Introdução à Programação

Capítulo IV

Mecanismos de Controlo

Engenharia Informática e de Sistemas

Instruções simples / blocos de instruções

Uma instrução em linguagem **C** seguida por um ponto e vírgula (**;**) é considerada uma instrução simples.

Um bloco de instruções tem a forma:

{ *instruções* **}**

Colocando chavetas a envolver um conjunto de instruções, força-se o compilador a tratá-las como se fossem uma única instrução.

Exemplo:

```
{
    num_linha=0;
    num_pagina++;
}
```

Apesar de cada uma das instruções internas ser terminada com um ponto e vírgula (**;**), no fim do bloco não é necessário colocar este caracter.

Instruções de seleção: if - else

A instrução **if-else**, é uma das instruções de controlo de fluxo da linguagem **C**. Esta permite indicar quais as circunstâncias em que determinada instrução (ou conjunto de instruções) deve ser executada.

Sintaxe:

```
if (condição)
        instrução1;
[else
        instrução2;]
```

Caso se pretenda realizar não apenas uma, mas um conjunto de instruções - quer no **if**, quer no **else** - estas devem ser escritas entre chavetas (**{ }**) de forma a serem consideradas como um único **bloco de instruções**.

Modo de funcionamento:

- A condição é avaliada;
- Se o resultado for *verdadeiro*, executa a *instrução1*;
- Se o resultado for *falso*, executa a *instrução2* (caso exista o **else**).

Instruções if-else encadeadas

Existem situações em que o teste de uma condição não é suficiente para tomar uma decisão. Pode ser necessário testar mais do que uma condição.

Sintaxe:

```
if(condição1)
        instrução1;
else if(condição2)
        instrução2;
else if(condição3)
        instrução3;
else
        instrução4;
```

Aconselhada

```
if(condição1)
    instrução1;
else
    if(condição2)
        instrução2;
    else
        if(condição3)
            instrução3;
        else
            instrução4;
```

As instruções **if-else** encadeadas permitem efetuar a seleção entre vários caminhos alternativos. As condições são avaliadas por ordem. Assim que surge uma verdadeira, a instrução associada é executada e o resto da estrutura é ultrapassado. A instrução associada ao último **else** (que pode não existir) é executada se todas as condições forem falsas.

Sempre que existam instruções **if-else** encadeadas, cada componente **else** pertence sempre ao último **if** (que ainda não tenha um **else** associado).

Exemplo 1a

```
/* Inverso de um número real */

#include <stdio.h>

void main()
{
    float x;

    printf("Introduza um valor: ");
    scanf("%f", &x);
    /* verificar se é diferente de zero */
    if(x)
       printf("%5.4f\n", 1/x);
}
```

Exemplo de execução:

```
Introduza um valor: 5
0.2000
Introduza um valor: 0
```

Exemplo 1b

```
/* Inverso de um número real  (Nova versão) */

#include <stdio.h>

void main()
{
    float x;

    printf("Introduza um valor: ");
    scanf("%f", &x);
    if(x)
       printf("%5.4f\n", 1/x);
    else
       printf("Valor nao e valido\n");
}
```

Exemplo de execução:

```
Introduza um valor: 5.0
0.2000
Introduza um valor: 0
Valor nao e valido
```

Exemplo 2a

```
/* Auxiliar de Cálculo */

#include <stdio.h>
void main()
{
    int res;

    printf("Qual o resultado de 12 + 6 ?”);
    scanf("%d", &res);
    if(res == 12 + 6)
        printf("Correto!\n");
    else
        printf("A resposta e %d\n", 12 +6);
        printf("O Resultado %d esta errado\n", res);
}
```

Exemplos de utilização:

```
Qual o resultado de 12 + 6 ?   23
A resposta e 18
O Resultado 23 esta errado

Qual o resultado de 12 + 6 ?   18
Correto!
O Resultado 18 esta errado
```

