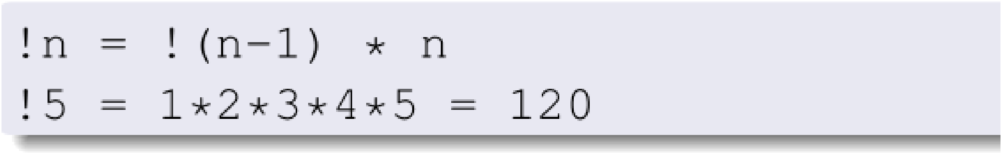
Maya/PyMel Motion Capture

Overview

- Recursive functions in Python
- PyNodes and the DAG
- DAG traversal using recursive functions
- Some additional PyMEL for the assignment

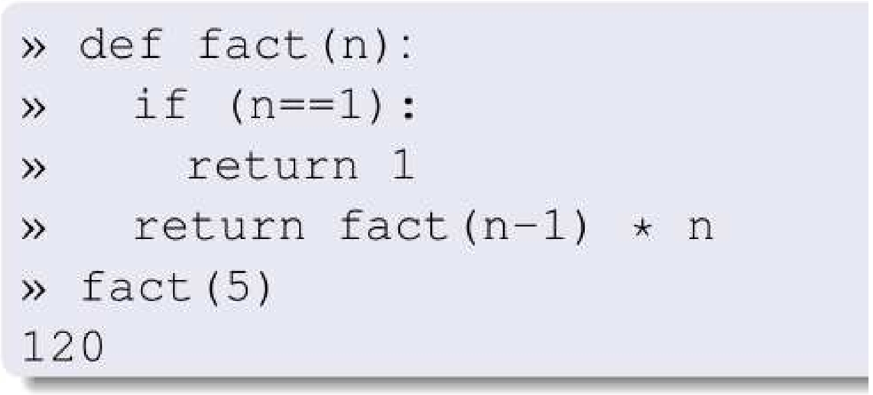
Recursive functions

- What is a recursive function ?
  - In math…



```
!n = !(n-1) * n
!5 = 1*2*3*4*5 = 120
```

  - In code:

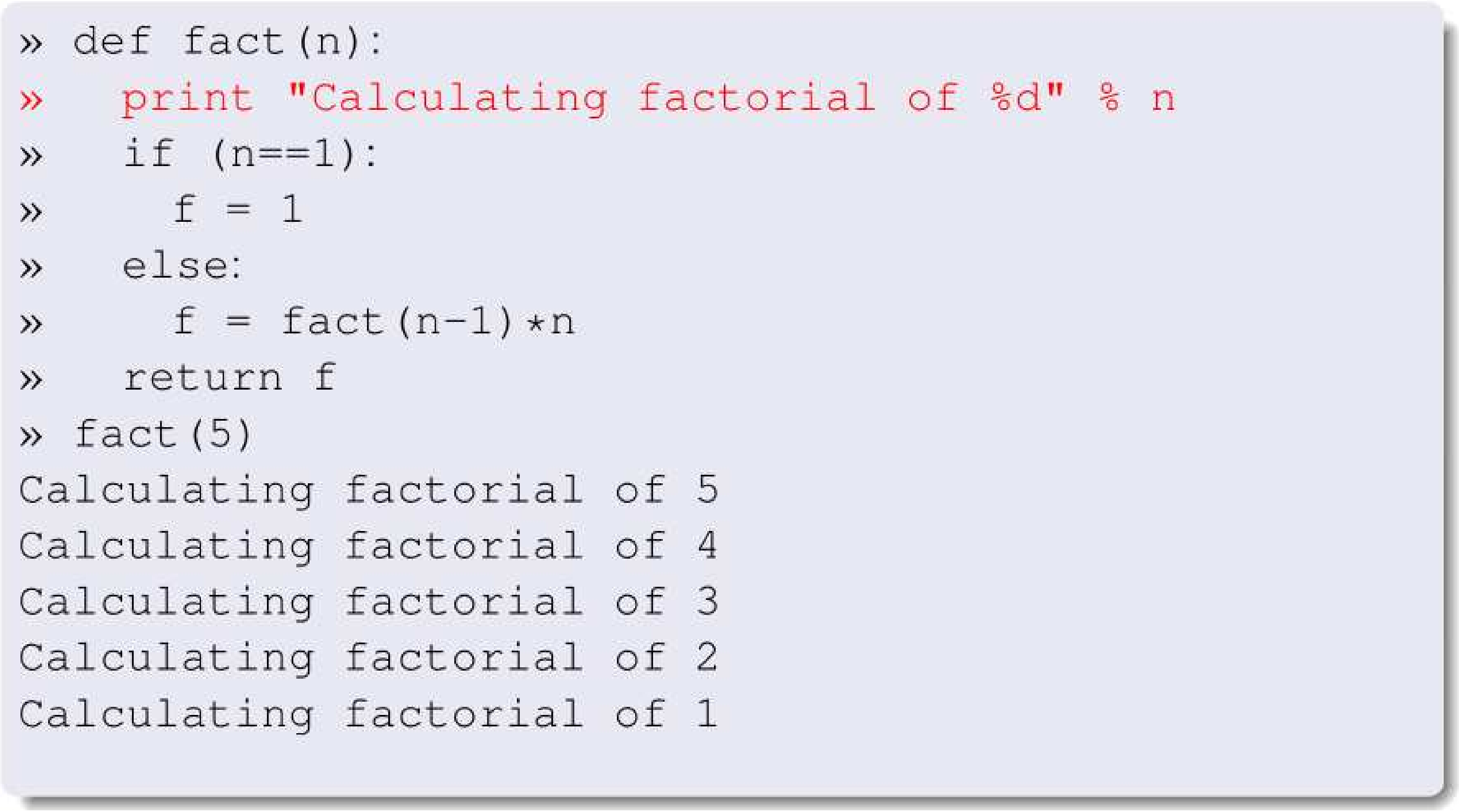


```
» def fact(n):
»   if (n==1):
»     return 1
»   return fact(n-1) * n
» fact(5)
120
```

