_001),((Hsapiens_001,(Hsapiens_002,Hsa
piens_003)),(Rnorvegicus_001,(Mmusculus_001,Rnorvegicus_002))));
```

3.1 Working with phylogenetic trees

3.1.1 Read a phylogenetic tree

Load the example tree as a phylogenetic tree

Relevant methods: PhyloTree

3.1.2 Obtain the species names of the different leaves

Read the example tree and have a look at how the species names have been uploaded: "node.species" and compare it to the name of the leaf.

Relevant attribute: node.species

3.1.3 Change the species naming function

Load the tree again ensuring that the species name that is taken is the string before the "_"

Make sure that the species name is correct for each leaf

Relevant argument: PhyloTree, "sp_naming_function"

3.1.4 Check monophyly

In common cladistic usage, a monophyletic group is a taxon (group of organisms) which forms a clade, meaning that it contains all the descendants of the possibly hypothetical closest common ancestor of the members of the group.

Given the following tree, are the human (Hsa) and chimp (Ptr) sequences always monophyletic?

newick = "(((((Hsa1, Ptr1), Mmu1), ((Hsa1, Mms1), Ptr1)), (Mms2, Rno2)), Cin1);"

Relevant methods: node.is_monophyletic

3.2 Species aware rooting

3.2.1 Given the following species tree, create a dictionary of ages of all species relative to human (Hsa)

Although it can be done automatically, you can define such a dictionary manually by inspecting the tree topology.

tree = "(((Hsa, Ptr), (Rno, Mms)), (Dme, Aga));"

```
#                               /-Hsa
#                     /--------|
#                     |         \-Ptr
#           /---------|
#           |         |         /-Rno
#           |         \--------|
#----------|                     \-Mms
#           |
#           |         /-Dme
#           \---------|
#                     \-Aga
#
'
```

3.2.2 Root the following tree to the oldest leaf relative to human (Hsa) using the previous dictionary of relative ages.

```
newick = "
((Dme_001:2.1,Aga_001:0.4),((Hsa_001:0.9,(Hsa_002:0.7,Hsa_003:0.5)
),(Rno_001:0.5,(Mms_001:1.2,Rno_002:1.1))));"
```

Relevant methods: get_farthest_oldest_leaf, set_outgroup

3.2.3 Calculate the relative age of the different nodes of the previous tree using our example dictionary of relative ages

species2age = {"Hsa": 1,"Mmu": 2,"Ptr": 2,"Bta": 3,"Mdo": 3,"Mms": 3,"Cfa": 3,"Rno": 3,"Gga": 4,"Xtr": 5,"Dre": 6,"Fru": 6,"Cin": 7,"Aga": 8,"Ame": 8,"Cbr": 8,"Cel": 8,"Dme": 8}

Relevant methods: get_age

3.3 Orthology and Paralogy predictions

Tree file: data_course_ete/cyt_unrooted.txt

species2age = {"Hsa": 1,"Mmu": 2,"Ptr": 2,"Bta": 3,"Mdo": 3,"Mms": 3,"Cfa": 3,"Rno": 3,"Gga": 4,"Xtr": 5,"Dre": 6,"Fru": 6,"Cin": 7,"Aga": 8,"Ame": 8,"Cbr": 8,"Cel": 8,"Dme": 8}

3.3.1 Obtain the evolutionary events pertaining to a leaf

Root the tree at Dme0013542. How many duplications occurred in the lineage of Hsa0011971?
Relevant methods: get_my_evol_events

3.3.2 Obtain all the evolutionary events of the previous tree

With a rooted tree, obtain all the evolutionary events: get_descendant_evol_events

3.3.3 Search for orthologs

Scan the list of evolutionary events and obtain the speciation events

3.3.4 Search for paralogs

Repeat the exercise but look for paralogs, this time also print the names of the sequences involved in the event

Relevant attributes: event.node

3.3.5 Visualize the tree

Have a look at the tree now that the prediction has been made

3.4 Tree reconciliation

3.4.1 Reconcile the tree

Reconcile a tree with the species tree.

tree = "(Ago1,Kla1,((Kwa1,Skl1),(Kpo1,(Sca1,(Cgl1,(Sba1,(Sku1,((Sce1,Smi1),Smi2)))))))));"

species tree = "(((((((Smi,(Sce,Spa)),Sku),Sba),Sca),Cgl),Kpo),((Skl,Kwa),(Kla,Ago)));"

species2age =
{"Sce":1,"Spa":2,"Smi":3,"Sku":4,"Sba":5,"Sca":6,"Cgl":7,"Kpo":8,"Kla":9,"Ago":9,"Kwa":9,"Skl":9}
'

3.4.2 Visualize the reconciled tree

Visualize the reconciled tree obtained above.

3.5 More advanced exercises

3.5.1 compare orthology predictions

Compare the orthology predictions obtained when using the species overlap algorithm and the reconciliation method.

Tree: course_data_ete/citoc_aphid.nw

Species tree: course_data_ete/aphid_tree.nw

3.5.2 Detect one to one orthology relationships between human (xxxx_HUMAN) and mouse (xxxx_MOUSE) sequences

