

MC102 – Algoritmos e Programação de Computadores

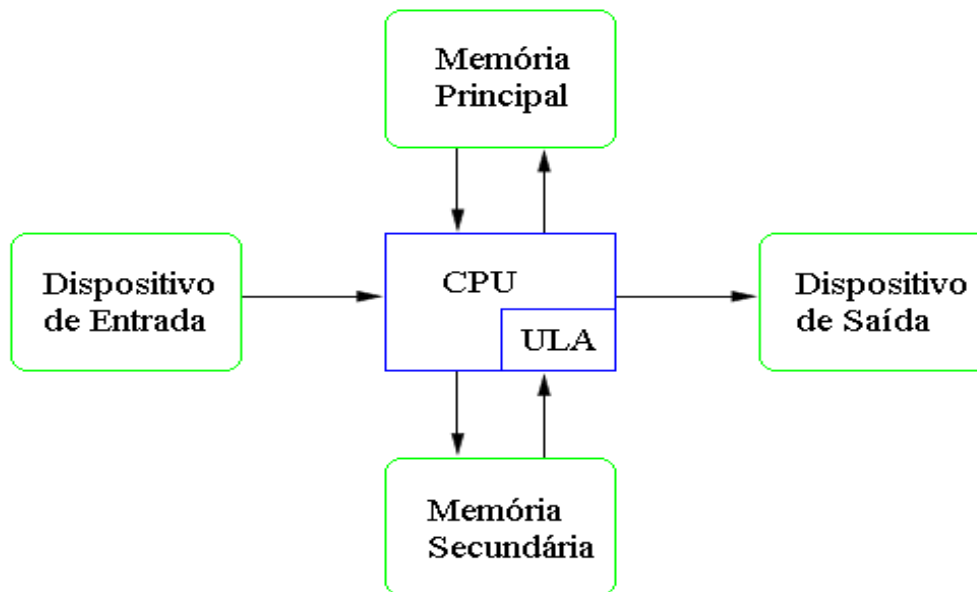
Prof. Fábio Augusto Menocci Cappabianco

Aula 1: Conceitos Básicos*

1. Organização do Computador

O computador é uma máquina que funciona mediante instruções ditadas por seres humanos ou por outras máquinas para executar tarefas. As aplicações são inúmeras, desde realizar cálculos e transmitir informações por uma rede até tocar uma música ou exibir um filme.

Um computador pode ser visto da seguinte forma:



Na figura acima, um computador é composto de uma unidade de processamento central (CPU), que contém uma unidade lógica e aritmética (ULA), de dispositivos de entrada, de dispositivos de saída, de memória principal e de memória secundária. As setas indicam a direção do fluxo de dados pelo computador.

- Dispositivos de entrada são utilizadas para receber informações ou instruções exteriores. Ex. Teclado, mouse, câmera de vídeo.
- Dispositivos de saída são utilizadas para exibir os resultados de uma computação. Ex. Monitor, impressora, caixa de som.
- Unidade Central de Processamento (CPU) ou processador é a parte principal do computador. Ela que realiza efetivamente as tarefas requisitadas, utilizando-se tanto das memórias como dos dispositivos de entrada e saída.

* baseada na aula do professor Alexandre Falcão.

- Unidade Lógica Aritmética (ULA) situa-se dentro da CPU. Ela possui a estrutura para realizar operações matemáticas.
- Memória Principal é uma memória rápida utilizada para armazenagem enquanto o computador está ligado. O tipo de memória principal mais utilizado é a memória RAM.
- Memória Secundária ou dispositivo de armazenamento guarda dados por um tempo indefinido. O computador pode ser desligado sem perder informações, porém este tipo de memória é mais lenta.

Obs. As memórias são consideradas dispositivos de entrada e saída, pois tanto fornecem quanto recebem dados da CPU.

2. Números Binários:

Os computadores não lêem números em base decimal. Os números utilizados são em base binária.

Bit: Número representado por '0' ou '1'.

Byte: Conjunto de oito bits. Ex. "00001110"

Palavra: Conjunto de bytes. Ex. 286 palavras de 2 bytes ou 16 bits, 386 até Pentium IV palavras de 4 bytes ou 32 bits, Pentium IV palavras de 8 bytes ou 64 bits.

