# Python Programming Language

## Introduction

Sachin

PVPPCOE

December 21, 2015

# Introduction

- Scripting language
- Easy to learn/Powerful
- Efficient data structures
- Object Oriented
- Dynamic Typing
- Extended support for C(CPython), Java(Jython) & (.NET)IronPython
- Runs on most platforms
- Latest stable version `3.4.3`

# Why Python?

- Readable code
- Faster development cycle
- Powerful standard libraries
- Powerful interpreter
- `https://www.python.org/about/success/`

# Features

- Easy

```
1 first_name = 'Sachin'
2 print("Hello %s!" % first_name)
3 print("Hello " + first_name + "!")
4 # Hello Sachin!
5 print("Hello",first_name)
6 # Hello Sachin
```

# Features

- Inbuilt libraries

```
1 for i in range(3):
2     print(i)
3 # 0
4 # 1
5 # 2
```

# Features

- Inbuilt libraries

```
1 import crypt
2 crypt.crypt('mysupersecretpassword', 'py')
3 # 'pyjwbR4BKuWLo'
```

# Features:

Powerful interpreters

- python
- ipython
- bpython(*we will use bpython*)

# Features

- *Indentation specifies scope*

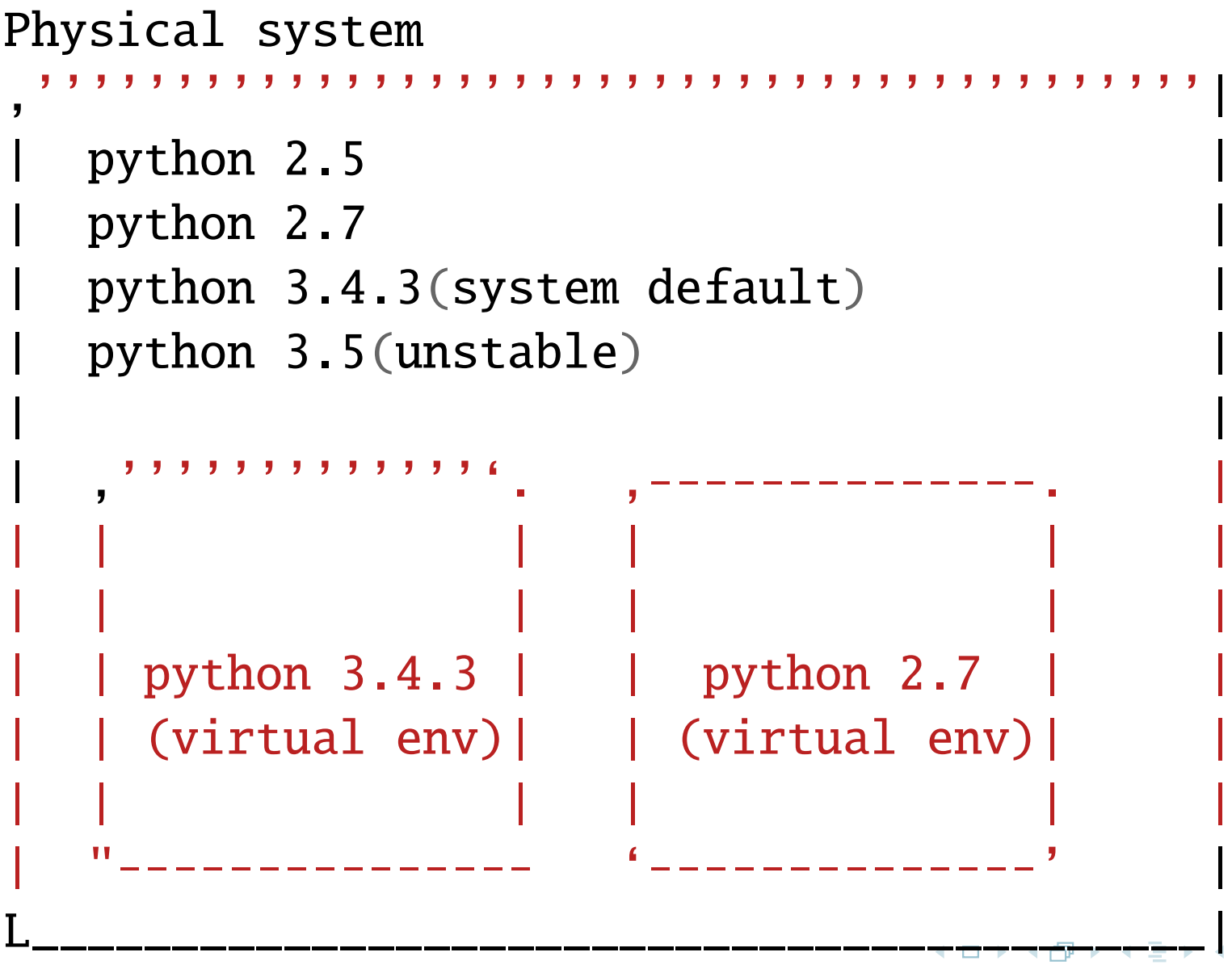
```
1    if cars:
2        for car in cars:
3            print(car.speed)
```