- Example in WingIDE

Order of execution

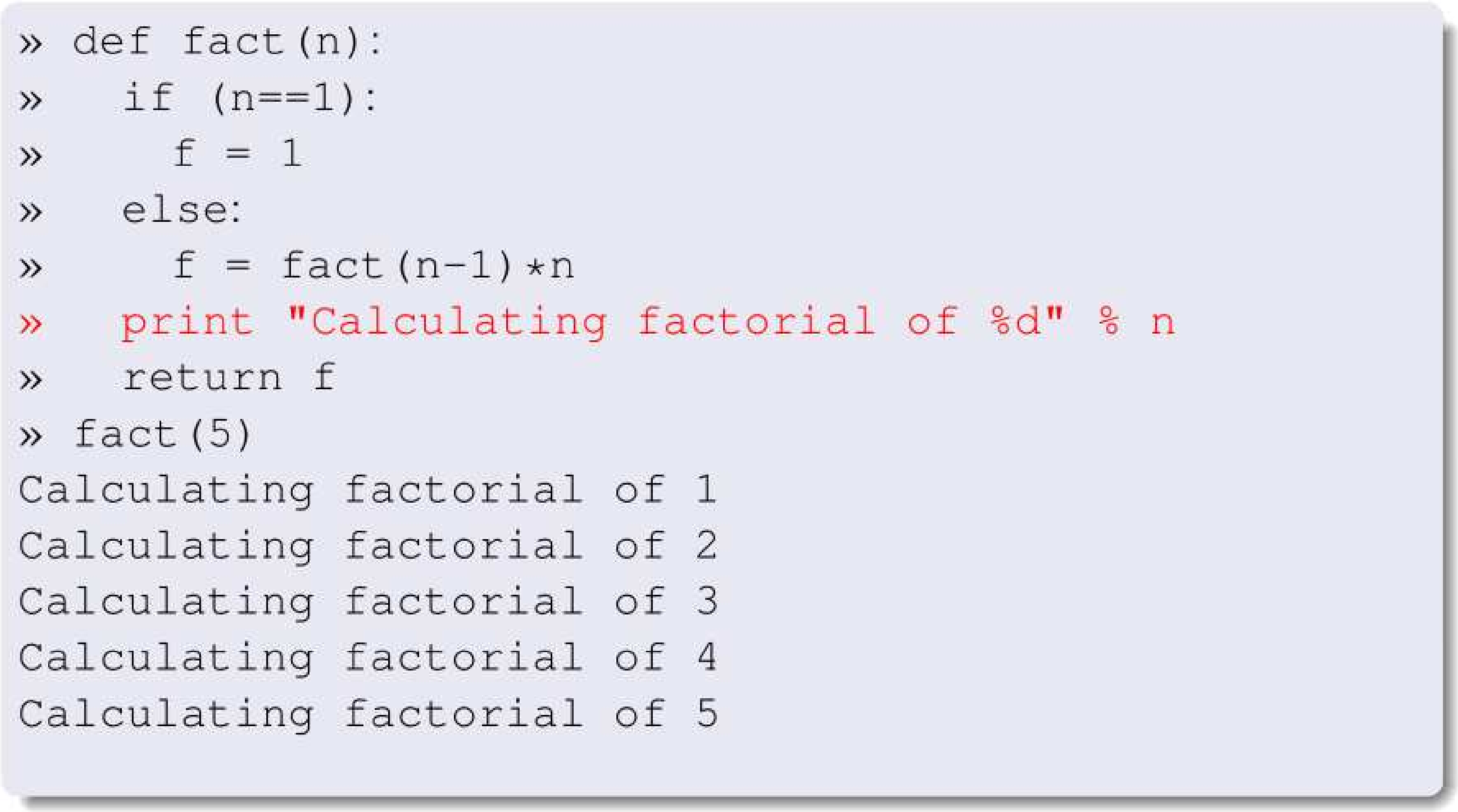
- Statements before the recursive call



```
» def fact(n):
»   print "Calculating factorial of %d" % n
»   if (n==1):
»     f = 1
»   else:
»     f = fact(n-1)*n
»   return f
» fact(5)
Calculating factorial of 5
Calculating factorial of 4
Calculating factorial of 3
Calculating factorial of 2
Calculating factorial of 1
```

