Python

Introduction

Background and Expectations

Previous experience in programming?

In Python?

Any particular expectations?

Future career plans?

History and successes

Guido Von Rossum in 1989

V1: 1994

Today: 2.7 and 3.6

Google

YouTube

Dropbox

Nasa

NYSE

.. you!

Zen of python

```
>import this
```

Beautiful is better than ugly.

Readability counts.

Explicit is better than implicit.

...

Characteristics

Newlines and tabs count

Constructions borrowed from functional languages

But object oriented programming language

Duck typing

no semicolons “;” no brackets “{...}”

C Versus Python

```c
int factorielle(int n) {
     if (n < 2) {
         return 1;
     } else {
         return n * factorielle(n - 1);
     }
 }
```

```c
int array[10];
for(int i=0;i<10;i++){
    array(i) = 2*i;
}
```

Python

```python
def factorielle(n):
    if n < 2:
        return 1
    else:
        return n * factorielle(n - 1)
```

```python
array = [2*i for i in range(1,10)]
```

Elementary types

None

Boolean

int and float

string

list and tuple

set and frozenset

dictionary

None

True/False

2 and 2.2

“abcd”

[3,5,3,9] and (3,5,3,9)immutable

{3,5,9} mutable and immutable

{“sergey”:3.5,“anna”:5.75,}

Operators

+, -, *, /, //, %, **

==, !=, <, <=, >, >=

is, is not

and, or, not

```
107 % 10 # 7

107 // 10 # 10

2 ** 3 # 8

a < x < b possible !
```

Lists

L[i] : k=i (from 0)

L[i :j]   i=<l<j-1

L[i :j :k] : step of k

```
L = [3, 5, 7, 9]

L[2] # 7

L[1 :3] # [5, 7]

L[0 :3 :2] # [3, 7]

L[-1] # 9

L[1 :] # [5, 7, 9]

L[ :2] # [3, 5]

L[ :] # [3, 5, 7, 9]
```

Lists 2

`x in L`

`L1 + L2 : concatenate`

`L * n,`

`len(L), min(L), max(L)`

```
L = [3, 5, 7, 9]

3 in L # True

[3,5] + [7,9] # [3, 5, 7, 9]

2 * [1, 2] # [1, 2, 1, 2]

4, 3, 9
```

Dictionary

```
{"key1" : val1, "key2" : val2}

dict(key1=val1, key2=val2)

dict([('key1', val1), ['key2', val2]])
```

```
counts = {'turtle' : 4, 'rabbit' : 3}

counts[turtle] += 1

print "turtle :", counts['turtle']
```

Conditions

```
if condition1 :
    block 1
elif condition2 :
    block 2
else :
    block 3
```

```
if a == b :
    a += 1 # in the block
    print "a > b !"
print "out of the block if"
```

Loops

```
while condition :
    block
```

```
for variable in iterable :
    block
```

```
while a < x < b:
  x += random.rand()
```

```
for i in range(1, 10):
  array[i] = 5 + math.sqrt(i)
```

Functions

```
def hello(name):
    print ‘hello %s’ % name


def mean(numbers):
    return sum(numbers) / len(numbers)
```

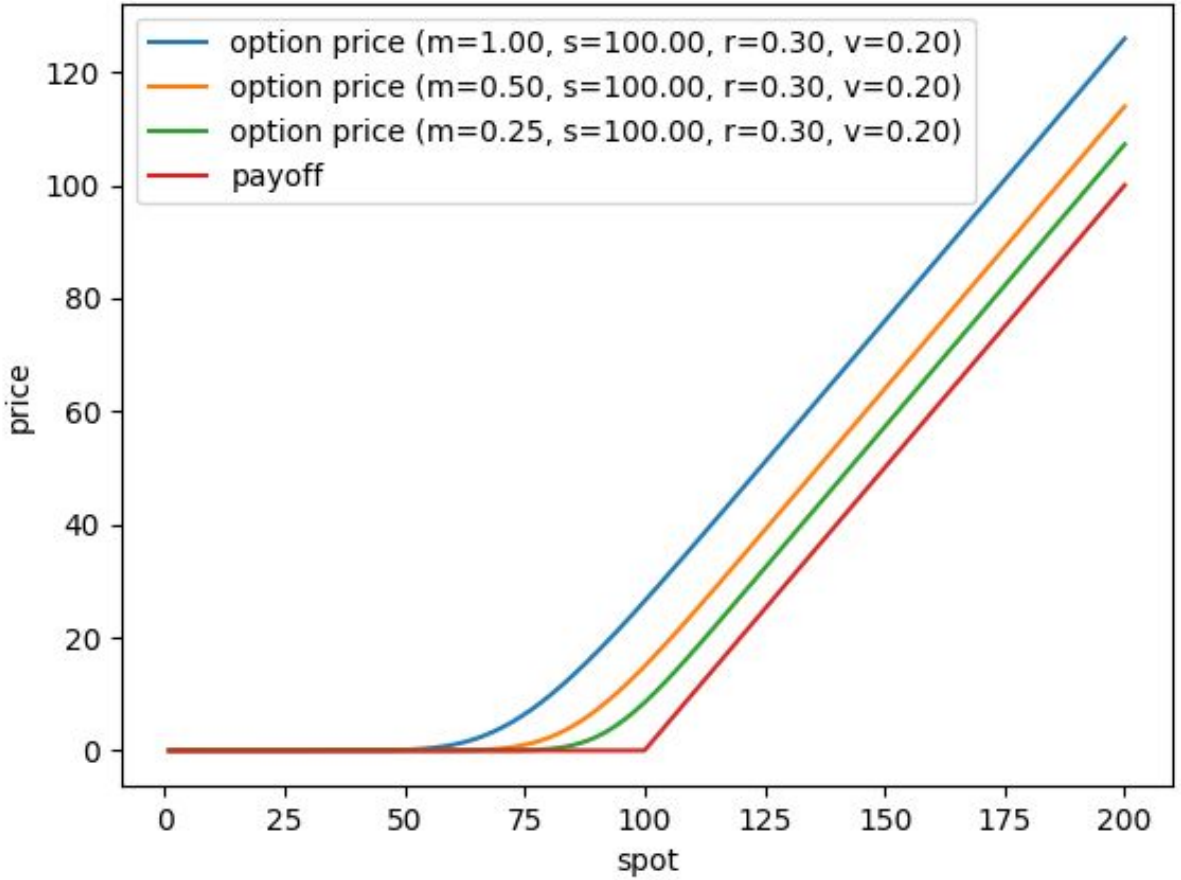
```
hello(“Carlos”)
    # hello Carlos


mean([1,2,3]) # 2
```

