

# Python

Introdução à Programação

SI1 - BSI

# Conteúdo

- História
- Instalação
- Apresentação da Interface
- Comandos Básicos
- Exercícios

# História

- Criada em 1989 pelo holandês Guido van Rossum no Centrum voor Wiiskunde em Informatica (CWI), em Amsterdã, Holanda
- Influenciada pela linguagem **ABC**, desenvolvida no CWI por Guido e outros nas décadas de 70 e 80
  - ABC tinha um foco bem definido: ser uma linguagem de programação para usuários inteligentes de computadores que não eram programadores: Físicos, Cientistas Sociais e até Lingüistas
- O projeto de sistema operacional distribuído **Amoeba** precisava de uma linguagem de script
  - Nasce o Python

# Bases e Fundamentos

- Elementos que eram **bem sucedidos** no ABC
- Estruturas de dados poderosas: **Listas, Dicionários, Strings**
- Usar **identação** para delimitar blocos, eliminando chaves
- Fácil de portar
  - Além do Amoeba, pode ser usada em **Unix, Linux, Macintosh** e **Windows** (multiplataforma)

# Ambiente

- **Universidade**
  - pessoas altamente especializadas para desenvolver e opinar sobre os elementos do projeto
- **Descontraído**
  - o nome Python vem da série de humor na TV Monty Python's Flying Circus
- **Sem prazos, Sem pressão**
  - o desenvolvimento não foi pressionado por estratégias de marketing, prazos, clientes ou qualquer outro fator que pudesse influenciar nas decisões de projeto, resultando em maior qualidade.
- **Software Livre**

# Características

- **Interpretada**
  - usa máquina virtual (PVM – Python Virtual Machine), facilita portabilidade.
- **Interativa**
  - pode-se programar interativamente, os comandos são executados enquanto são digitados. Facilita testes, desenvolvimento rápido e outros. Facilitadores estão presentes `help(obj)`.
- **Orientada a Objetos**
  - tudo (ou quase tudo) é objeto: números, strings, funções, classes, instâncias, métodos, ...
- **Tipagem Dinâmica**
  - A definição do tipo de um objeto é feita em tempo de execução. Um objeto tem tipo, uma variável, não.

# Para que serve?

- **Prototipação** rápida
- Desenvolvimento **Web**
- Acesso a **Banco de Dados**
- Manipulação de **String**
- Computação **numérica** e **científica**
- **Jogos**
- Aplicações **3D**
- Modelagem de **Hardware**