Os números podem ser representados em hexadecimal para facilitar leitura. Ex. 1010b = Ah

Obs. A sigla KB significa 1024 bytes.

3 Definições

Dados: quaisquer tipos de informações manipuladas por um computador.

Comandos: instruções para que um computador execute tarefas. Ex. A instrução ADD faz com que o computador some dois números.

Programa ou **Software:** um conjunto de instruções que, quando executadas desempenham uma tarefa maior. Ex. jogo, animação, CAD.

Arquivo: parte de programas ou dados agrupados por alguma razão ou objetivo humano. Ilustração: guarda-roupas.

Hardware: algum dispositivo ou a CPU do computador.

Sistema operacional: programas para gerenciar, organizar e proteger o computador. Ex. Windows, Linux.

Linguagem de programação: consiste na sintaxe (gramática) e semântica (significado) utilizado para escrever um programa. Língua de comunicação com o computador.

Linguagem de programação de alto nível: linguagem de codificação de programa independente do tipo de máquina. Mais próxima de linguagem humana. Ex. C, Pascal, Basic, Fortran, Java.

printf("a"); - imprime uma letra a no monitor

Linguagem de programação de baixo nível: linguagem baseada em primitivas das funções básicas do computador. Depende do tipo de máquina. Ex. Assembly.

```
MOV DS,65  
MOV AH,09  
INT 21
```

Linguagem de máquina: semelhante à linguagem de baixo nível. Um programa em linguagem de máquina pode ser executado pelo computador.

```
0000100101011001  
1100111000011011  
1010101110001101
```

Obs. Código ilustrativo

Compilador: traduz um programa de linguagem de alto nível para um programa em linguagem de máquina. O programa compilado é guardado em arquivos diferentes do programa em linguagem de alto nível e pode ser executado pelo computador.

Assembler: traduz um programa de linguagem de baixo nível para linguagem de máquina.

Interpretador: traduz um programa de linguagem de alto nível para linguagem de máquina, sem guardar em arquivos novos. Enquanto a tradução é feita o programa é executado pelo computador.

Algoritmo: Esquema para se escrever um programa em linguagem de programação. Organizar a idéia do problema. Pode ser visto como uma receita de bolo. Na receita primeiro se descrevem os ingredientes e depois o modo de preparo. No algoritmo, descreve-se primeiro os recursos necessários e depois o modo de execução do programa.

4. Objetivos do curso

Os objetivos do curso são de aprender:

- i. descrever problemas e criar algoritmos para eles;
- ii. uma linguagem de programação de alto nível;
- iii. escrever programas na linguagem de alto nível a partir de algoritmos;
- iv. compilar o programa em linguagem de alto nível;
- v. executar o programa;
- vi. testar e corrigir erros de programas;

No nosso caso a linguagem de programação a ser utilizada é a linguagem C. O compilador utilizado será o gcc ou o Turbo C.

5. Exemplo de Programa.

Problema: somar dois números e colocar o resultado no monitor.

Algoritmo:

Quais os recursos necessários?

- Um dispositivo de entrada dos dois números - teclado.
- Um dispositivo de saída – monitor.
- Um espaço na memória principal para guardar temporariamente os números utilizados.
- CPU para executar a soma.

Como executar?

- Ler os números do dispositivo de entrada e guardar na memória;
- Fazer soma;
- Guardar o resultado da soma na memória;
- Escrever resultado no monitor;

Memória principal organizada em variáveis. Para se utilizar um pedaço da memória, para guardar uma variável, escolhe-se o tipo da variável e dá-se um nome para ela. Alguns tipos: inteiro, ponto flutuante, caractere, vetor.

Programa em linguagem C:

```
int main(){
    int a;
    int b;
    int t;
    scanf("%d",&a);
    scanf("%d",&b);
    t = a + b;
    printf("%d\n",t);
    return 0;
}
```

Melhorando código: Caso a variável 'a' não seja necessária depois da soma

```
int main(){
    int a,b;
    scanf("%d%d",&a,&b);
    a = a + b;
    printf("%d\n",a);
    return 0;
}
```