Linguagem C

Comandos de Controle de Programas

Prof. Daniel Ferreira

Instituto Federal do Ceará
Campus Maracanaú

Comandos de Seleção I

Verdadeiro e Falso na Linguagem C

Verdadeiro - expressão que retorne diferente de ZERO.
Falso - expressão que retorne ZERO.

Tipos de Comandos de Seleção:

if,
if-aninhados,
if-else-if,
Operador Ternário (?),
switch,
switch-aninhados.

Comando **if** I

- Sintaxe I:
  if (<expressão>)
    <comando>;
  else
    <comando>;
- Sintaxe II:
  if (<expressão>) {
    <comando>;
    <comando>;
  } else {
    <comando>;
    <comando>;
  }
- Semântica: O comando ou o bloco de comandos é executado se a <expressão> for verdadeira.

Comando **if** II

Exemplo:

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
  int numero;
  scanf("%d",&numero);

  if(numero % 2) printf("O número é ÍMPAR!!!! \n");

  if(! (numero % 2)){
     printf("O número é PAR!!!! \n");
  }

  return 0;
}
```

Comando **if-else-if** I

- Sintaxe:
  if (<expressão>)
    <comando>;
  else if (<expressão>)
    <comando>;
  else
    <comando>;
- Semântica: O comando ou o bloco de comandos é executado se a <expressão> for verdadeira.
- Caso nenhuma expressão seja avaliada como verdadeira o bloco de comandos pertencente ao **else** será executado.

Comando **if-else-if** II

Exemplo:

```
#include <stdio.h>

int main()
{
  int numero;
  scanf("%d",&numero);

  if(numero % 2){
     printf("O número é ÍMPAR!!!! \n");
  } else if(! (numero % 2)){
     printf("O número é PAR!!!! \n");
  } else {
     printf("Impossível imprimir isso aqui!!!! \n");
  }
  return 0;
}
```

Operador Ternário ? I

- Sintaxe:
  <tipo> <variavel> = (<expressao_logica> ? <expressao_1> : <expressao_1>);
- Semântica: A <expressao_logica> é avaliada. Se verdadeiro <expressao_1> é executada. Em caso contrário, <expressao_2> será executada.
- Vale ressaltar, que <expressao_1> e <expressao_2> devem ser do mesmo tipo que <tipo>.

Operador Ternário ? II

Exemplo:

```
#include <stdio.h>

int main()
{
  int numero;

  scanf("%d",&numero);

  printf("O número %d é  %s !!! \n", numero,

                (numero % 2 ? "impar" : "par"));

  return 0;

}
```

Seleção Múltipla **switch** I

- Sintaxe:

```
switch(<expressão>){
  case constante1:
    <sequencia_de_comandos>
  case constante1:
    <sequencia_de_comandos>
  case constante1:
    <sequencia_de_comandos>
  case constante1:
    <sequencia_de_comandos>
  default:
    printf(”Nenhuma opção selecionada!”);
}
```

- Semântica: O valor da <expressao> é testado contra os valores das constantes especificadas nos comandos **case**.

Seleção Múltipla **switch** II

- O **switch** só testa igualdade, diferentemente do **if**.
- Duas constates não podem ter valores idênticos.
- Se constantes **caracteres** são usadas, são convertidas em seus valores inteiros.

Seleção Múltipla **switch** III

Exemplo:

```
#include <stdio.h>
#include <stdlib.h>

void menu()
{
  char ch;
  printf("1. Checar ortografia \n");
  printf("2. Corrigir Erros de Ortografia \n");
  printf("3. Mostrar Erros de Ortografia \n");
  printf("Pressione qualquer Outra Tecla para Abandonar \n \n");
  printf("Entre com sua escolha: ");

  ch = getchar();

  switch(ch){
    case '1':
      checar_ortografia();
      break;
    case '2':
      corrigir_erros();
      break;
    case '3':
      mostrar_erros();
      break;
    default :
      printf("Nenhuma opção selecionada!");
  }