# Quem usa Python?





# Quem usa no Brasil?

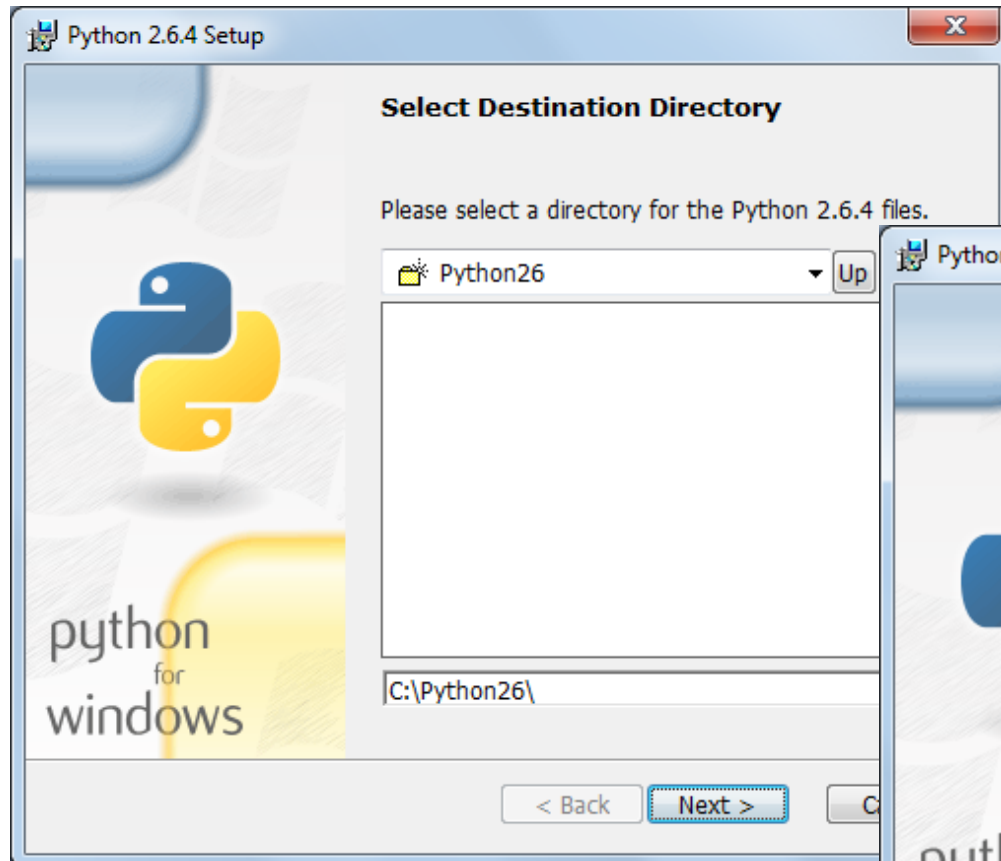
- **Embratel**
  - monitoramento das interfaces de backbone e clientes de internet e scripts de uso interno
- **Conectiva**
  - Gerenciamento de pacotes da distribuição Linux e ferramentas de uso interno
- **Async**
  - desenvolvimento de software de automação comercial
- **GPr Sistemas**
  - Desenvolvimento de aplicações sob encomenda, sistemas como monitoramento de transporte terrestre via satélite são as soluções já feitas
- Outras que utilizam o Python para sistemas Web, como **Varig**, **SERPRO**, **CertiSign**, **OAB/São Paulo**...