Functions 2

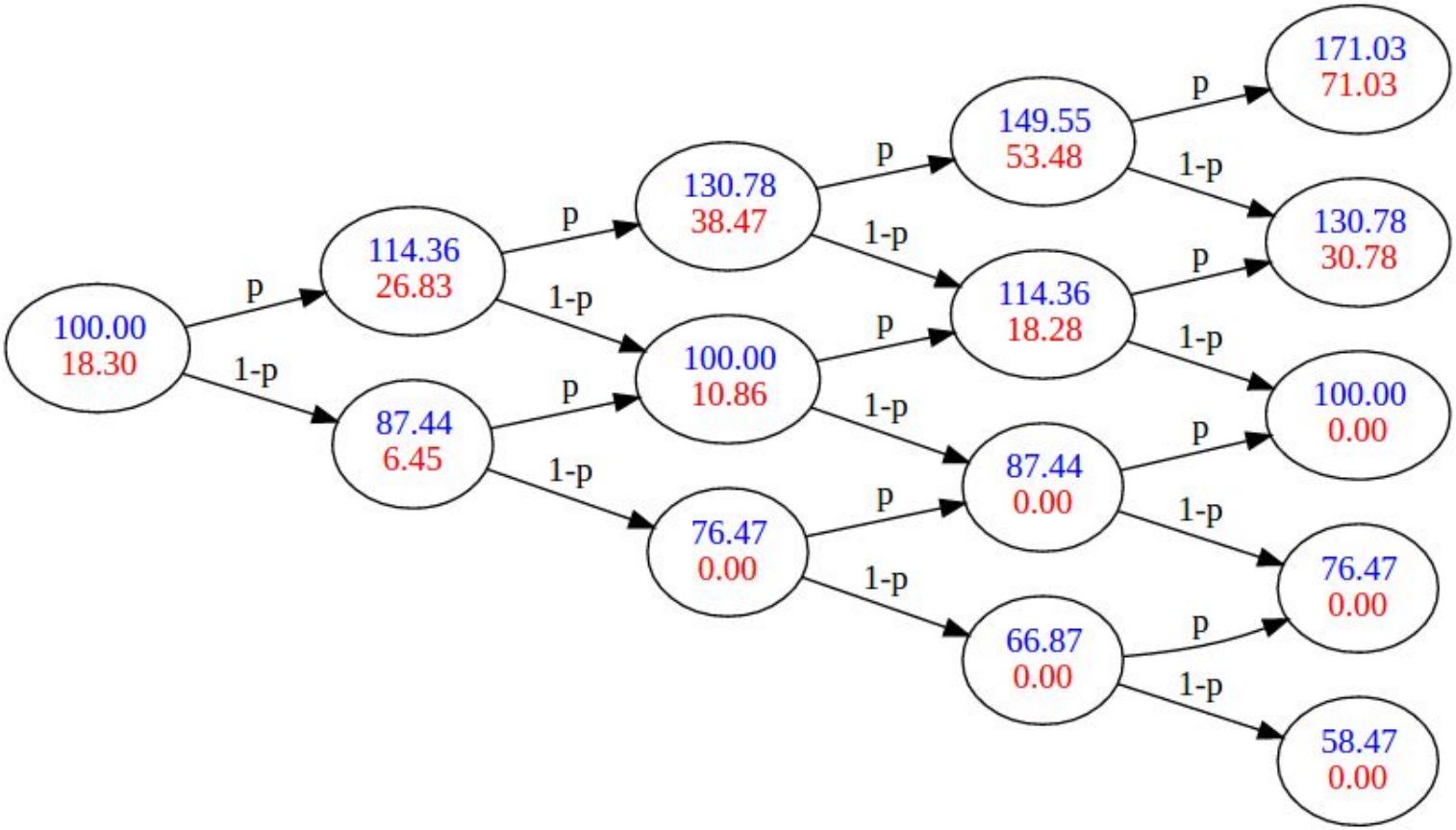
```
def f(x):
    return x

def g(x,a):
    return lambda x: x+a

def sqrt_func(x, func):
    return math.sqrt(func(x))


add1 = lambda x: x+1
```

```
f(5) # 5

g(5,1) # 5

sqrt_func(4,f) # 2

sqrt_func(4,g(5)) # 3



def add1(x):
    return x+1
```

Plots


option price (m=1.00, s=100.00, r=0.30, v=0.20)
option price (m=0.50, s=100.00, r=0.30, v=0.20)
option price (m=0.25, s=100.00, r=0.30, v=0.20)
payoff
price
spot
0
20
40
60
80
100
120
0
25
50
75
100
125
150
175
200

Graphviz for Binomial Tree



Python.org