Exemplo 2b

```
/* Auxiliar de Cálculo Corrigido (Utilização de Blocos de Instruções) */

#include <stdio.h>
void main()
{
    int res;

    printf("Qual o resultado de 12 + 6 ?”);
    scanf("%d", &res);
    if(res == 12 +6)
        printf("Correto!\n");
    else
    {
        printf("A resposta e %d\n", 12 +6);
        printf("O Resultado %d esta errado\n", res);
    }
}
```

Exemplos de utilização:

```
Qual o resultado de 12 + 6 ?   23
A resposta e 18
O Resultado 23 esta errado

Qual o resultado de 12 + 6 ?   18
Correto!
```

Exemplo 3

```
/* Associar a parte else ao if correto: o objetivo deste
programa é fazer uma classificação parcial de um número inteiro.
No entanto, uma das versões tem um comportamento diferente do
esperado. */

/* Versão 1: Pretende classificar números positivos e negativos */

#include <stdio.h>
void main()
{
    int x;

    printf("Numero: ");
    scanf("%d", &x);
    if (x)
        if(x > 0)
            printf("Positivo");
        else
            printf("Negativo");
}
```

Exemplo de execução:

```
Numero: 34
Positivo
Numero: -4
Negativo
Numero: 0
```

Exemplo 4

```
/* Versão 2: Pretende classificar números positivos e nulos */

#include <stdio.h>
void main()
{
    int x;

    printf("Numero: ");
    scanf("%d", &x);
    if (x)
        if(x > 0)
            printf("Positivo");
    else
        printf("Zero");
}
```

Exemplo de execução:

```
Numero: 34
Positivo
Numero: -4
Zero
```

Este programa tem um comportamento incorreto! A versão corrigida será:

```
#include <stdio.h>
void main()
{
    int x;

    printf("Numero: ");
    scanf("%d", &x);
    if (x)
    {
        if(x > 0)
            printf("Positivo");
    }
    else
        printf("Zero");
}
```

Exemplo 5a

```
/* Máquina de Calcular Rudimentar – Versão if */

#include <stdio.h>

void main()
{
    char op;
    float num1, num2;

    printf("Expressao para calcular no formato\n(Num1 operacao Num2):”);
    scanf("%f %c %f", &num1, &op, &num2);

    if(op == '+')
        printf("Res: %.1f\n", num1 + num2);
    else if (op == '-')
        printf("Res: %.1f\n", num1 - num2);
    else if (op == '*')
        printf("Res: %.1f\n", num1 * num2);
    else if (op == '/' && num2 != 0.0)
        printf("Res: %.1f\n", num1 / num2);
    else
        printf("Operacao invalida\n");
}
```

Exemplos de execução:

```
Expressao para calcular no formato
(Num1 operacao Num2): 12 / 9
Res: 1.3

Expressao para calcular no formato
(Num1 operacao Num2): 0 # 2
Operacao invalida

Expressao para calcular no formato
(Num1 operacao Num2): 1.234 / 0
Operacao invalida
```

Instruções de seleção: switch

Esta instrução adapta-se particularmente à tomada de decisões em que o número de possibilidades é elevado, uma vez que reduz a complexidade de **if-else** consecutivos e encadeados.

Sintaxe:

```
switch (expressão){
    case constante1: instruções1
    ...
    case constanten: instruçõesn
   [default: instruções]
}
```

A *expressão* que surge entre parêntesis a seguir à palavra **switch**, pode ser uma expressão qualquer, cujo resultado seja um valor numérico dos tipos **int** ou **char**.

O mesmo acontece com as *constantes* relativas aos vários **case**, que também só podem ser dos tipos **int** ou **char**.

