



Bioinformatical problem solving with Python



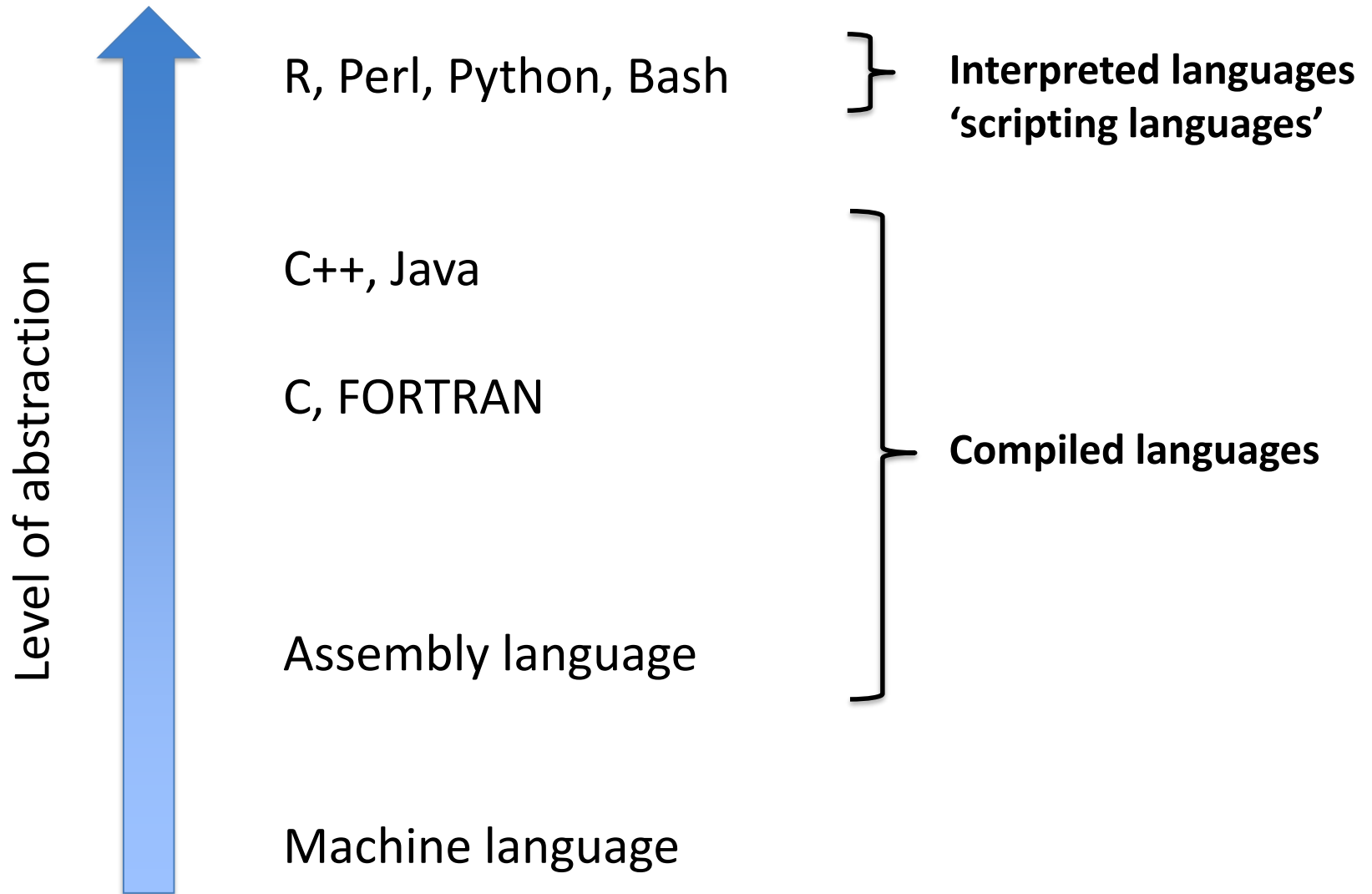
Wednesdays 17:30-19:00, M801
alexander.nater@uni-konstanz.de

- Get a general understanding of how programming languages work
- Learn to use Python as a general-purpose language
- Interactively discuss problems and potential solutions
- Ideally, apply skills to problems related to your own projects
- Alternatively, find problems on Rosalind:
<http://rosalind.info>

