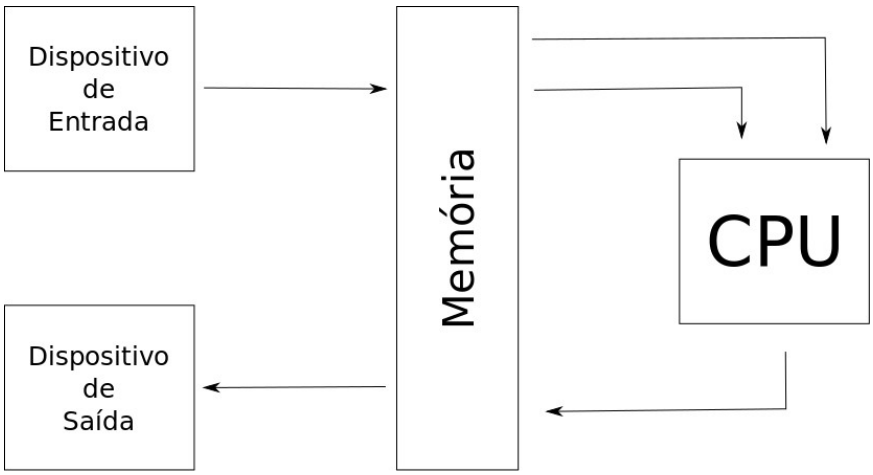
- O conjunto de instruções (*CdI* ou *ISA*) de um computador é o conjunto de passos (*instruções*) dos algoritmos que o mesmo reconhece.
  - Como exemplo, o robozinho do jogo viciante reconhece as instruções “Andar para frente”, “Pular”, “Virar à esquerda”, etc.
- As instruções do *CdI* de um computador *que resolve um problema* podem ser classificadas em quatro tipos principais:

(Observação: *Por enquanto*, vamos considerar apenas algoritmos que *não* têm estruturas de condição (Se..., Senão))

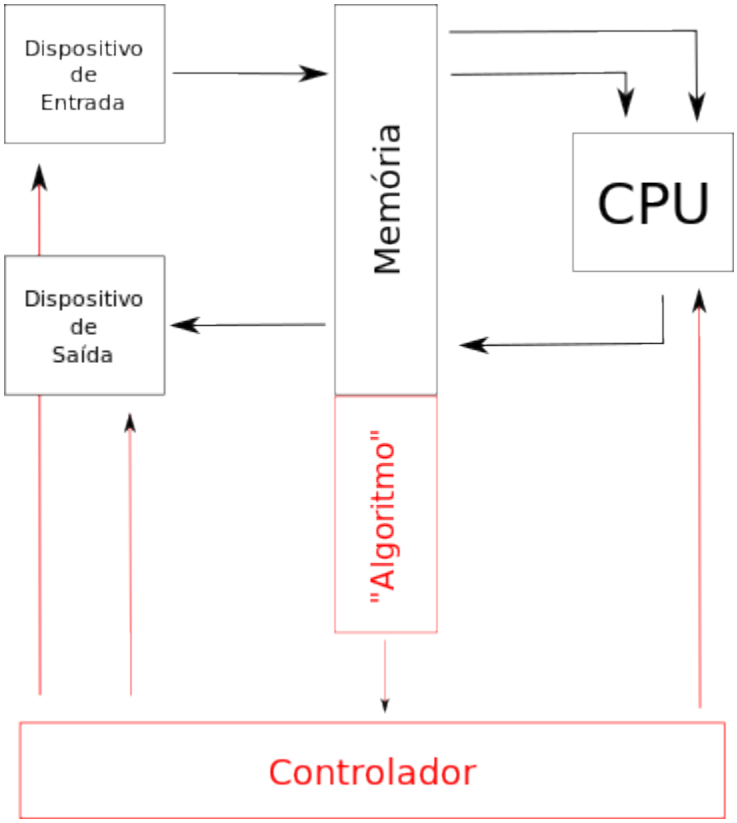
- “Escutar”, ou **Ler** dados
  - Dados são lidos dos *Dispositivos de entrada*: Teclado, Mouse, etc
    - Vamos usar apenas o teclado nesta disciplina
- “Dizer”, ou **Imprimir** dados
  - Dados são impressos nos *Dispositivos de saída:* Monitor, Impressora, etc
    - Vamos usar apenas o monitor nesta disciplina
- “Fazer conta”, ou **Calcular**
  - Cálculos são realizados pelo *Processador (CPU)*
- “Anotar dados” e “ler dados anotados”
  - Dados são anotados nas *Memórias:* Memória RAM, HD, etc
    - Vamos usar apenas a memória RAM nesta discplina

Os componentes estão ligados de acordo com a **Arquitetura de Von-Neumann.** Um diagrama ***simplificado*** da arquitetura é mostrado a seguir.