Modo de funcionamento:

- Os vários **case** e o **default** podem surgir por qualquer ordem.
- Se o valor da *expressão* for igual a alguma das *constantes (constante$_1$,....,constante$_n$)*, então são executadas as instruções associadas ao **case** correspondente, bem como todas as instruções de todos os **case** que se encontrem a seguir (e também as do **default**).
- Se o valor da *expressão* não for igual a nenhuma das referidas *constantes*, então são executadas as instruções do **default**. (Como se pode observar pela sintaxe, o **default** é opcional. Se este não existir nada é executado e a instrução **switch** termina)

A instrução **break**:

Esta instrução efetua uma transferência de controlo explícita, permitindo a saída imediata de um **switch**. (A execução do programa continua na instrução a seguir ao final do **switch**.)

Assim, colocando uma instrução **break** após cada grupo de instruções associadas a um **case**, evita-se que as dos **case** seguintes sejam também executadas. Como é óbvio, no último **case** ou no **default** (caso exista), não é necessário usar a referida instrução.

Exemplo 6a

```
/* Associar um Valor Qualitativo a uma Nota Quantitativa */

#include <stdio.h>

void main()
{
    int nota;

    printf("Qual a nota? ");
    scanf("%d", &nota);

    switch(nota)
    {
        case 1: printf("Mau!\n");
        case 2: printf("Mediocre!\n");
        case 3: printf("Suficiente!\n");
        case 4: printf("Bom!\n");
        case 5: printf("Excelente!\n");
        default:printf("Nota invalida!\n");
    }
}
```

Exemplos de Execução:

```
Qual a nota? 3
Suficiente!
Bom!
Excelente!
Nota invalida!

Qual a nota? 0
Nota invalida!
```

Exemplo 6b

```
/* Versão Corrigida */

#include <stdio.h>
void main()
{
    int nota;

    printf("Qual a nota? ");
    scanf("%d", &nota);
    switch(nota)
    {
         case 1: printf("Mau!\n");
                 break;
         case 2: printf("Mediocre!\n");
                 break;
         case 3: printf("Suficiente!\n");
                 break;
         case 4: printf("Bom!\n");
                 break;
         case 5: printf("Excelente!\n");
                 break;
         default:printf("Nota invalida\n");
                 break;
    }
}
```

Exemplo 7

```
/* Versão Simplificada */

/* O programa só indica se o aluno passou ou reprovou */

#include <stdio.h>
void main()
{
   int nota;

   printf("Qual a nota? ");
   scanf("%d", &nota);
   switch(nota)
   {
      default: printf("Nota invalida\n");
               break;
      case 1:
      case 2:printf("Reprovou\n"); break;
      case 3:
      case 4:
      case 5: printf("Passou\n"); break;
   }
}
```

Exemplo 5b

```
/* Máquina de Calcular Rudimentar – Versão switch */
#include <stdio.h>

void main()
{
    char op;
    float num1, num2;

    printf("Expressao para calcular no formato\n(Num1 operacao Num2):”);
    scanf("%f %c %f", &num1, &op, &num2);

    switch (op) {
      case '+':
        printf("Res: %.1f\n", num1 + num2);  break;
      case '-':
        printf("Res: %.1f\n", num1 - num2);  break;
      case '*':
        printf("Res: %.1f\n", num1 * num2);  break;
      case '/':
        if ( num2 ) {
          printf("Res: %.1f\n", num1 / num2);  break;
        }
      default:
        printf("Operacao invalida\n");  break;
    }
}
```

Instruções de repetição (ciclos): for

Sintaxe:

```
for(expr1;expr2;expr3)
    instrução;     /*corpo do ciclo */
```

expr1: expressão de inicialização que é executada uma única vez antes da entrada no ciclo;

expr2: expressão que controla o fim do ciclo. Enquanto a expressão for verdadeira (diferente de zero), a instrução (ou bloco de instruções) é executada. A avaliação da expressão é sempre efetuada antes da execução do corpo do ciclo;

expr3: operação que é executada no fim de cada iteração do ciclo.

