MC102 – Algoritmos e Programação de Computadores

2ª Aula – Programa, entrada e saída de dados

1. Objetivos

- Falar sobre programa de computador, diferenciando programa em linguagem de máquina, de programa em linguagem de montagem e de programa em linguagem de alto nível
- Apresentar a Linguagem C e a estrutura geral de um programa nesta linguagem de alto nível
- Falar sobre entrada e saída de dados e apresentar as funções `printf()` e `scanf()` da Linguagem C
- Apresentar operadores aritméticos da Linguagem C

2. Motivação

Toda a linguagem de programação apresenta uma estrutura. Para que se dê início à atividade de programação de computadores, é indispensável conhecer ao menos a estrutura e as características gerais da linguagem de programação escolhida.

Comandos de entrada e saída de dados possibilitam a um programa aplicar o mesmo algoritmo a diferentes valores de entrada e produzir dados para outros programas ou mesmo úteis a um ser humano.

3. Conceitos

3.1 Programa de computador (Revisão)

A linguagem compreendida pelos computadores é a **linguagem de máquina**, cujo alfabeto é formado apenas por duas letras: os dígitos binários[1] (ou *bits*) 0 e 1. Escrever programas em linguagem de máquina utilizando apenas 0*s* e 1*s*, entretanto, é uma tarefa chata e bastante sujeita a erros.

> *"O uso da máquina para programar a própria máquina foi a forma encontrada pelos pioneiros da programação para traduzir programas escritos em uma linguagem simbólica para a linguagem da máquina."* (Hennessy e Patterson, 2000)

A linguagem de montagem (ou *assembly*) foi criada para facilitar a programação de computadores, mas ela ainda obriga o programador a escrever uma linha para cada instrução a ser executada pela máquina, forçando-o a raciocinar como máquina. O programa que traduz a linguagem de montagem para a linguagem de máquina é denominado **montador** (ou *assembler*).

[1] O uso de números binários para representar instruções e dados é a base da teoria computacional.

Instruções são conjuntos de *bits* inteligíveis pelo computador e que podem ser associadas a números. Por exemplo, os *bits* 1000 1100 1010 0000 informam a um determinado computador que ele deve somar dois números.

Já as linguagens de programação de alto nível oferecem uma série de vantagens ao programador: (1) permitem que ele raciocine de uma forma mais natural, usando palavras em inglês e notações algébricas; (2) colaboram para sua produtividade, por serem mais concisas que as linguagens de montagem; (3) e favorecem a portabilidade, uma vez que programas escritos em linguagens de alto nível são independentes do computador no qual foram desenvolvidos.

3.2 Linguagem C

Atualmente, os programadores têm a sua disposição diferentes linguagens de programação de alto nível. A Linguagem C é uma delas que, em geral, é compilada. Portanto, como apresentado na aula anterior, um programa codificado em linguagem C é traduzido pelo compilador em um programa executável, ou seja, codificado em linguagem de máquina. Às vezes essa tradução da linguagem de alto nível para a linguagem de máquina também passa por um montador, conforme apresentado na Figura 1, a seguir.

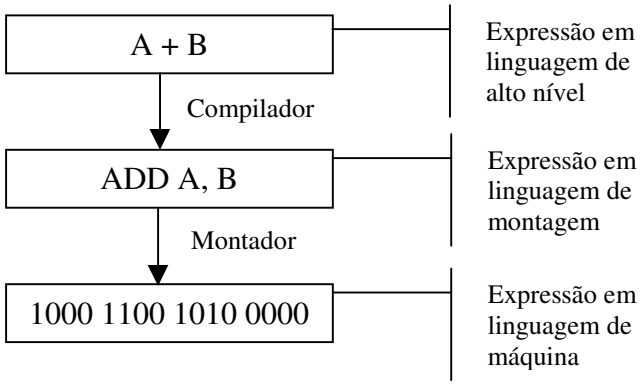

Figura 1: Representação de uma expressão em diferentes linguagens computacionais

Sobre a Linguagem C:

- É uma linguagem de programação de propósito geral.
- Seus tipos de dados fundamentais são caracteres, inteiros e ponto flutuante de diversos tamanhos. Também oferece uma hierarquia de tipos de dados derivados criados com apontadores, vetores, estruturas e uniões.
- Expressões são formadas com operadores e operandos.
- Provê construções fundamentais de fluxo de controle exigidas por programas bem estruturados[2]: agrupamento de comandos, tomada de decisão (*if-else*), seleção de um dentre um conjunto de casos possíveis (*switch*), laços com teste de término no topo (*while*, *for*) ou no fundo (*do*), e saída antecipada do laço (*break*).
- Apresenta facilidades para modularização.