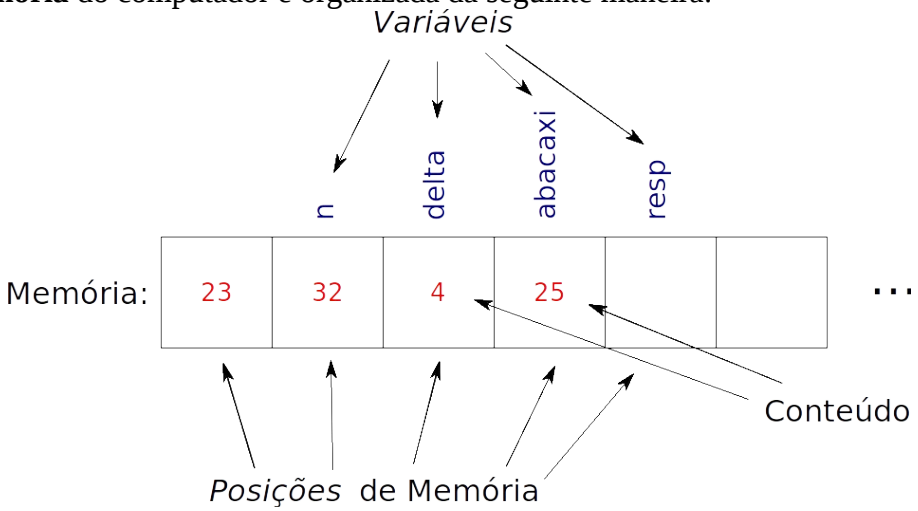
- O dispositivo de entrada alimenta a memória com a entrada fornecida pelo usuário
- A CPU realiza cálculos com dados escritos na memória, e também a alimenta
  - A cada *ciclo* do processador, um cálculo é realizado
  - A *frequência* de um processor é igual ao número de *ciclos por segundo* do mesmo.
    - Assim, um processador de 3.2 Ghz faz 3.435.973.836 =~ 3 bilhões de cálculos por segundo
- O dispositivo de saída é alimentado com dados da memória e os exibe para o usuário

- Uma **unidade controladora** é responsável por, a cada passo do “algoritmo”, *avisar* o circuito do que ele deve fazer.
  - Esta unidade fica dentro da CPU (mas desenhamos fora para um melhor entendimento)



- Assim, por exemplo, se o próximo passo do “algoritmo” é “ler um número”, o controlador irá enviar sinais para ativar o dispositivo de entrada.

- A **Memória** do computador é organizada da seguinte maneira:



- A memória é divida em “pedaços”. Cada “pedaço” é chamado de *posição* (ou *célula*) de memória
  - Em *uma* posição de memória pode-se anotar *uma* informação (*um* número ou *uma* letra)
  - Em computadores “modernos”, cada posição de memória tem tamanho de 4 *bytes* (= 32 *bits*).
    - Por exemplo, se sua memória RAM é de 2 Gb, ela tem (2 gigabytes)/(4 bytes por posição) = ½ gigaposições = ½ * 1024^3 = 536.870.912 posições de memória.
- A informação escrita em uma posição de memória é dita seu *conteúdo* (ou *valor*).
- O “nome” (rótulo) de uma posição de memória é dito ***variável.***
  - É pelo seu nome que uma posição é referenciada.

    - Tem esse nome porque referencia uma posição de memória cujo conteudo é *variável* (seu conteúdo pode ser alterado pela CPU, se o "algoritmo" assim quiser).
    - Os nomes são escolhidos pelo programador, arbritariamente.
    - O nomes não influenciam o funcionamento do algoritmo.
- Por questão de simplicidade, é aceito falar "variável" ao invés "posição de memória" quando conveniente.
    - Ex. Você pode dizer que "o conteúdo da *variável* delta é 4" ao invés de "O conteúdo da *posição de memória* chamada delta é 4". Ambas as formas são aceitas.

- O que deve-se colocar no "algoritmo" para que o controlador *saiba lê-lo*?
    - Codificações que representam as instruções suportadas pelo processador (ie. O CdI do processador)
    - Curiosidade: Quando a Intel lança um novo processador, ela divulga quais são as codificações para cada operação. O capítulo 5 de http://download.intel.com/products/processor/manual/325462.pdf descreve o CdI de uma família de processadores da Intel.
- Este é um exemplo de um algoritmo escrito com o CdI do Intel:

```
MOV %eax, (&n)
MOV %ebx, (&abacaxi)
ADD %eax, %ebx
DIV  %eax, 2
MOV (&resp), %eax
```

- Não é trivial escrever algoritmos assim.
- Uma **linguagem de programação** (LP) é o "idioma" em que um algoritmo pode ser "falado"
- A linguagem usada por um processador é dita *Linguagem de máquina* (ou *Assembly*)
    - MOV, DIV, etc é linguagem de máquina
- São outras linguagens: C, Python, Pascal, etc
- O mesmo algoritmo descrito acima pode ser escrito...
    - ...Em C: resp = (n*abacaxi)/2;
    - ...Em Python: resp = (n*abacaxi)//2
    - ...Em Pascal: resp := (n*abacaxi) div 2
- Note que os exemplos acima descrevem o **mesmo** algoritmo, escrito de maneiras diferentes.

- Um **Programa** é um algoritmo descrito em uma dada linguagem de programação
    - Um *compilador* converte um programa em uma dada linguagem para um programa em linguagem de máquina.
        - São linguagens compiladas: C, Pascal,...
    - Um *interpretador* simula a arquitetura de von Neumman em *software*.
        - É uma linguagem interpretada: Python

- Leia **todo** este documento novamente, substituindo *"Algoritmo"* (entre aspas), quando houver, por *Programa em Linguagem de Máquina.*