Exemplo 1a:

```
#include <stdio.h>
void main()
{
  int i, numero;
    printf("Numero:");
    scanf("%d",&numero);
    for(i=1;i<=10;i++)
        printf("%2d * %2d = %2d\n",numero,i,numero*i);
}
```

Não é obrigatório existirem sempre as **expr1, expr2** e **expr3** ou mesmo o **corpo** do ciclo:

Exemplo 2:

```
#include <stdio.h>
void main()
{
  int numero;
    printf("Numero:");
    scanf("%d",&numero);
    for( ; numero; )
       printf("%d\n", numero--);
}
```

Cuidado com ciclos que não terminam (Exemplo 2 com numero negativo e Exemplo 3):

Exemplo 3:

```
#include <stdio.h>
void main()
{
  int i;

    for( i=10;i>0; i--)
       printf("%d\n", i++);
}
```

Exemplo 4a:

```
#include <stdio.h>

void main(void)
{
  int linha, valor, numero;

    printf("Diga o valor do lado:");
    scanf("%d", &numero);
    for ( linha=1; linha<=numero; linha++ )
    {
       for ( valor=linha; valor<=numero; valor++ )
         printf("%3d", valor);
       for ( valor=1; valor<linha; valor++ )
         printf("%3d", valor);
       printf("\n");
    }
}
```

Possível resultado de execução:

```
Diga o valor do lado:6
 1  2  3  4  5  6
 2  3  4  5  6  1
 3  4  5  6  1  2
 4  5  6  1  2  3
 5  6  1  2  3  4
 6  1  2  3  4  5
```

Exemplo 4b:

```
#include <stdio.h>
void main(void)
{
  int linha, valor, coluna, numero;

    printf("Diga o valor do lado:");
    scanf("%d", &numero);
    for ( linha=1; linha<=numero; linha++ )
    {
       for ( coluna=1, valor=linha; coluna<=numero; coluna++, valor++ )
       {
          printf("%3d", valor);
          if ( valor==numero )
             valor = 0;
       }
       printf("\n");
    }
}
```

Possível resultado de execução:

```
Diga o valor do lado:6
1  2  3  4  5  6
2  3  4  5  6  1
3  4  5  6  1  2
4  5  6  1  2  3
5  6  1  2  3  4
6  1  2  3  4  5
```

Instruções de repetição (ciclos): while

Sintaxe:

```
while(expressão)
      instrução;   /*corpo do ciclo */
```

A execução da instrução (ou bloco de instruções) é efetuada enquanto a avaliação da expressão produzir um valor diferente de zero. Esta avaliação é sempre feita antes da execução do corpo do ciclo. (Se logo à entrada do ciclo **while** a expressão tiver um valor igual a zero, o corpo do ciclo não é executado uma única vez.)

Exemplo 1b:

```
#include <stdio.h>
void main()
{
  int i = 1, numero;
    printf("Numero:");
    scanf("%d",&numero);
    while(i<=10)
    {
        printf("%2d * %2d = %2d\n",numero,i,numero*i);
        i++;
    }
}
```

Instruções de repetição (ciclos): do-while

Sintaxe:

```
do
    instrução;      /*corpo do ciclo */
while(expressão);
```

Neste ciclo a instrução (ou bloco de instruções) é executada e só depois é que a expressão é avaliada. Se o resultado da avaliação for diferente de zero, a execução do ciclo continua. Caso contrário, termina.

No ciclo **do-while** o corpo do ciclo é sempre executado pelo menos uma vez, ainda que a expressão seja falsa (igual a zero).