# Instalação

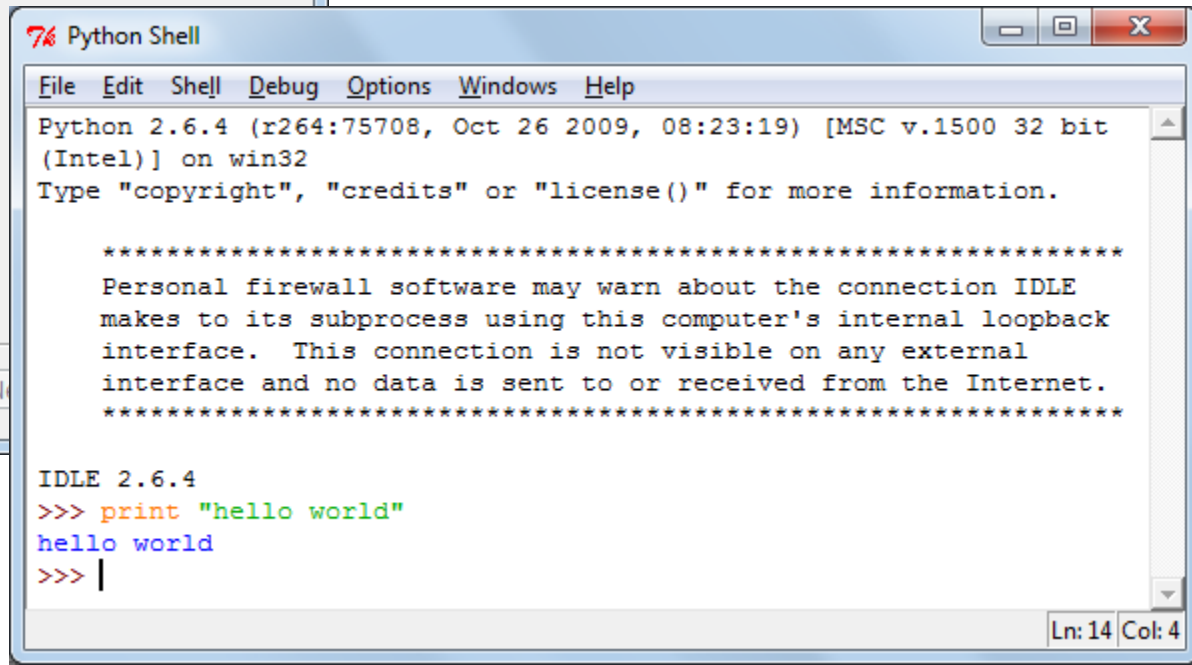
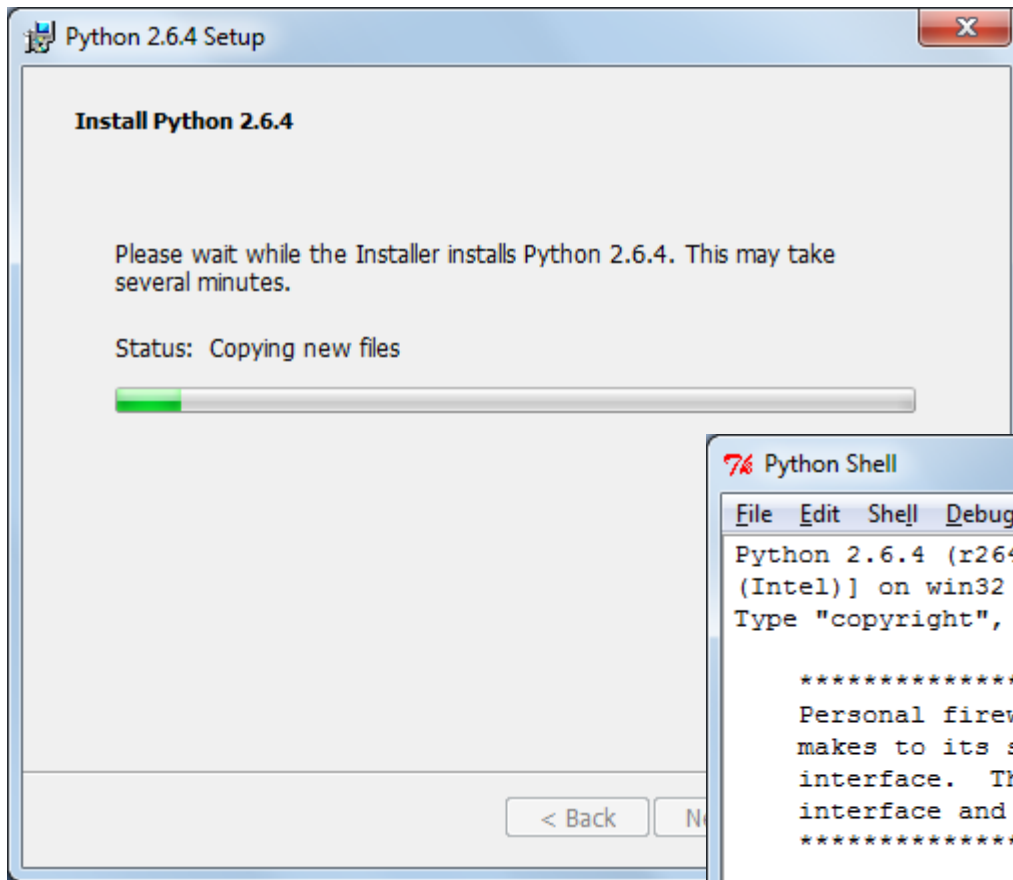
- <http://python.org>
  - Seção de Downloads



# Instalação



# Instalação



# Execução de Aplicações

- Para executar o programa `app.py`, basta digitar na linha de comando no diretório de instalação do Python:

```
C:\Python26> python C:\<pasta> app.py
```

- No **Linux** pode-se mudar a permissão para executar

```
#> chmod +x app.py
```

```
#> ./app.py
```

- No **Windows**, outra forma seria clicar duas vezes no fonte

# Usando o Ambiente

- Para iniciar o **shell** basta digitar o comando (no diretório de instalação):

```
#> python
```

- Quando o **shell** é iniciado aparecerá **>>>** indicando que ele está ativo e **pode receber comandos**