[2] Lembrar do conceito de algoritmo.

Programa em C:

> *"Um programa em C, independentemente de seu tamanho, consiste em funções e variáveis. Uma função contém comandos que especificam as operações de computação a serem feitas, e as variáveis armazenam valores usados durante a computação."* (Kernighan e Ritchie, 1989:6)

```
#include <stdio.h>

main() {

printf ("primeiro programa\n");

}
```

Figura 2: Um programa em Linguagem C

- `#include <stdio.h>`

  Informa ao compilador para incluir informação sobre a biblioteca padrão de entrada/saída.

- `main()`

  É a função principal do programa escrito em C. Um programa C começa a ser executado do início da função **`main`**. Os comandos de uma função são delimitados por chaves **{ }**.

- `printf ("primeiro programa\n");`

  É uma chamada à função **`printf`**, que imprime na saída padrão (a tela do computador). Neste caso o que é impresso na tela é a cadeia de caracteres "`primeiro programa\n`". `\n` indica nova linha.

Tabela 1: Caracteres de Escape da Linguagem C

`\a`	Caracter de alerta (*bell*)
`\b`	Retrocesso
`\f`	Alimentação de formulário
`\n`	Nova linha
`\r`	Retorno de carro
`\t`	Tabulação horizontal
`\v`	Tabulação vertical
`\\`	Contra-barra
`\?`	Ponto de interrogação
`\'`	Apóstrofo
`\"`	Aspas

\ooo	Número octal
\xhh	Número hexadecimal

Programa C – Comentários:

"Comentários são observações que o programador faz no código do programa para poder entendê-lo melhor mais tarde, ou permitir que outros possam entender mais facilmente o programa. É parte da documentação do programa." (Trevisan, 2002)

```
/* Um programa em linguagem C
 * Exemplo de uso de comentários
 */

#include <stdio.h>


main() {

//Imprime uma string:
printf ("primeiro programa\n");


}
```

Figura 3: Comentários em um programa escrito em Linguagem C

Em C, todo comentário começa com o par de caracteres /* e termina com */. Não deve haver nenhum espaço entre o asteriscos e a barra. O compilador ignora qualquer texto entre os símbolos de comentários. Atenção: um comentário não pode conter outro comentário!

A maioria dos compiladores C também aceita comentários de apenas uma linha, ou seja, que terminam no final da linha. Neste caso, o comentário é feito colocando-se // no início do texto de comentário.

3.3 Entrada e Saída

Retomar o papel da entrada e da saída de dados para um algoritmo computacional, exemplificando com exemplos do cotidiano (ex. uso do caixa eletrônico, caixa de supermercado, *login* em um sistema).

Na aula introdutória de laboratório, foi utilizado o programa gcc:

```
gcc primeiro-programa.c -o primeiro-programa
```

Há um exemplo de uso da função `scanf` no exemplo (1), mas esta função deve ser melhor trabalhada junto ao tópico "Variáveis, comandos de atribuição, constantes". Já é possível, entretanto, apresentar a importância de existir uma maneira de um programa armazenar dados.

Exercitar o uso da função `printf()`. Ver exemplos (2) e (3).

3.4 Operadores Aritméticos

Tabela 2: Operadores Aritméticos Binários da Linguagem C

+	Soma
–	Subtração
*	Multiplicação
/	Divisão
%	Módulo da divisão inteira

Precedência:

Os operadores aritméticos se associam da esquerda para direita. Os operadores binários + e – possuem a mesma precedência, que é menor que a precedência de *, / e %, que por sua vez é menor que a do + e – unário.

4. Exemplos

1) Exercitar mais o conceito de algoritmo.

   (recomendação: início da aula, antes de começar o conteúdo previsto)

   - Algoritmo de Euclides, que calcula o máximo divisor comum (MDC) de dois números inteiros positivos. mdc (x, y) = mdc (y, x mod y); mdc (x, 0) = x.