Order of execution (2)

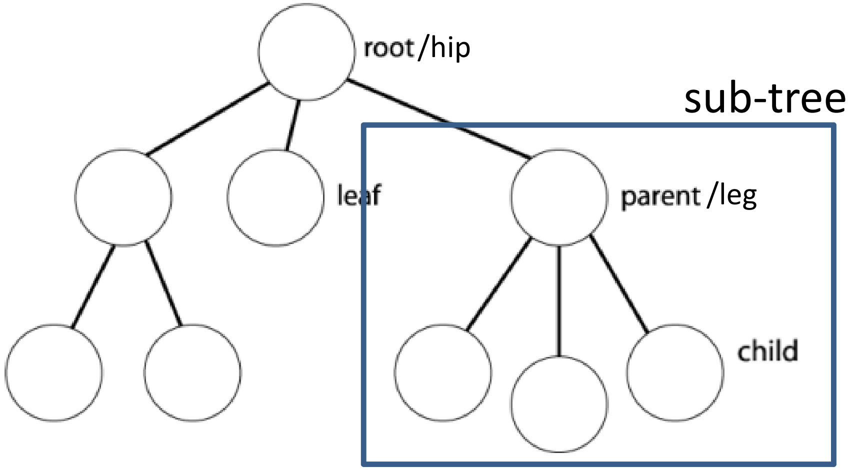
- Statements after the recursive call



```
» def fact(n):
»   if (n==1):
»     f = 1
»   else:
»     f = fact(n-1)*n
»   print "Calculating factorial of %d" % n
»   return f
» fact(5)
Calculating factorial of 1
Calculating factorial of 2
Calculating factorial of 3
Calculating factorial of 4
Calculating factorial of 5
```

