MC-102 — Aula 01
Introdução à Programação de Computadores

Instituto de Computação – Unicamp

2016

Roteiro

Por que aprender a programar?

- Neste curso vocês aprenderão o básico para se criar programas.
- Exemplos de programas: Firefox , Angry Bird, MatLab, Spotify.
- Aprender a programar é uma atividade básica de um cientista ou engenheiro da computação.

Por que aprender a programar?

- *Eu não sou da computação !!!* Por que programar?
- Possíveis Respostas:
  - Porque é legal!
  - Posso ter algum retorno financeiro com isso!

Por que aprender a programar?

Eu sou das engenharias!
Alguns exemplos:

- Como engenheiro você deverá ser capaz de automatizar algum processo.
  - Você poderá criar programas para gerenciar e automatizar algum processo que hoje é manual.
- Como engenheiro você deverá ser capaz de desenvolver novas ferramentas ou protótipos.
  - Para criar ferramentas/protótipos você deverá fazer simulações computacionais para a realização de testes preliminares.
- Você poderá enxergar situações onde uma solução computacional pode trazer benefícios.
  - Mesmo que você não implemente (programe) a solução você poderá propô-la e será capaz de "conversar" com o pessoal de TI para implementar a solução.

Por que aprender a programar?

Eu sou das áreas científicas! Matemática, Física, Química etc.
Alguns exemplos:

- Como cientistas vocês devem propor uma hipótese e testá-la.
  - Em vários casos onde os sistemas podem ser " modelados matematicamente", são criados programas que fazem a simulação do sistema para checagem de uma hipótese.
- Você deverá resolver sistemas complexos de equações que não necessariamente podem ser resolvidos por softwares padrões (como MatLab).
  - Vocês deverão implementar seus próprios resolvedores.
- Simulações.
  - Muitos dos modelos propostos para explicar algum fenômeno são simulados computacionalmente. Implementar os modelos é uma tarefa básica.

O que esperar deste curso

- Vocês aprenderão o básico para desenvolver programas.
- Utilizaremos a linguagem Python.
- Vocês **NÃO** vão aprender a usar programas neste curso (como office, etc).
- Vocês **VÃO** ter porém, uma boa noção de como criar programas como o office, etc.

O que será necessário

- Você deverá ter acesso a um computador.
- Para criar um programa, utilizamos um *editor de texto* (para escrever o código do programa) e um *compilador/interpretador*.
- O compilador é o que transforma o código em um programa executável.
- O interpretador é um programa que executa diretamente os comandos da linguagem.
- Se você usa linux, MAC OS, ou Windows você poderá utilizar qualquer editor simples como *emacs, kyle, etc*.
- Será preciso instalar o compilador/interpretador *Python*. Você pode baixá-lo do site (instale a versão Python 3.*).
  https://www.Python.org/downloads/
- O Python já vem com um editor chamado IDLE que você poderá utilizar para escrever os seus programas.

O que será necessário

Para ir bem neste curso:

- Faça todos os laboratórios.
- Faça e implemente as listas de exercícios.
- E finalmente faça e implemente as listas de exercícios.

O que será necessário

Para ir bem neste curso:

- Faça todos os laboratórios.
- Faça e implemente as listas de exercícios.
- E finalmente faça e implemente as listas de exercícios.

O que será necessário

Para ir bem neste curso:

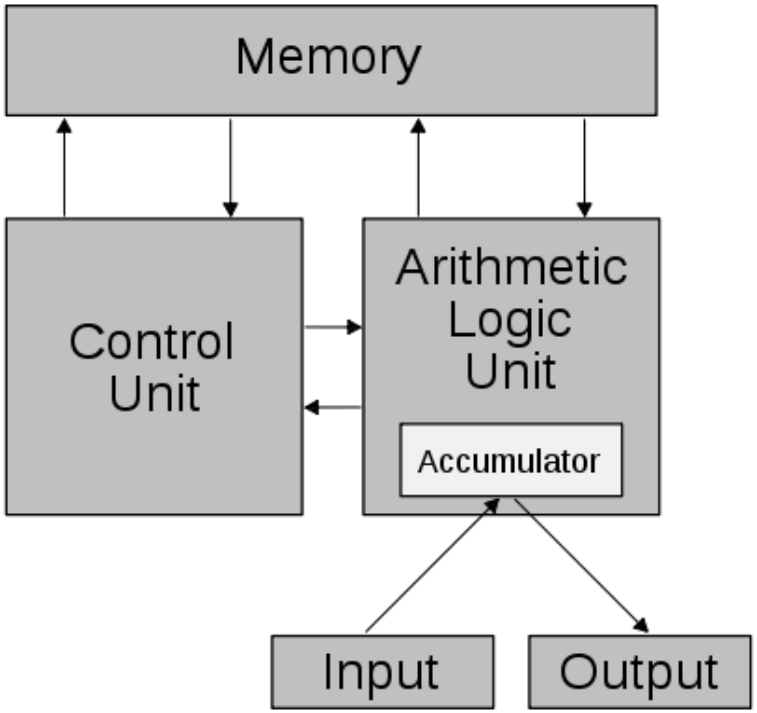
- Faça todos os laboratórios.
- Faça e implemente as listas de exercícios.
- E finalmente faça e implemente as listas de exercícios.