```
Phy00085K5_HUMAN_newick = '
((((Phy000EOOA_XENTR:0.328822,(((((((((Phy00085JI_HUMAN:0.003508,(
Phy000B82C_PANTR:0.021391,Phy0007XA1_HUMAN:0.024086)0.758000:0.007
679)
0.726000:0.003461,Phy00085K5_HUMAN:0.066499)0.756000:0.005442,Phy0
008AT2_HUMAN:0.000001)0.986000:0.042419,Phy0003SBL_CANFA:0.079562)
0.399279:0.003292,Phy00026X9_BOVIN:0.019987)0.686000:0.015627,(Phy
000CU0N_RAT:0.018916,Phy000A05R_MOUSE:0.005641)0.983000:0.040844)
0.890000:0.026331,Phy00097LS_MONDO:0.099539)0.951000:0.051392,Phy0
0076B6_CHICK:0.114225)0.908000:0.042524,((Phy0006S1Q_TAKRU:0.03388
7,
Phy000DEYQ_TETNG:0.045116)0.992000:0.108875,Phy000665I_DANRE:0.113
457)0.996000:0.110827)0.129000:0.018783)0.230000:0.065105,
Phy0004FGG_CIOIN:0.798160)0.972000:0.462935,Phy00059F9_DICDI:1.428
615)0.629000:0.402703,Phy0005XY9_DROME:0.877980,Phy00005WQ_ANOGA:1
.290136);'
```

* load the tree changing the species_name_function to automatically detect the species code of each leaf
* root the tree to Phy0005XY9_DROME
* detect evolutionary events
* scan the events to decide which represents one to one orthology relationships

4.- Tree visualization

4.1. Load the following tree and show it including branch length, branch support and leaf name information

```
tree = '
((((Phy000EOOA_XENTR:0.328822,(((((((((Phy00085JI_HUMAN:0.003508,(
Phy000B82C_PANTR:0.021391,Phy0007XA1_HUMAN:0.024086)0.758000:0.007
679)
0.726000:0.003461,Phy00085K5_HUMAN:0.066499)0.756000:0.005442,Phy0
008AT2_HUMAN:0.000001)0.986000:0.042419,Phy0003SBL_CANFA:0.079562)
0.399279:0.003292,Phy00026X9_BOVIN:0.019987)0.686000:0.015627,(Phy
000CU0N_RAT:0.018916,Phy000A05R_MOUSE:0.005641)0.983000:0.040844)
0.890000:0.026331,Phy00097LS_MONDO:0.099539)0.951000:0.051392,Phy0
0076B6_CHICK:0.114225)0.908000:0.042524,((Phy0006S1Q_TAKRU:0.03388
7,
Phy000DEYQ_TETNG:0.045116)0.992000:0.108875,Phy000665I_DANRE:0.113
457)0.996000:0.110827)0.129000:0.018783)0.230000:0.065105,
Phy0004FGG_CIOIN:0.798160)0.972000:0.462935,Phy00059F9_DICDI:1.428
615)0.629000:0.402703,Phy0005XY9_DROME:0.877980,Phy00005WQ_ANOGA:1
.290136);'
```

Relevant method: TreeStyle (show_leaf_name, show_branch_length, show_branch_support), t.show

4.2. Plot the same tree but write the result as a png image

Relevant method: t.render

(image format is auto detected from file name extension)
(you can also export the image as an SVG file, so you can edit it with any vector graphics program such as Inkscape or Adobe Illustrator)

4.3. Visualize the same tree in a circular way and rotate it 90 degrees

Relevant method: TreeStyle.mode, TreeStyle.rotation

Try to plot the same tree using only half of the circumference (TreeStyle.arc_spam)

4.4. Using the same tree, change the size and color of all HUMAN sequences

You will need to create a custom layout function and pass it to the show method. Example:

```
def mylayout(node):
        if node.is_leaf():
                node.img_style["size"] = 15
```

Relevant methods: node.img_style, layout function

4.5. Add a text-face to all leaf names showing their branch length in color and with a large font size on the bottom part of the branches

Relevant methods: faces module -> add_faces_to_node, AttrFace

example:

```
from ete3 import Tree, faces

t = Tree()
t.populate(10)

def layout(node):
  F = faces.AttrFace("dist")
  faces.add_face_to_node(F, node, column=0, position = "branch-bottom")

t.show(layout)
```