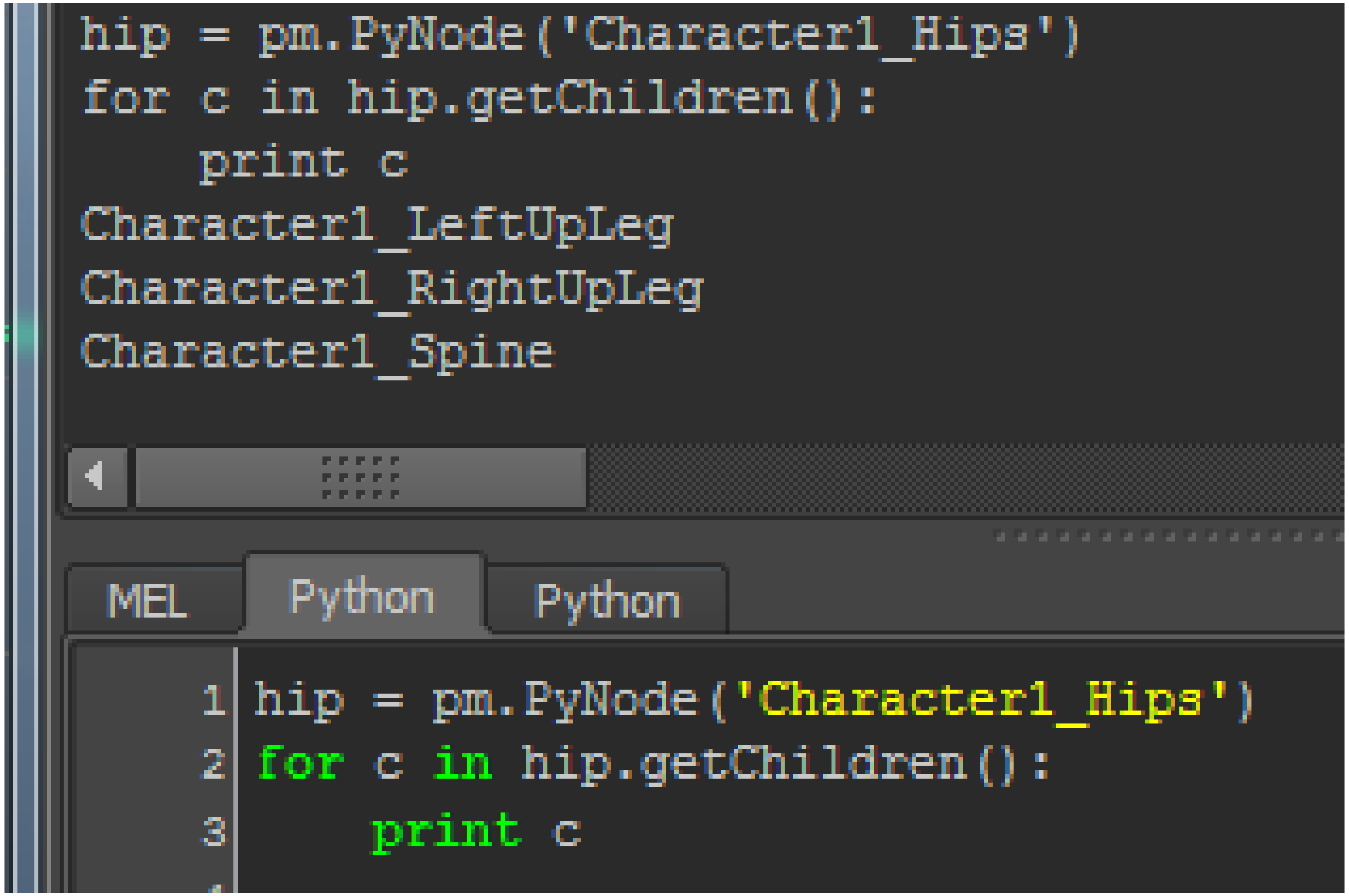
PyNodes and the DAG

- The DAG of a Skeleton
  - Tree like structure
    - Root (usually the hip)
    - 3 Children (legs, spine)
- It can be seen as a ***recursive*** structure
  - The hip is the root and has 3 children
  - Consider the sub-tree, starting from one leg
    - Now, the leg is the root, and has three children (weird leg)



PyNodes and the DAG (2)