O que é um computador?

- Computador: o que computa, calculador, calculista. (dicionário Houaiss).
- Um computador é uma máquina que, a partir de uma entrada, realiza um número muito grande de cálculos matemáticos e lógicos, gerando uma saída.

Hardware e dispositivos

- Usualmente chamamos de *Hardware* todos os dispositivos físicos que compõem um computador.
- Temos por exemplo: CPU, Disco Rígido, Memória, etc.
- Estes dispositivos seguem uma organização básica como na figura (Arq. de Von Neumann).



Hardware e dispositivos

Todo o hardware opera com sinais digitais: sem energia e com energia. Normalmente usamos valores 0 e 1 para representar isto.

- Chamamos estes sinais de Bit $\rightarrow$ Valores 0 ou 1.
- Chamamos de *Byte* $\rightarrow$ um agrupamento de 8 bits.
- Todas as informações armazenadas no computador são representadas por números 0s e 1s. Informações como letras, símbolos, imagens, programas são todas vários 0s e 1s.

Software

São os programas que executam tarefas utilizando o hardware de um computador.

- Os softwares são compostos por um conjunto de instruções que operam o hardware.
- Temos abaixo, por exemplo, três instruções para um computador de 32 bits.
- Um software é composto por milhares de instruções deste tipo.

```
0100 0010 0011 0101 0101 0100 0011 0110
0100 1110 1100 1100 1001 0110 0110 1000
0000 0101 1111 1110 1101 0011 0000 1100
```

Organização básica de um ambiente computacional

- Um ambiente computacional é organizado como uma hierarquia de funções, onde cada uma é responsável por uma tarefa específica.

Programas de Aplicação
Compiladores
Sistema operacional
Hardware

Organização básica de um ambiente computacional

Programas de Aplicação.

- Como usuários, interagimos com os programas de aplicação.
- Neste curso iremos descer nesta hierarquia, para construirmos novos programas de aplicação.
- Para construir novos programas podemos escrever diretamente códigos digitais que serão executados por um computador.
- Uma maneira mais simples é usar um compilador para uma linguagem de programação específica.

Programas de Aplicação
Compiladores
Sistema operacional
Hardware

Organização básica de um ambiente computacional

Compiladores e Linguagens de Programação.

- Uma linguagem de programação é um conjunto de comandos que são mais "próximos" da linguagem humana do que os sinais digitais.
- Neste curso estamos interessados no estudo da *linguagem de programação Python*.
- Um *compilador* é um programa que lê um código de uma linguagem de programação e o transforma em um programa executável.
- O compilador realiza esta tarefa juntamente com um *assembler*.

```
for i in range (10):          loop:  add c, a, b       0100 0010 0011 0101 0101 0100
    c = a+b                         add i, i, 1       0110 0110 0111 0101 0101 0100
                                    bnq i, 10, loop   1111 0000 0111 0101 0101 0100
```

Organização básica de um ambiente computacional

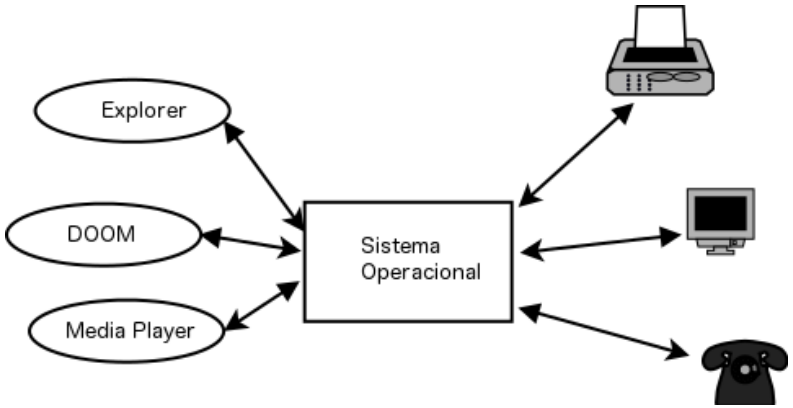
Compiladores e Linguagens de Programação.

- No caso específico de Python, o compilador traduz o código Python para o que chamamos de bytecode.
- O bytecode é um código de baixo nível que é executado em uma máquina virtual Python.
- Quando instalamos o compilador Python em um computador, na verdade instalamos tanto o compilador, o interpretador e a máquina virtual onde os programas em Python serão executados.
- O programa Python é também um interpretador pois ele pode ser usado para executar diretamente os comandos em Python.