- Exemplo:

```
#> python
```

```
>>> print "HelloWorld!!!"
```

```
HelloWorld!!!
```

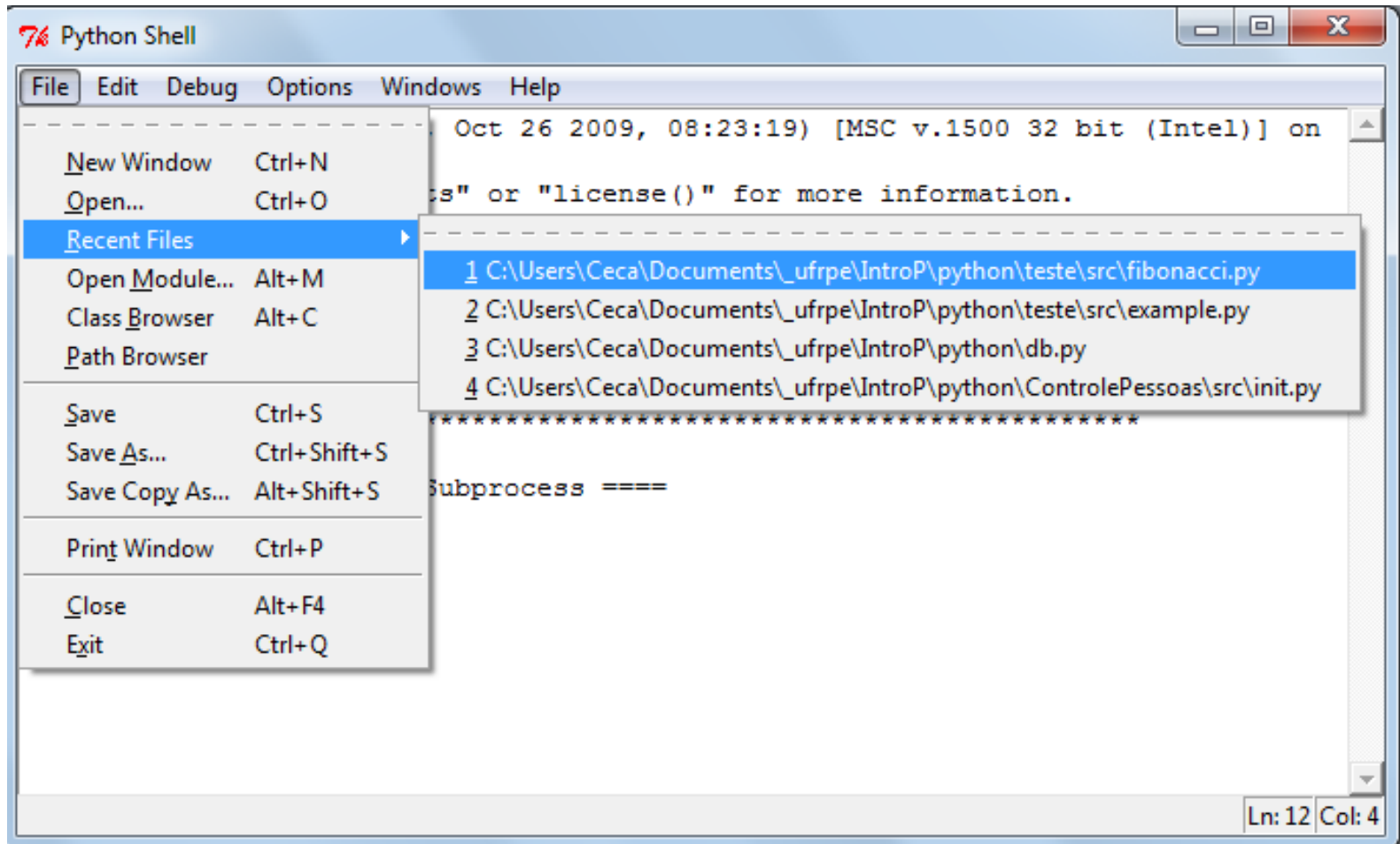
```
>>>
```

# Usando o Ambiente

- **Python Virtual Machine**
- O código fonte é compilado automaticamente gerando **bytecodes**
  - Arquivos compilados têm extensão **“.pyc”** ou **“.pyo”**
- Terminal interativo (**shell**)
  - Teste de **sintaxe**
  - Obter **ajuda**
  - Emitir comandos **individualmente**