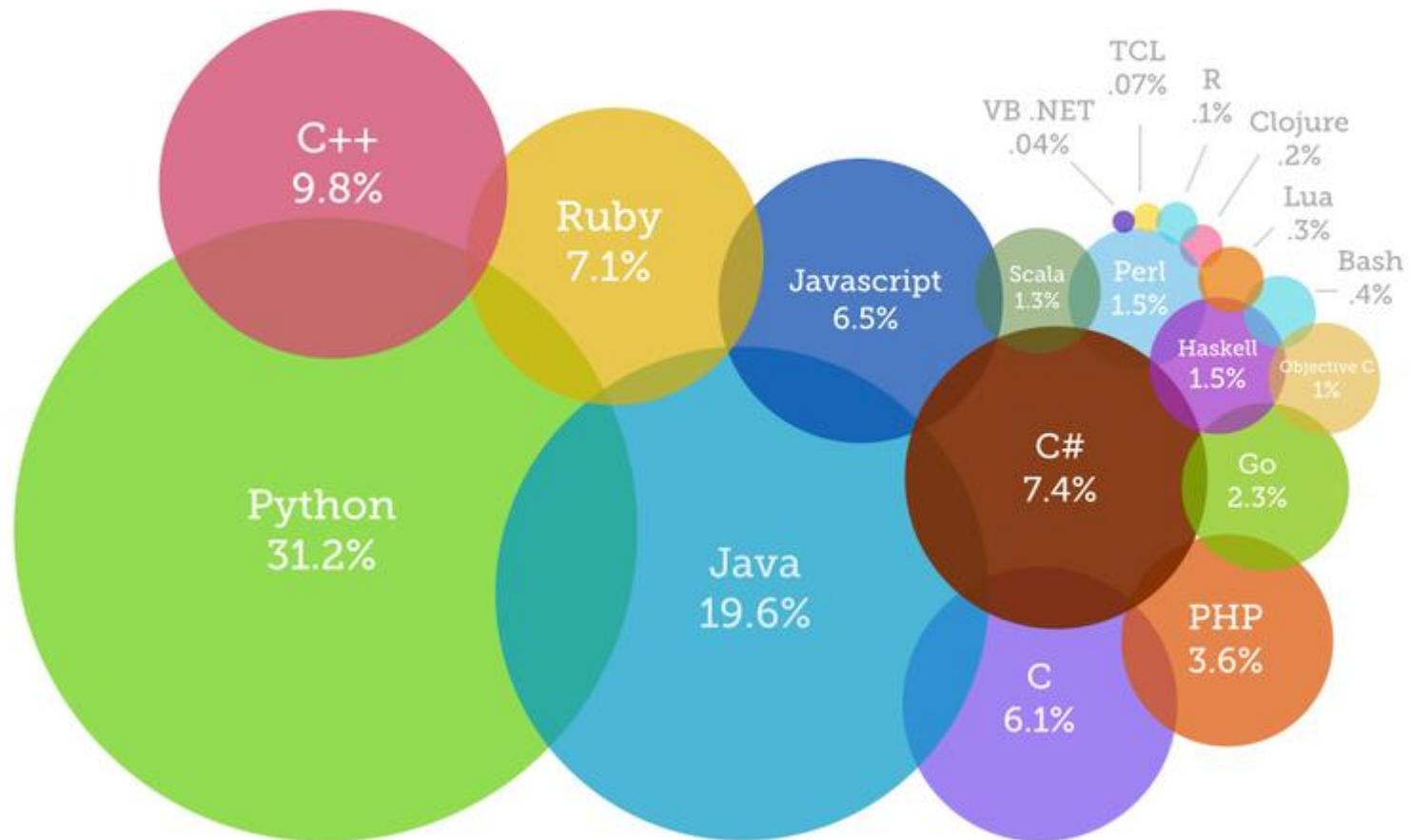


```
def add5(x):
    return x+5

def dotwrite(ast):
    nodename = getNodeName()
    label=symbol.sym_name.get(int(ast[0]),ast[0])
    print '    %s [label="%s' % (nodename, label),
    if isinstance(ast[1], str):
        if ast[1].strip():
            print '= %s";' % ast[1]
        else:
            print '"]'
    else:
        print '"]';'
        children = []
        for n, child in enumerate(ast[1:]):
            children.append(dotwrite(child))
        print ',    %s -> {' % nodename
        for i, name in enumerate(children):
            print '%s' % name,
```

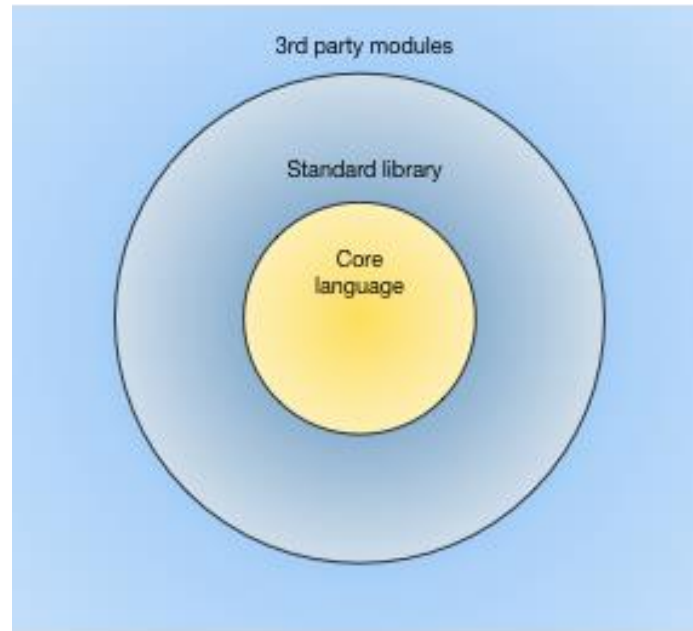


Most Popular Coding Languages of 2015



- Easy to learn, clear syntax
- Very flexible, from simple scripts to complex programs
- Very popular in scientific computing
- Jack of all trades, master of none
- Broad range of external packages
- Good plotting libraries

- Two lines of Python: Python 2.7.x and Python 3.6.x
- Differences in the syntax
- No backward compatibility
- We will learn Python 3, but not all external packages adapted yet.



- A text editor (e.g., gedit, emacs, vim)
- The Python interpreter
- A terminal application to run the interpreter in.

- Already preinstalled on most Linux-based OS.
- We will use Anaconda for easy package management:
<https://anaconda.org>
- We will use Jupyter Notebook as digital lab book:
<http://jupyter.org> (included in Anaconda)

- Official Python documentation:
<http://docs.python.org>
- The Python tutorial:
<http://docs.python.org/tutorial>