# Features

Package utils

- `pip`
- `easy_install`

# Python virtual environment



# Python virtual env

## Installation

```
sudo apt-get install python-virtualenv
sudo apt-get install python3-dev
sudo apt-get install python-setuptools
```

## Create virtual env

```
mkdir ~/virtualenvs
which python3.4
# /usr/bin/python3.4
virtualenv -p /usr/bin/python3.4 \
   ~/virtualenvs/workshop
```

# Virtual env

## Activate

```
source ~/virtualenvs/workshop/bin/activate
```

## Deactivate

```
deactivate
```

# Install packages

bpython

```
pip install bpython
```

# List install packages

```
pip list
# alabaster (0.7.6)
# amqp (1.4.6)
# anyjson (0.3.3)
# argh (0.26.1)
# astroid (1.3.6)
# Babel (1.3)
# billiard (3.3.0.20)
# blessings (1.6)
# bpython (0.14.2)
# celery (3.1.18)
# ..
```

# bpython

```
$ bpython
# bpython version 0.14.2 on top of Python 3.4.3
# /home/sachin/virtualenvs/workshop/bin/python3.4
>>>
```

# Examples: String manipulation

```
1  first_name = 'Sachin'
2  first_name.upper()  # SACHIN
3  first_name.capitalize()  # Sachin
4  first_name.count('a')  # 1
5  first_name.count('d')  # 0
6  first_name.endswith('n')  # True
7  first_name.startswith('s')  # False
8  first_name.startswith('s'.upper())  # True
9  ''.join(first_name)  # Sachin
10 ' '.join(first_name)  # S a c h i n
11 '.'.join(first_name)  # S.a.c.h.i.n
```

# Examples: Arithmetic

```
1 first_name*3   # SachinSachinSachin
2 first_name+3   # TypeError
```

# How to find help?

```
1 first_name.__doc__  # default docstring
2 first_name.__dir__()  # methods
```

# Examples

```
1 first_name.partition.__doc__
2 first_name.partition('ch')  # ('Sa', 'ch', 'in')
```

# Exercise 1

- Can you convert `Sachin` to `sACHIN` ?
  *Hint: I am inverting the case*

# Exercise 2

- Try to find out how this happened?

```
first_name.center(first_name.__len__()+2, '*')
```

*Output was:* `*Sachin*`

# Exercise 3

- We saw how we multiply a string using

```
first_name*3   # SachinSachinSachin
```

Can you print a string in following pattern?

```
Sachin Sachin Sachin
```

*Hint: Try to use inbuilt methods available to* `first_name`*.*
*You may want to explore .ljust() & .strip()*

# Solution

1. `first_name.swapcase()`
2. `.center(width, [string])` take two argument a *width* and a *string*. *width* is mandatory, if *string* is not specified, default is assumed as *space*. I want to ensure that my name should be in center with one space on both sides. Total width I needed was 8(6 + 2). I used a builtin method .*_len_()* to calculate width and added 2 to the result. Finally I specified *string* as '*'
3. `(first_name.ljust(7)*3).strip()`

# End

## Email

iclcoolster@gmail.com

## Blog

http://psachin.github.io