O funcionamento de computadores e introdução ao C

1.Introdução

Independentemente das diferenças no aspecto físico, praticamente todos os computadores podem ser considerados como divididos em seis unidades lógicas ou seções, são elas:

- **Unidade de entrada (input unit):** Esta é a seção de recepção do computador. Obtém informações dos vários dispositivos de entrada e as coloca à disposição de outras unidades para que possam ser processadas
- **Unidade de saída (output unit):** Esta é a seção de expedição do computador. Ela leva as informações que foram processadas pelo computador e as envia aos vários dispositivos de saída para torná-las disponíveis para o uso no ambiente externo ao computador.
- **Unidade de memória (memory unit):** Essa seção de armazenamento do computador, com acesso rápido e a capacidade relativamente baixa. Ela conserva as informações que foram fornecidas através da unidade de entrada para que possam estar imediatamente disponíveis para o processamento quando se fizer necessário. Também conserva as informações que já foram processadas até que sejam enviadas para os dispositivos de saída pela unidade de saída.
- **Unidade aritmética e lógica (ALU):** Esta é a seção de fabricação do computador. Ela é responsável pela realização dos cálculos como adição, subtração, multiplicação e divisão. Ela contém os mecanismos de decisão que permitem ao computador comparar dois itens, por exemplo.
- **Unidade Central de Processamento (CPU):** Esta é a seção administrativa do computador. Ela é responsável pela supervisão do funcionamento das outras seções. A CPU informa à unidade de entrada quando as informações devem ser lidas na unidade de memória, informa à ALU quando as informações da unidade de memória devem ser utilizadas em cálculos e informa à unidade de saída quando as informações devem ser enviadas da unidade de memória para determinados dispositivos de saída
- **Unidade de memória secundária (secondary storage unit):** Esta é a seção de armazenamento de alta capacidade e de longo prazo do computador. Os programas ou dados que não estiverem sendo usados ativamente por outras unidades são colocados normalmente em dispositivos de memória secundária até que sejam outra vez necessários.

Os primeiros computadores eram capazes de realizar apenas um trabalho ou tarefa de cada vez. Esta forma de funcionamento de computadores é chamada frequentemente de processamento em lotes de usuário único. O computador executa um único programa de cada vez enquanto processa dados em grupos (*batches*).

À medida que os computadores se tornaram mais poderosos, tornou-se evidente que o processamento em lotes de usuário único raramente utilizava com eficiência os recursos do computador. Muitos trabalhos e tarefas poderiam ser executados simultaneamente. Isto é chamado de **multiprogramação**. Ela envolve operações simultâneas de muitas tarefas do

computador, onde o computador compartilha seus recursos entre as tarefas que exigem sua atenção.

Timesharing é um caso especial de multiprogramação no qual os usuários podem ter acesso ao computador através de dispositivos de entrada/saída ou terminais. Em um sistema computacional típico de timesharing, pode haver dezenas de usuários compartilhando o computador ao mesmo tempo. O computador faz isso tão rapidamente que pode executar o serviço de cada usuário várias vezes por segundo.

Os programadores hoje em dia escrevem instruções em várias linguagens de programação, algumas entendidas diretamente pelo computador e outras que exigem passos intermediários de tradução. Elas podem ser divididas em 3 tipos gerais: linguagens de máquina, linguagens *assembly* e linguagens de alto nível.

2.A linguagem de programação C

C é uma linguagem de propósito geral, que é geralmente associada com os sistemas operacionais UNIX, dos quais foram desenvolvidos baseados nele. Ela não é associada a nenhum sistema operacional, e apesar de ser chamada de **linguagem de sistemas operacionais**, porém, seu uso se estende além deste campo. O C foi desenvolvido a partir das linguagens BCPL e o B. A linguagem BCPL foi desenvolvida em 1967 por Martin Richards para escrever software de sistemas operacionais e compiladores. Ken Thompson modelou muitos recursos da linguagem B e usou para criar as primeiras versões do sistema operacional UNIX. A linguagem C foi desenvolvida por Dennis Ritchie.

O C independe do hardware, logo, é possível escrever programas em C que sejam portáteis para a maioria dos computadores.