     Entrada: 2 valores inteiros positivos m e n (m > n)

     Saída: máximo divisor comum de m e n.

     Passo 1: Adote x = m e y = n;
     Passo 2: Adote r = (resto de x dividido por y);
     Passo 3: Adote novos valores x = y e y = r;
     Passo 4: Se r é diferente de 0, volte ao passo 2; senão pare com a resposta x.

Figura 4: Algoritmo de Euclides (Miyazawa, 2001:3)

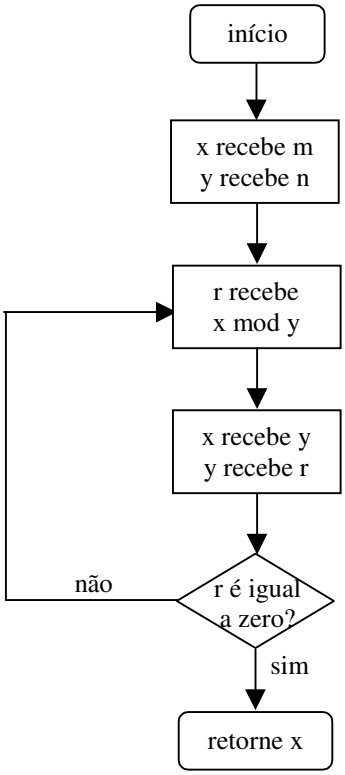

Figura 5: Representação do Algoritmo usando fluxograma

```c
#include <stdio.h>
main() {
    int x, y;
    scanf ("%d %d", &x, &y);
    do {
        r = x % y;
        x = y;
        y = r;
    }while(r != 0);
    printf ("%d\n", x);
}
```

Figura 6: Representação do Algoritmo na Linguagem C

◊ Mostrar funcionamento para exemplos de m e n (ex. m = 10, n = 3; m = 8 e n = 4)

- Algoritmo que determina se um número inteiro é primo.

2) Exercitar formatação da saída de dados com a função `printf`.

```c
//I
#include <stdio.h>
main() {
   printf ("oi mundo");
}
```

```c
//II
#include <stdio.h>
main() {
   printf ("oi mundo");
   printf ("louco");
}
```

```c
//III
#include <stdio.h>
main() {
  printf ("oi mundo ");
   printf ("louco");
}
```

```c
//IV
#include <stdio.h>
main() {
   printf ("oi mundo ");
   printf ("louco");
   printf ("oi mundo ");
   printf ("louco");
}
```

```c
//V
#include <stdio.h>
main() {
   printf ("oi mundo ");
   printf ("louco\n");
   printf ("oi mundo ");
   printf ("louco");
}
```

```c
//VI
#include <stdio.h>
main() {
   printf ("oi mundo louco\n");
   printf ("oi mundo louco");
}
```

3) Exercitar a precedência em operações aritméticas, com a função `printf` e o uso de caracteres de escape.

```c
//I
#include <stdio.h>
main() {
   printf ("4+5*6");
}
```

```c
//II
#include <stdio.h>
main() {
   printf ("%d", 4+5*6);
}
```

```c
//III
#include <stdio.h>
main() {
   printf ("%d", (4+5)*6);
}
```

```c
//III
#include <stdio.h>
main() {
   printf ("%x", (4+5)*6);
}
```

Referências

Hennessy, John L., Patterson, David A. "Abstrações e Tecnologia Computacionais". Em: *Organização e Projeto de Computadores – A Interface Hardware e Software*, 2ª edição, LTC editora., 2000, 551p.

Kernighan, Brian W.; Ritchie, Dennis M. *C A Linguagem de Programação Padrão ANSI*, 2ª edição, Rio de Janeiro, Campus, 1989, 289p.

Rubira, Cecília M. F. *Notas de Aula*, 1º semestre de 2000.

Schildt, H. *C Completo e Total*, 3ª edição, São Paulo, Makron Books, 827p. (trad. de Roberto Carlos Mayer, *C: The Complete Reference*, McGraw-Hill, 1995).

Tramontina, Gregório B. *Aula 1 – MC102 Turmas K e L*, 24 de Agosto 2004.

Trevisan, Norton. *Notas de Aula*, 2ª semestre de 2002, [online]: http://www.dcc.unicamp.br/~norton/paginas/mc102/mc102.html