- PyNode references to nodes in the DAG

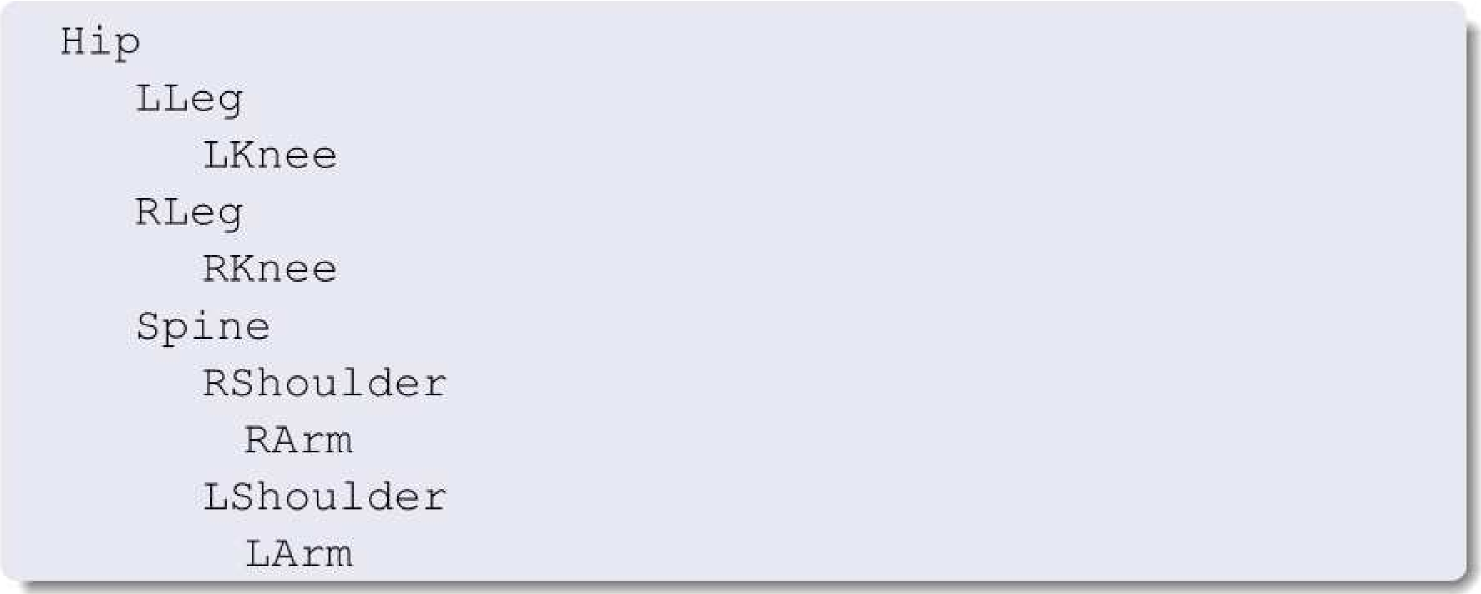


```
hip = pm.PyNode('Character1_Hips')
for c in hip.getChildren():
    print c
Character1_LeftUpLeg
Character1_RightUpLeg
Character1_Spine
```

```
hip = pm.PyNode('Character1_Hips')
for c in hip.getChildren():
    print c
```

PyNodes and the DAG (3)

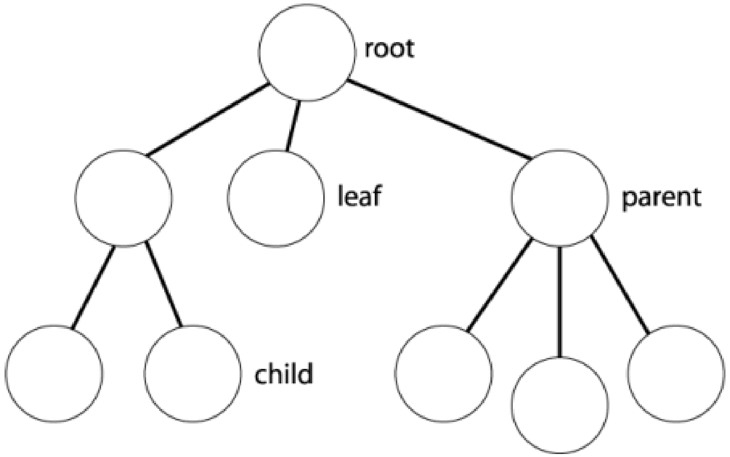
- Treat each node in the DAG as a root of a sub-tree
- The biggest sub-tree is the skeleton, from the hip
- Inside, there are other sub-trees



```
Hip
   LLeg
     LKnee
   RLeg
     RKnee
   Spine
     RShoulder
       RArm
     LShoulder
       LArm
```

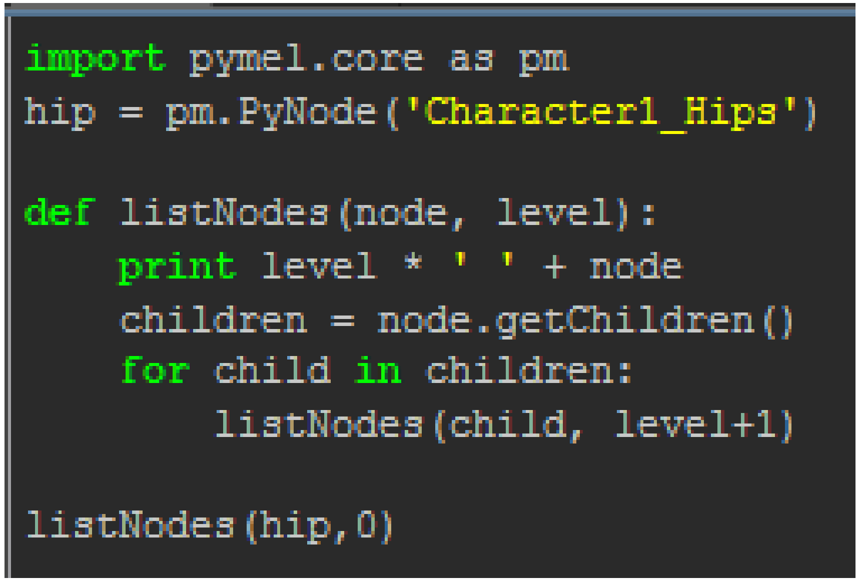
Tree traversal with a recursive function