Exemplo 1c:

```
#include <stdio.h>
void main()
{
  int i = 1, numero;
    printf("Numero:");
    scanf("%d",&numero);

    do
    {
        printf("%2d * %2d = %2d\n",numero,i,numero*i);
        i++;
    }while(i<=10);

}
```

Exemplo 1d:

```
/*  Programa que escreve a tabuada de um número entre 1 e 10  */

#include <stdio.h>

void main()
{
  int n, i;

    do{
        printf("Numero: ");
        scanf("%d", &n);
    } while(n <= 0 || n > 10);

    for(i=1; i <= 10; i++)
        printf("%2d *%2d = %2d\n",n, i, n*i);
}
```

Exemplo de execução:

```
Numero: 21
Numero: -3
Numero: 4
 4 * 1 =  4
 4 * 2 =  8
 4 * 3 = 12
 4 * 4 = 16
 4 * 5 = 20
 4 * 6 = 24
 4 * 7 = 28
 4 * 8 = 32
 4 * 9 = 36
 4 *10 = 40
```

Exemplo 5

```
/*  Calcular a nota média de um conjunto de alunos */

#include <stdio.h>

void main()
{
  int nota, total, al_actual, n;

    printf("Numero de alunos: ");
    scanf("%d", &n);

    total = 0;
    al_actual = 1;
    while(al_actual <= n)
    {
        printf("Nota do aluno  %d: ", al_actual);
        scanf("%d", &nota);
        total +=nota;
        al_actual ++;
    }
    printf("\nA media e %d\n", total/n);
}
```

Exemplo de execução:

```
Numero de alunos: 4
Nota do aluno  1: 12
Nota do aluno  2: 15
Nota do aluno  3: 13
Nota do aluno  4: 18

A media e 14
```

Exemplo 6

```
/*  Calcular a soma de um conjunto de números inteiros  */
/*       (termina quando for introduzido o zero)        */

#include <stdio.h>

void main()
{
  int total, numero, conta;

    total = 0;
    conta = 0;
    printf("Introduza os numeros (0 para terminar):");
    scanf("%d", &numero);
    while(numero)      /*ou while(numero!=0)*/
    {
        total +=numero;
        conta ++;
        scanf("%d", &numero);
    }
    printf("\nForam introduzidos %d numeros e a soma e %d\n", conta,
          total);
}
```

Exemplo de execução:

```
Introduza os numeros (0 para terminar):
23  5  -2  9  10  0

Foram introduzidos 5 numeros e a soma e 45
```

Exemplo 7

```
/*   Programa para escrever um quadrado de asteriscos no monitor (a
dimensão do lado pode variar entre 2 e 10)  */

#include <stdio.h>

void main()
{
  int linha, coluna, i;

    do {
        printf("Dimensao do quadrado: ");
        scanf("%d", &i);
    } while(i < 2 || i > 10);

    for(linha = 1; linha <= i; linha ++)
    {
        for(coluna=1; coluna<=i; coluna++)
            printf("*");
        printf("\n");
    }
}
```

Exemplo de execução:

```
Dimensao do quadrado: 7
*******
*******
*******
*******
*******
*******
*******
```

Exemplo 8

```
/* Programa para ler 10 inteiros positivos e apresentar a soma dos
números impares e a soma dos números pares */

#include <stdio.h>

void main()
{
  int n, impares, pares, i;

    for(impares=0,pares=0,i=1; i<=10; i++)
    {
        do {
            printf("Numero %d: ", i);
            scanf("%d", &n);
        } while(n <= 0);
        if(n%2)
            impares += n;
        else
            pares += n;
    }
    printf("Soma dos pares: %d\n", pares);
    printf("Soma dos impares: %d\n", impares);

}
```

Exemplo de execução:

```
Numero 1: 2
Numero 2: 5
Numero 3: 7
Numero 4: -1
Numero 4: 3
Numero 5: 8
Numero 6: 1
Numero 7: 2
Numero 8: 5
Numero 9: 3
Numero 10: 2
Soma dos pares: 14
Soma dos impares: 24
```

Exemplo 9a

```
/* Cálculo do fatorial de um número inteiro positivo */

#include <stdio.h>

void main()
{
  int i, n, total;

    printf("Numero: ");
    scanf("%d", &n);

    total = 1;
    for (i = 1; i <=n; i++)
        total *= i;
    printf("O fatorial de %d e %d \n", n, total);

}
```