# Usando o Ambiente

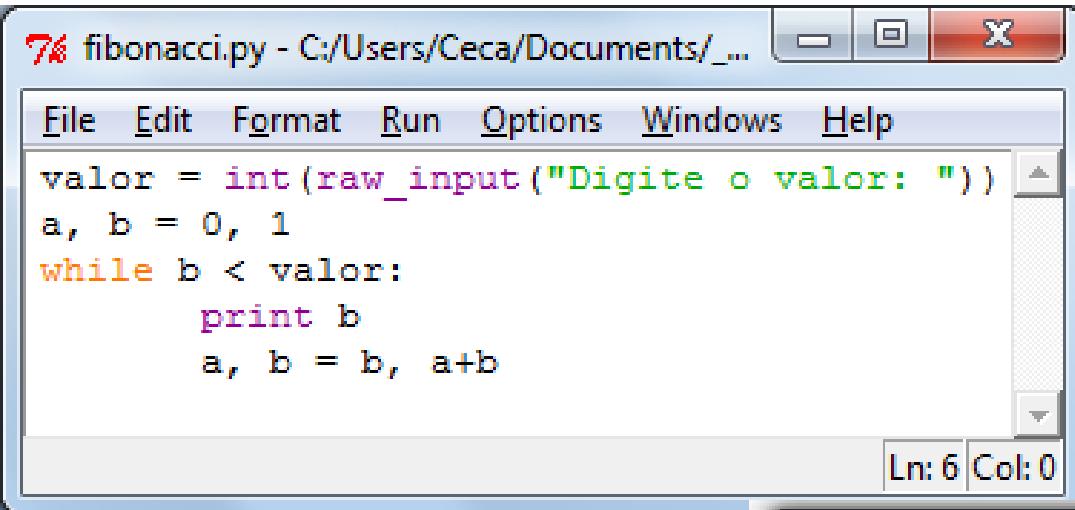
- O **shell** tem um **editor** de texto





# Usando o Ambiente

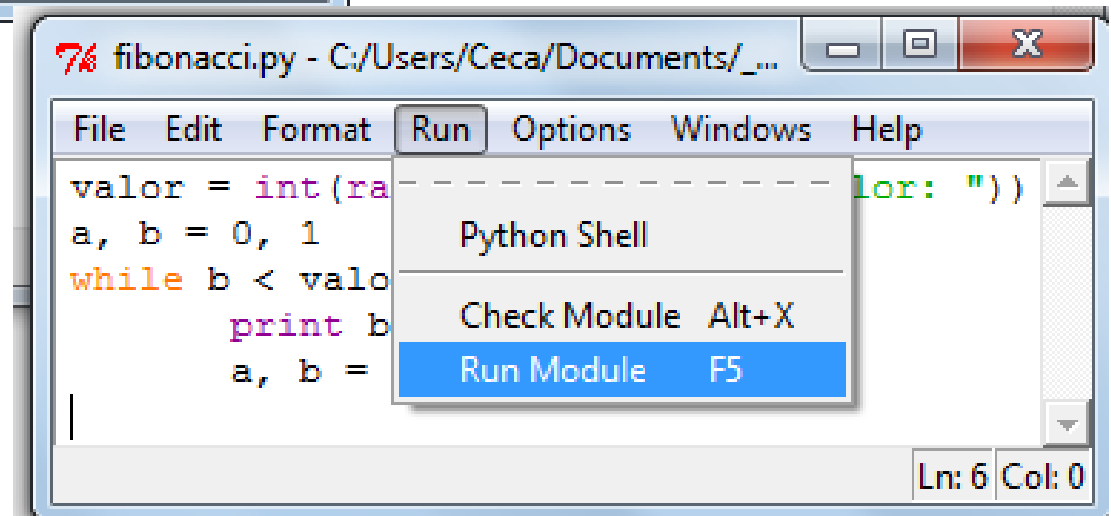
- E também executa programas...