- Recursive functions are great for tree traversal
- Write a general case, for a root and its children
- Treat **each node as a root of a sub-tree**



Tree traversal with a recursive function (2)

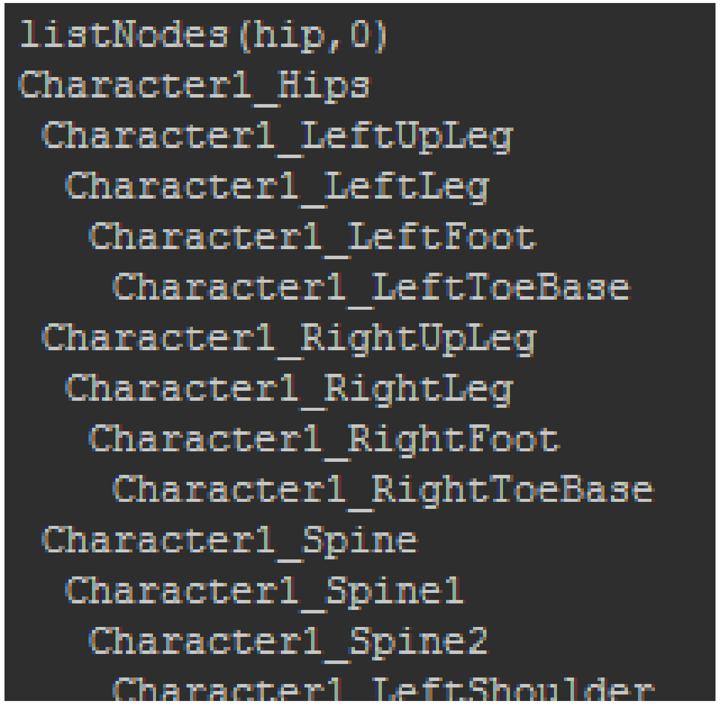
- Create a Maya Skeleton and try this code



```python
import pymel.core as pm
hip = pm.PyNode('Character1_Hips')

def listNodes(node, level):
    print level * ' ' + node
    children = node.getChildren()
    for child in children:
        listNodes(child, level+1)

listNodes(hip,0)
```



```
listNodes(hip,0)
Character1_Hips
 Character1_LeftUpLeg
  Character1_LeftLeg
   Character1_LeftFoot
    Character1_LeftToeBase
 Character1_RightUpLeg
  Character1_RightLeg
   Character1_RightFoot
    Character1_RightToeBase
 Character1_Spine
  Character1_Spine1
   Character1_Spine2
    Character1_LeftShoulder
```

Additional PyMEL code - Transforms

- Grab a reference to a node/joint

  ```
  node = pm.PyNode(‘node123’)
  ```

- Select a node in Maya from code

  ```
  pm.select(‘node123’)
  pm.select(node)
  ```

- Create a joint

  ```
  pm.joint(name=‘new_joint’) // parent if selected
  ```

- Transforms and inverse

  ```
  Tx = node.getTransformation()
  inverse = Tx.asMatrixInverse()
  ```

- Translation (pymel.core.datatypes as dt)

  ```
  Trans = node.getTranslation()
  node.setTranslation(dt.Vector(1,1,1))
  ```

- Rotations, get() and set()

  ```
  node.rotate.set(otherNode.rotate.get())
  ```

- Vector * Matrix
  - Be careful, matrices are usually 4x4, and vectors can be 3 or 4 tuples
  - Use VectorN for vector4, with a 1 for w coord if it is a point.

Additional PyMEL code - Animation

- Setting keys from code

```
# set timeline
pm.setCurrentTime(100)
# create key for attribute at current time
node.translate.setKey()
node.rotate.setKey()
```

Additional PyMEL code - Files

- Opening a File Dialog

```
# get path to file
path = pm.fileDialog()
# import any sort of file (plugin enabled!)
pm.importFile(path)
```

- More PyMel docs:
  - PyMel DOCS

Questions ?