A linguagem providencia fluxo de controle fundamental para a construção de programas bem estruturados. Declarações grupais, tomadas de decisões (***if-else***), seleção de um possível caso (***switch***), loops com teste de terminação no início (***while***) e no fim (***do-while***). Funções podem retornar valores básicos, estruturas, uniões ou ponteiros. Qualquer função pode ser chamada recursivamente. Variáveis locais são tipicamente “automáticas”, ou criadas de novo com cada chamada de função. Definições de funções podem não ser aninhadas, mas variáveis podem ser declaradas no estilo estrutura de blocos.

Também não providencia nenhuma operação para objetos compostos como strings de caracteres, sets, listas ou arrays. Não existem operações que manipulem arrays inteiros ou até mesmo strings, porém, estruturas podem ser copiadas como uma unidade. Não são definidos armazenamentos locais facilitados além de alocação estática. Não existe coletor de lixo ou coletor de pilhas. C não providencia facilidades de I/O, não existe declarações do tipo READ ou WRITE e acesso a arquivos pré-construídos.

Porém, C oferece controle de fluxo single-thread, testes, loops, agrupamento e subprogramas, mas nada de multiprogramação, programação paralela, sincronização e co-rotinas.

Uma coleção de cabeçalhos provê acesso uniforme a declarações de funções e tipos de dados. A maioria destes cabeçalhos está na **standard I/O library do sistema UNIX**. As bibliotecas padrão, se não utilizadas, deve-se evitar chamá-las e incluí-las no código.

C é uma linguagem de baixo nível, isto é, ela trabalha praticamente com o que computadores trabalham como caracteres, números e endereços de memória que podem ser combinados e alterados no mesmo. Embora C possa se acertar muito bem com o hardware de vários computadores, ele é dependente da arquitetura da máquina usada. Com cuidado, é possível escrever códigos portáteis para outros tipos de máquina, sem a necessidade de alteração do código.

C não é uma linguagem fortemente tipada, mas se desenvolveu. A definição original de C mal vista, porém permitida são o uso de ponteiros e inteiros. O padrão hoje em dia exige uma declaração bem feita e conversões explícitas que são reforçadas por bons compiladores. Estes compiladores, sempre irão apontar erros de sintaxe durante a etapa de compilação do código.

3.A biblioteca padrão do C

Os programas em C consistem em módulos ou elementos denominados *funções*. É possível programar todas as funções de que precisa para formar um programa em C, mas a maioria dos programadores tira proveito de um excelente conjunto de funções denominado *C Standard Library*. Usar funções existentes evita reinventar a roda. No caso das funções *standard* ANSI, você sabe que elas foram bem desenvolvidas cuidadosamente e sabe que, por estar usando funções disponíveis em praticamente todas as implementações do ANSI C, seus programas terão uma grande possibilidade de serem portáteis. Usar as funções da biblioteca *standard* C em vez de escrever suas próprias funções pode também melhorar o desempenho do programa.

4.Os fundamentos do Ambiente C

Todos os sistemas C são constituídos geralmente de três partes: o ambiente, a linguagem e a C *standard library*. Os programas em C passam normalmente por 6 fases para serem executados:

- **Edição:** Esta fase consiste na edição de um arquivo. Isto é realizado com um programa editor. O programa editor digita um programa em C com o editor e faz as correções necessárias. O programa então é armazenado em um dispositivo de armazenamento secundário. Os nomes de programas em C devem ter a extensão **.c**. Em seguida o programador emite o comando *compilar* o programa.
- **Pré-processamento:** Um programa *pré-processador* é executado automaticamente antes da fase de tradução começar. O pré-processador C obedece a comandos especiais chamados de *diretivas do pré-processador* que indicam que devem ser realizadas determinadas manipulações no programa antes da compilação. Estas manipulações

consistem normalmente em incluir outros arquivos no arquivo a ser compilado e substituir símbolos especiais por texto de programa.

- **Compilação:** O compilador traduz o programa em C para o código de linguagem de máquina.
- ***Linking*:** Normalmente os programas em C contêm referências a funções definidas em outros locais, como nas bibliotecas padrão ou nas bibliotecas de um grupo de programadores que estejam trabalhando em um determinado projeto. Assim, o código-objeto produzido pelo compilador C contém normalmente lacunas devido à falta dessas funções. Um *linker* faz a ligação do código-objeto com o código das funções que estão faltando para produzir uma imagem executável.
- **Carregamento:** Um programa deve ser colocado na memória antes que possa ser executado pela primeira vez. Isto é feito pelo carregador (*loader*), que apanha a imagem executável do disco e a transfere para a memória
- **Execução:** Sob controle de sua CPU, executa as instruções do programa, uma após a outra