A screenshot of a Python IDE window titled "fibonacci.py - C:/Users/Ceca/Documents/\_...". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Windows", and "Help". The code editor contains the following Python code:

```
valor = int(raw_input("Digite o valor: "))
a, b = 0, 1
while b < valor:
    print b
    a, b = b, a+b
```

The status bar at the bottom right shows "Ln: 6 Col: 0".



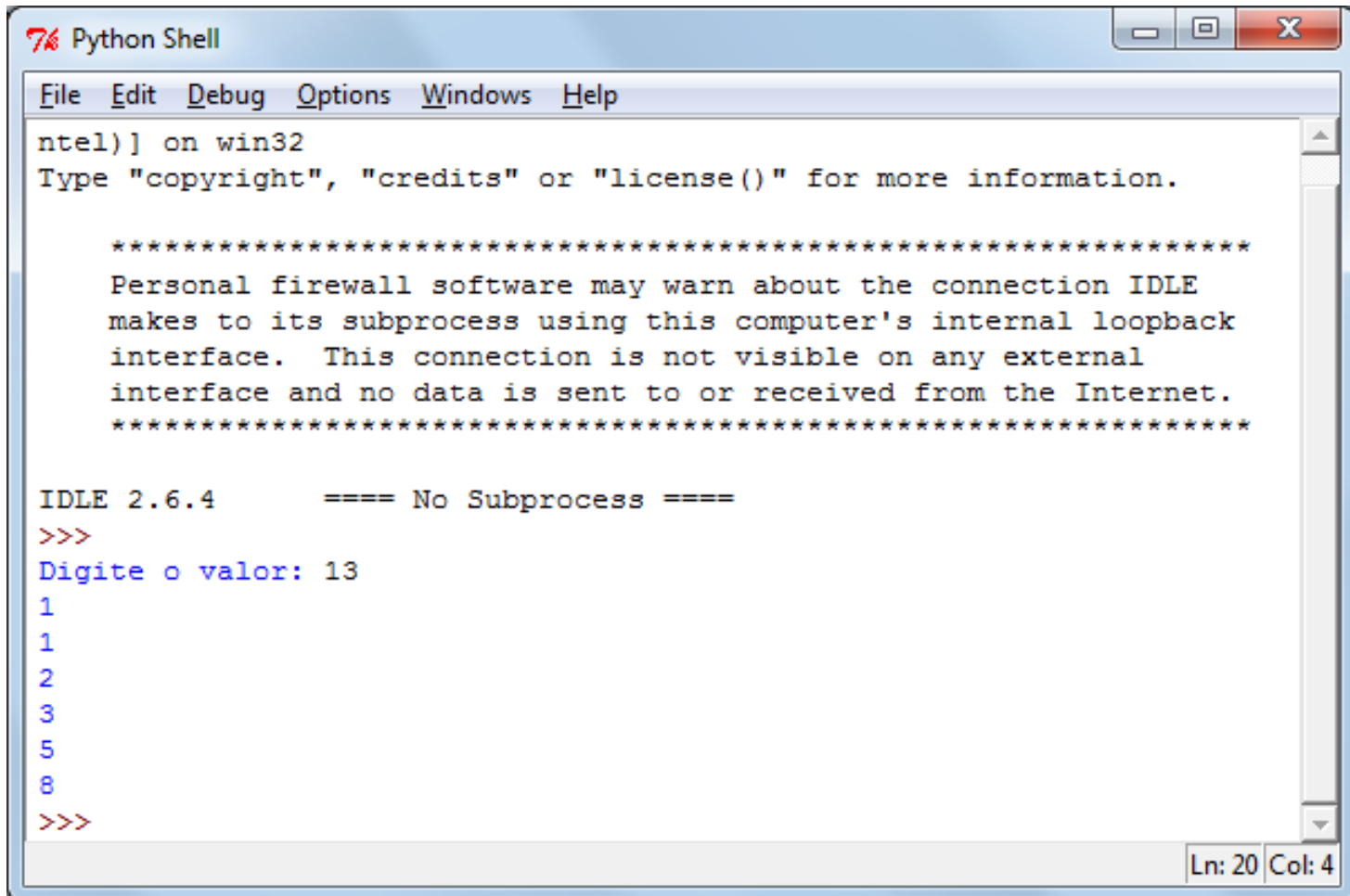
A screenshot of the same Python IDE window, but with the "Run" menu open. The menu items are:

- Python Shell
- Check Module Alt+X
- Run Module F5

The "Run Module F5" option is highlighted in blue. The code editor shows the same code as the previous screenshot, but the text is partially obscured by the menu. The status bar at the bottom right shows "Ln: 6 Col: 0".

# Usando o Ambiente

- E também executa programas...



The screenshot shows a window titled "Python Shell" with a menu bar containing "File", "Edit", "Debug", "Options", "Windows", and "Help". The main text area contains the following output:

```
ntel)] on win32
Type "copyright", "credits" or "license()" for more information.

*****
Personal firewall software may warn about the connection IDLE
makes to its subprocess using this computer's internal loopback
interface. This connection is not visible on any external
interface and no data is sent to or received from the Internet.
*****

IDLE 2.6.4      ==== No Subprocess ====
>>>
Digite o valor: 13
1
1
2
3
5
8
>>>
```

The status bar at the bottom right of the window displays "Ln: 20 Col: 4".

# Características

- Uso de ";" no fim de comandos não é obrigatório

- Dinamicamente tipada

- Exemplo

->>>a = 10

->>>a = "teste"

# Características

- Comentários de são feitos usando '#'
- Não possui declaração de tipos

– Java

```
int a = 0;
```

– Python

```
a = 0
```

- Não possui comandos declarativos (“óbvios”)

– Java

```
Aluno n = new Aluno();
```

– Python

```
n = Aluno()
```

# Dados e Operações

Operação	Resultado
$x + y$	Soma dos valores $x$ e $y$
$x - y$	Subtração de $x$ por $y$
$x * y$	Multiplicação de $x$ por $y$
$x / y$	Divisão de $x$ por $y$
$x // y$	Divisão de $x$ por $y$ , obs.: Pegando o piso.
$x \% y$	Resto da divisão de $x$ por $y$
$+x$	Não altera nada
$-x$	Inverte o sinal de $x$
$\text{abs}(x)$	Valor absoluto de $x$
$\text{int}(x)$	$x$ convertido em inteiro
$\text{long}(x)$	$x$ convertido em long
$\text{float}(x)$	$x$ convertido em float
$\text{complex}(\text{re}, \text{im})$	Um número complexo com parte real $\text{re}$ e imaginária $\text{im}$
$x ** y$	$x$ elevado a $y$
$\text{pow}(x, y)$	$x$ elevado a $y$

# Dados e Operações

Símbolo	ação comparativa
"<"	Menor que
"<="	menor ou igual
">"	maior que
">="	maior ou igual
"=="	igual (objeto ->referência)
"!="	diferente
"<>"	diferente
"is"	igualdade de objetos
"is not"	diferença de objetos

# Operações

<code>a += b</code>	<code>a = a + b</code>
<code>a -= b</code>	<code>a = a - b</code>
<code>a *= b</code>	<code>a = a * b</code>
<code>a /= b</code>	<code>a = a / b</code>
<code>a **= b</code>	<code>a = a ** b</code>
<code>a %= b</code>	<code>a = a % b</code>

# Expressões Booleanas

- Também chamadas expressões lógicas
- Resultam em verdadeiro (**True**) ou falso (**False**)
- Usadas em comandos *condicionais* e de *repetição*
- Analisar o estado de uma computação e escolher o **próximo passo**



# Expressões Booleanas

- Operadores
  - Relacionais:  $>$  ,  $<$  ,  $==$  ,  $!=$  ,  $>=$  ,  $<=$
  - Booleanos: **and** , **or** , **not**
- Expressão avaliada da esquerda para a direita