Exemplo 9b

```
/* Versão 2: Corpo do Ciclo for vazio (todas as instruções são efetuadas
nas expressões de controle) */

#include <stdio.h>

void main()
{
  int i, n, total;

    do {
        printf("Numero: ");
        scanf("%d", &n);
    } while (n<0);

    for(total = 1, i = 1; i <=n; total *= i++)
        ;
    printf("O fatorial de %d e %d \n", n, total);

}
```

Exemplo de execução:

```
Numero: -5
Numero: 4
O fatorial de 4 e 24



Numero: 7
O fatorial de 7 e 5040
```

As instruções *break* e *continue*

- **break**

A instrução **break** permite a saída imediata de um ciclo, ignorando as expressões que controlam a sua execução. Quando esta instrução é encontrada, a execução do programa continua na instrução imediatamente a seguir ao final do ciclo.

Pode utilizar-se esta instrução em qualquer dos ciclos apresentados anteriormente.

- **continue**

A instrução **continue** provoca o início da próxima iteração do ciclo onde está incluída. Todas as instruções entre o **continue** e a expressão que controla o ciclo são ignoradas.

Nos ciclos **while** e **do-while**, a expressão de controlo é imediatamente avaliada. No ciclo **for**, primeiro é executada a expressão designada por **expr3** (ver sintaxe), e a seguir é que é avaliada a expressão de controlo do ciclo.

Exemplo 10

```
/* Utilização da instrução break */

/* Cálculo do quadrado e da raiz quadrada de números positivos (termina
quando for introduzido um número negativo ou nulo) */

#include <stdio.h>
#include <math.h>

void main()
{
  float x;

    for(;;)
    {
        printf("Numero: ");
        scanf("%f", &x);
        if(x <= 0)
            break;
        printf("Quadrado: %4.2f\t", x*x);
        printf("Raiz Quadrada: %4.2f\n", sqrt(x));
    }
    printf("Programa terminado\n");
}
```

Exemplo de execução:

```
Numero: 23
Quadrado: 529.00   Raiz Quadrada: 4.80
Numero: 9
Quadrado: 81.00    Raiz Quadrada: 3.00
Numero: -3
Programa terminado
```

Exemplo 11a

```
/* Utilização da instrução continue */

/* O programa lê cinco números inteiros e apresenta a soma dos números
positivos */


/* Versão 1: Ciclo for */

#include <stdio.h>

void main()
{
  int n, soma, i;

    soma = 0;
    for(i=1; i<= 5; i++)
    {
        scanf("%d", &n);
        if(n <= 0)
            continue;
        soma += n;
    }
    printf("Soma:%d\n", soma);
}
```

Exemplo 11b

```
/* Utilização da instrução continue */

/* O programa lê cinco números inteiros e apresenta a soma dos números
positivos */


/* Versão 2: Ciclo while  */

#include <stdio.h>

void main()
{
  int n, soma, i;

    soma = 0;
    i = 1;
    while(i <= 5)
    {
        scanf("%d", &n);
        i++;
        if(n <= 0)
            continue;
        soma += n;
    }
    printf("Soma:%d\n", soma);
}
```

Exemplo 12

```
#include <stdio.h>     /*Programa que verifica se um número é ou não primo */
void main()
{
  int n, d;
  char c;
    while(1)
    {
        printf("Numero: ");
        scanf("%d", &n);
        if(n <= 0)
            continue;
        for(d=2; d <= n/2; d++)
            if(n % d == 0)
                break;
        if(d < n/2)
            printf("%d e divisivel por %d\n", n, d);
        else
            printf("O numero %d e primo\n", n);
        printf("Continuar? ");
        fflush(stdin);
        scanf("%c", &c);
        if (c != 's' && c != 'S')
            break;
    }
}
```

Exemplo de execução:

```
Numero: 11
O numero 11 e primo
Continuar? s
Numero: -5
Numero: 9
9 e divisivel por 3
Continuar? N
```