}
```

Comandos de Iteração I

Definição

Permitem que um conjunto de instruções sejam executadas até que ocorra uma certa condição.

Essa condição pode ser predefinida (for) ou com final aberto (while e do-while).

Laço **for**:

Sintaxe:

```
for(<inicialização>; <condição>; <incremento>)
   comando;
```

Comandos de Iteração II

Laço **for**:

<inicialização> - destinado a inicializar a variável de controle do laço.
<condição> - expressão para verificar necessidade de execução do comando.
<incremento> - comando para incrementar a variável de controle.

Semântica:
O comando pertencente ao laço for é executada repetidamente, desde que a condição seja VERDADEIRA. No momento, em que a condição assumir um valor FALSO o laço e terminado.

Comandos de Iteração III

Exemplo:

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
  int i;

  for(i = 0; i < 10; i++)
    printf("%d \n", i);

  return 0;
}
```

Comandos de Iteração IV

Laço Infinito

```
#include <stdio.h>

int main()
{
  for(;;)
    printf("Esse comando será
      executado repetidamente!!!");

  return 0;
}
```

Laço **while** I

Sintaxe

```
while(<condição>)
   <comando>;

while(<condição>){
   <comando>;
   <comando>;
}
```

Semântica

Semântica:
O laço while se repete sempre que a condição for VERDADEIRA.
O laço termina quando a a condição for FALSA.

Laço **while** II

Exemplo

```
#include <stdio.h>

int main()
{
  int i = 1;
  while(i <= 10){
    printf("%d \n", i);
    i++;
  }

  return 0;
}
```

Laço **do-while** I

Sintaxe

```
do{
   <comando>;
} while (<condição>);
```

Semântica

```
Semântica:
O laço do-while se repete sempre que a condição for
VERDADEIRA.
O laço termina quando a a condição for FALSA.
O(s) comando(s) do laço é executado no mínimo uma
 vez.
```

Laço **do-while** II

Exemplo

```
#include <stdio.h>

int main()
{
  int i = 1;
  while(i <= 10){
    printf("%d \n", i);
    i++;
  }

  return 0;
}
```

Comandos de desvio I

Sintaxe

Há 4 comandos de desvio incondicional: return, goto, break e continue.

Comando **return**

O comando return é usado para retornar (ou sair) de uma função.

Sintaxe:

```
return ; /*retorna lixo*/
return <expressão>; /*avalia a expressão e retorna o valor obtido*/
```

Comandos de desvio II

Comando **goto**

O comando goto é usado para desviar o fluxo de execução para um rótulo.

Sintaxe:

```
goto <rótulo>;
.
<rótulo>:
```

Comandos de desvio III

Exemplo: **goto**

```
x = 1;
loop1:
 x++;
if(x < 100)
  goto loop1;
```

Comandos de desvio IV

Uso do comando **break**

Usado no comando switch (como já vimos).
Usado em laços de iteração.

Comando **break** em laços

O comando break faz com que o processo de repetição dentro de um laço seja finalizado imediatamente. E o programa segue para o comando seguinte ao laço.

Comandos de desvio V

Exemplo: break

```
#include <stdio.h>

int main()
{
  int t = 1;
  while(i <= 10){
    if(i == 4)
      break;
    i++;
  }
  printf("%d \n", i);
  return 0;
}
```

Comandos de desvio VI

Comando **continue**

Ao invés de sair do laço, o fluxo segue para a próxima iteração.

Comandos de desvio VII

Exemplo **continue**

```
#include <stdio.h>

int main()
{
  int t = 1;
  while(i <= 10){
    if(i % 2 == 0)
      continue;
    printf("%d \n", i);
    i++;
  }
  return 0;
}
```

- Notas de aula: adaptado de prof. Ajalmar Rocha.
- Livro Base: C Completo e Total. Herbert Schildt. Capítulo 3.

OBRIGADO!!!