# Expressões Booleanas

```
>>> 1==1
```

```
True
```

```
>>> 1==2
```

```
False
```

```
>>> 1==1 or 1==2
```

```
True
```

```
>>> 1==1 and 1==2
```

```
False
```

```
>>> 1<2 and 2<3
```

```
True
```

```
>>> not 1<2
```

```
False
```

```
>>> not 1<2 or 2<3
```

```
True
```

```
>>> not (1<2 or 2<3)
```

```
False
```

# Comandos Básicos

```
>>> print 'Hello World!'  
Hello World!  
>>> print "Hello World!"  
Hello World!  
>>>
```

# Atribuição

```
>>> x=1
>>> x
1
>>> print x
1
>>> a,b=2,x
>>> print a, b
2 1
>>> a,b=5,x+a
>>> print a, b
5 3
>>>
```

# Entrada de Dados

- Função `raw_input()`: lê uma string do dispositivo de entrada padrão

```
>>> nome=raw_input("Digite seu nome: ")
Digite seu nome: Ana Paula
>>> print nome
Ana Paula
>>> idade=raw_input('Digite sua idade: ')
Digite sua idade: 13
>>> print idade
13
>>>
```

# Entrada de Dados

- Função `input()` : lê um valor do dispositivo de entrada padrão

```
>>> nome=input("Digite seu nome: ")
Digite seu nome: 'Ana Paula'
>>> idade=input('Digite sua idade: ')
Digite sua idade: 13
>>> print nome
Ana Paula
>>> print idade
13
>>>
```

# Saída de Dados

- Comando `print`

```
>>> print "Nome: %s, Idade: %d" %(nome, idade)
Nome: Ana Paula, Idade: 13
>>> print nome, idade
Ana Paula 13
```

# Tipos

- *Inteiro*

```
>>> type (idade)
<type 'int'>
```

- Longo

```
>>> a=10
>>> long(a)
>>> type (a)
<type 'long'>
```

- Float

```
>>> 3 / 2
1
>>> 3.0 / 2
1.5
```



# Tipos

- **String:** limitadas por aspas simples ou duplas

```
>>> print 'Alo "Mundo"!'
Alo "Mundo"!
>>> print "Alo 'Mundo'!"
Alo 'Mundo'!
>>> print '''
SyntaxError: EOL while
scanning single-quoted
string
>>> print '"\'
'''
```

# Cálculos

```
>>>2*2
```

```
4
```

```
>>>2/4
```

```
0
```

```
>>>2.0/4
```

```
0.5
```

```
>>>2-3
```

```
-1
```

```
>>>base=10
```

```
>>>altura=20
```

```
>>>area=base*altura
```

```
>>>print(area)
```

```
200
```

# Exercícios

1. Ler um número inteiro e exibir seu dobro.
2. Exibir a multiplicação de dois números reais informados pelo usuário.
3. Calcular a média aritmética de três notas fornecidas pelo usuário.
4. A imobiliária XYZ vende apenas terrenos retangulares. Faça um programa para ler as dimensões de um terreno e exibir a área do mesmo.

# Exercícios

5. Faça um programa para ler o salário de um funcionário e aumentá-lo em 20%. Imprima seu salário final.
6. Ler o valor de um cheque e escrever o quanto vai ser recolhido de CPMF. Considere que imposto recolhe uma taxa de 0,3%. Imprimir o valor do imposto.
7. Escreva uma seqüência de comandos para solicitar o nome e a matrícula do aluno. Em seguida exibir as informações no seguinte formato:
  - Nome do Aluno: “XXXXXXXX”, Matrícula: “ZZZZ”

# Bibliografia

- Python Tutorial -  
<http://www.python.org/doc/current/tut/tut.html>
- Dive into Python  
<http://www.diveintopython.org/>
- Python Brasil -  
<http://www.pythonbrasil.com.br/moin.cgi/DocumentacaoPython#head5a7ba2746c5191e7703830e02d0f5328346bcaac>
- Slides de Python: Rodrigo José Sarmiento Peixoto e Flávio Dias