Organização básica de um ambiente computacional

Sistema Operacional.

- Os programas possuem instruções que são executados no hardware.
- Mas o acesso ao hardware, como disco rígido, memória, processador, é controlado por um software especial conhecido como *sistema operacional.*
- O *sistema operacional* é o responsável pelo controle do hardware, incluindo segurança, gerenciamento de memória, dentre outros.
- Exemplos de sistema operacionais: *Windows, Mac OS, Linux, Android, iOS.*



Algoritmos

Ao criarmos um programa para realizar uma determinada tarefa devemos ser capazes de construir algoritmos.

- Algoritmo: Seqüência de passos, precisos e bem definidos, para a realização de uma tarefa.
- Algoritmos podem ser especificados de várias formas, inclusive em português.

Exemplo de algoritmo

Como ordenar as cartas de um baralho?

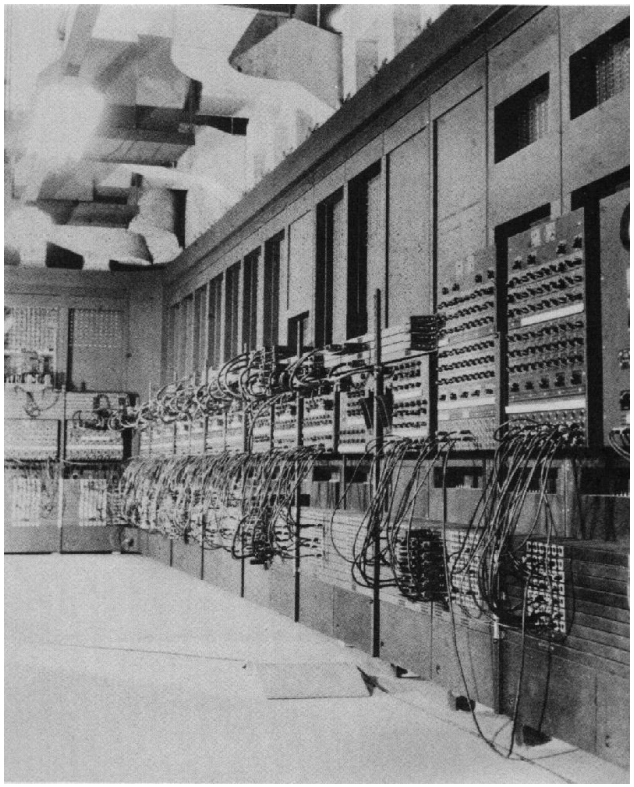
De algoritmos a programas

- Neste curso vamos aprender a criar algoritmos simples.
- Usaremos a linguagem Python para descrever os algoritmos.
- Os programas escritos em Python podem ser diretamente executados utilizando o interpretador Python.

Um pouco de história

Os primórdios da programação: programação em código absoluto ou binário (apenas 0s e 1s).

ENIAC



Um pouco de história

Uma melhoria: A Linguagem Assembly.

- Cria-se uma linguagem de baixo nível (Linguagem Assembly) para representar as instruções em código binário.
- Um programa, chamado montador ou assembler, faz a transformação em código absoluto.

```
LOOP:   MOV A, 3
        INC A
        JMP LOOP
```

Um pouco de história

Uma brilhante idéia: Criação de linguagens de alto nível e compiladores para estas.

- Mais distantes da máquina e mais próximas de linguagens naturais (inglês, português, etc.).
- Mesmo mais compreensíveis, elas não são ambíguas.
- Um compilador as transforma em código executável.

Exemplos de linguagens

- C
- Java
- Python

Primeiro programa em Python

Um programa em Python é um arquivo texto, contendo declarações e operações da linguagem. Isto é chamado de *código fonte*.

```
print("Ola turma de MC102!!")
```

Você pode salvar este arquivo como hello.py.

Como executar este programa

- Para executar este programa basta abrir um *terminal* e escrever:

```
$ python hello.py
Ola turma de MC102!
```

Relembrando

- Hardware e Software.
- Pilha de um ambiente computacional: Programas de Aplicações, Compilador, Sistema Operacional, Hardware.
- Código Binário, Assembly, Linguagem de Alto Nível.
- Algoritmos.

Informações Extras: O que são erros de compilação?

Caso o programa não esteja de acordo com as regras da linguagem, erros de compilação ocorrerão. Ler e entender estes erros é muito importante.

```
print("Ola turma de MC102!!)
```

```
$ python hello.py

  File "hello.py", line 3
    print("Ola turma de MC102!!
                               ^
SyntaxError: EOL while scanning string literal
```

Informações Extras: O que são erros de execução?

Acontecem quando o comportamento do programa diverge do esperado e podem acontecer mesmo quando o programa compila corretamente.

```
print("Ola turma de MC102!!%LKFDLJLK")
```

```
$ python hello.py
Ola turma de MC102!!%LKFDLJLK
```