



COMANDOS CONDICIONAIS/ REPETIÇÃO

CAPÍTULO 16: LAÇOS CONDICIONAIS

`https://books.goalkicker.com/CBook/`



Comandos Condicionais em C

❑ **IF** => SE-ENTÃO

❑ **SWITCH** => ESCOLHA-CASO

Comando IF

O comando if representa uma tomada de decisão do tipo "SE isto ENTÃO aquilo". A sua forma geral é:

```
if (condição) {  
    declaração;  
}
```

A condição do comando if é uma expressão que será avaliada.

A declaração pode ser um bloco de código ou apenas um comando. É interessante notar que, no caso da declaração ser um bloco de código, não é necessário (e nem permitido) o uso do ; no final do bloco. Isto é uma regra geral para blocos de código.

Exemplo de IF

```
int target = 10;  
if (target == 10) {  
    printf("Target is equal to 10");  
}
```

IF-Else

Podemos pensar no comando else como sendo um complemento do comando if. O comando if completo tem a seguinte forma geral:

```
if (condição)  
    declaração_1;  
else  
    declaração_2;
```

Exemplo de IF-Else

```
int foo = 1;  
int bar = 2;  
  
if (foo < bar) {  
    printf("foo is smaller than bar.");  
} else {  
    printf("foo is greater than bar.");  
}
```

IFs Encadeados

A estrutura if-else-if é apenas uma extensão da estrutura if-else. Sua forma geral pode ser escrita como sendo:

```
if (condição_1)  
    declaração_1;  
else if (condição_2)  
    declaração_2;  
else if (condição_3)  
    declaração_3;  
...
```


IFs Encadeados

```
int foo = 1;
int bar = 2;

if (foo < bar) {
    printf("foo is smaller than bar.");
} else if (foo == bar) {
    printf("foo is equal to bar.");
} else {
    printf("foo is greater than bar.");
}
```

IF Aninhado

O if aninhado é simplesmente um if dentro da declaração de um outro if externo.

O único cuidado que devemos ter é o de saber exatamente a qual if um determinado else está ligado.

```
if (condição_1) {  
    declaração_1;  
} else {  
    if (condição_2) {  
        declaração_2;  
    } else {  
        if (condição_3) {  
            declaração_3;  
        }  
    }  
}
```

Exemplo de if aninhado

```
int peanuts_eaten = 22;
int peanuts_in_jar = 100;
int max_peanut_limit = 50;

if (peanuts_in_jar > 80) {
    if (peanuts_eaten < max_peanut_limit) {
        printf("Take as many peanuts as you want!\n");
    }
} else {
    if (peanuts_eaten > peanuts_in_jar) {
        printf("You can't have anymore peanuts!\n");
    }
    else {
        printf("Alright, just one more peanut.\n");
    }
}
```

Testes Compostos

```
int foo = 1;
int bar = 2;
int moo = 3;

if (foo < bar && moo > bar) {
    printf("foo is smaller than bar AND moo is larger than bar.");
}

if (foo < bar || moo > bar) {
    printf("foo is smaller than bar OR moo is larger than bar.");
}
```


Comando SWITCH

O comando switch é próprio para se testar uma variável em relação a diversos valores pré-estabelecidos. Sua forma geral é:

```
switch (variável) {  
    case constante_1:  
        declaração_1;  
        break;  
    case constante_2:  
        declaração_2;  
        break;  
    ...  
}
```

Exemplo de SWITCH

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

```
int main () {  
    int num;
```

```
    printf ("Digite um numero: ");  
    scanf ("%d",&num);
```

```
    switch (num) {
```

```
        case 9:
```

```
            printf ("\n\nO numero e igual a 9.\n");
```

```
            break;
```

```
        case 10:
```

```
            printf ("\n\nO numero e igual a 10.\n");
```

```
            break;
```

```
        case 11:
```

```
            printf ("\n\nO numero e igual a 11.\n");
```

```
            break;
```

```
        default:
```

```
            printf ("\n\nO numero nao e nem 9 nem 10 nem 11.\n");
```

```
    }
```

```
    return(0);
```

```
}
```

SWITCH x IF

- ❑ switch **não aceita expressões**
- ❑ switch aceita **apenas constantes**
- ❑ o switch testa a variável e executa a declaração cujo case corresponda ao valor atual da variável. A declaração **default** é opcional e será executada apenas se a variável, que está sendo testada, não for igual a nenhuma das constantes
- ❑ o comando **break**, faz com que o switch seja interrompido assim que uma das declarações seja executada. Mas ele não é essencial ao comando switch. Se após a execução da declaração não houver um break, o programa continuará executando

SWITCH X IF

Os dois trechos de código abaixo são equivalentes, mas o código com switch fica mais limpo para ser lido.

```
int a = 1;

switch (a) {
case 1:
    puts("a is 1");
    break;
case 2:
    puts("a is 2");
    break;
default:
    puts("a is neither 1 nor 2");
    break;
}
```

```
int a = 1;

if (a == 1) {
    puts("a is 1");
} else if (a == 2) {
    puts("a is 2");
} else {
    puts("a is neither 1 nor 2");
}
```


COMANDO SWITCH

Podemos agrupar vários valores para a mesma instrução ser executada.

```
int a = 1;

switch (a) {
case 1:
case 2:
    puts("a is 1 or 2");
case 3:
    puts("a is 1, 2 or 3");
    break;
default:
    puts("a is neither 1, 2 nor 3");
    break;
}
```

CAPÍTULO 15: LAÇOS DE REPETIÇÃO

`https://books.goalkicker.com/CBook/`



Comando while

- Forma geral:

```
while (condição) {  
    declaração;  
}
```

Exemplo utilizando o while

```
int n = 0;  
while (n < 10) {  
    n++;  
}
```


Comando break

É usado quando precisamos sair do laço de repetição de maneira abrupta no momento da execução do comando break.

```
int n = 0;
while (1) {
    n++;
    if (n == 10) {
        break;
    }
}
```

Comando continue

É usado quando precisamos pular para a próxima iteração do laço de repetição sem fazer o teste de parada.

```
int n = 0;
while (n < 10) {
    n++;

    /* check that n is odd */
    if (n % 2 == 1) {
        /* go back to the start of the while block */
        continue;
    }

    /* we reach this code only if n is even */
    printf("The number %d is even.\n", n);
}
```

Comando while-do

- Forma geral:

```
do  
{  
    declaração;  
} while (condição) ;
```

- O comando executa o teste apenas no final.
- Não esqueça o ponto e vírgula no final

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

```
int main ()
```

```
{
```

```
    int i;
```

```
    do
```

```
    {
```

```
        printf ("\n\nEscolha a fruta pelo numero:\n\n");
```

```
        printf ("\t(1)...Mamão\n");
```

```
        printf ("\t(2)...Abacaxi\n");
```

```
        printf ("\t(3)...Laranja\n\n");
```

```
        scanf ("%d", &i);
```

```
    } while ((i<1) || (i>3));
```

```
    switch (i)
```

```
    {
```

```
        case 1:
```

```
            printf ("\t\tVoce escolheu Mamão.\n");
```

```
            break;
```

```
        case 2:
```

```
            printf ("\t\tVoce escolheu Abacaxi.\n");
```

```
            break;
```

```
        case 3:
```

```
            printf ("\t\tVoce escolheu Laranja.\n");
```

```
            break;
```

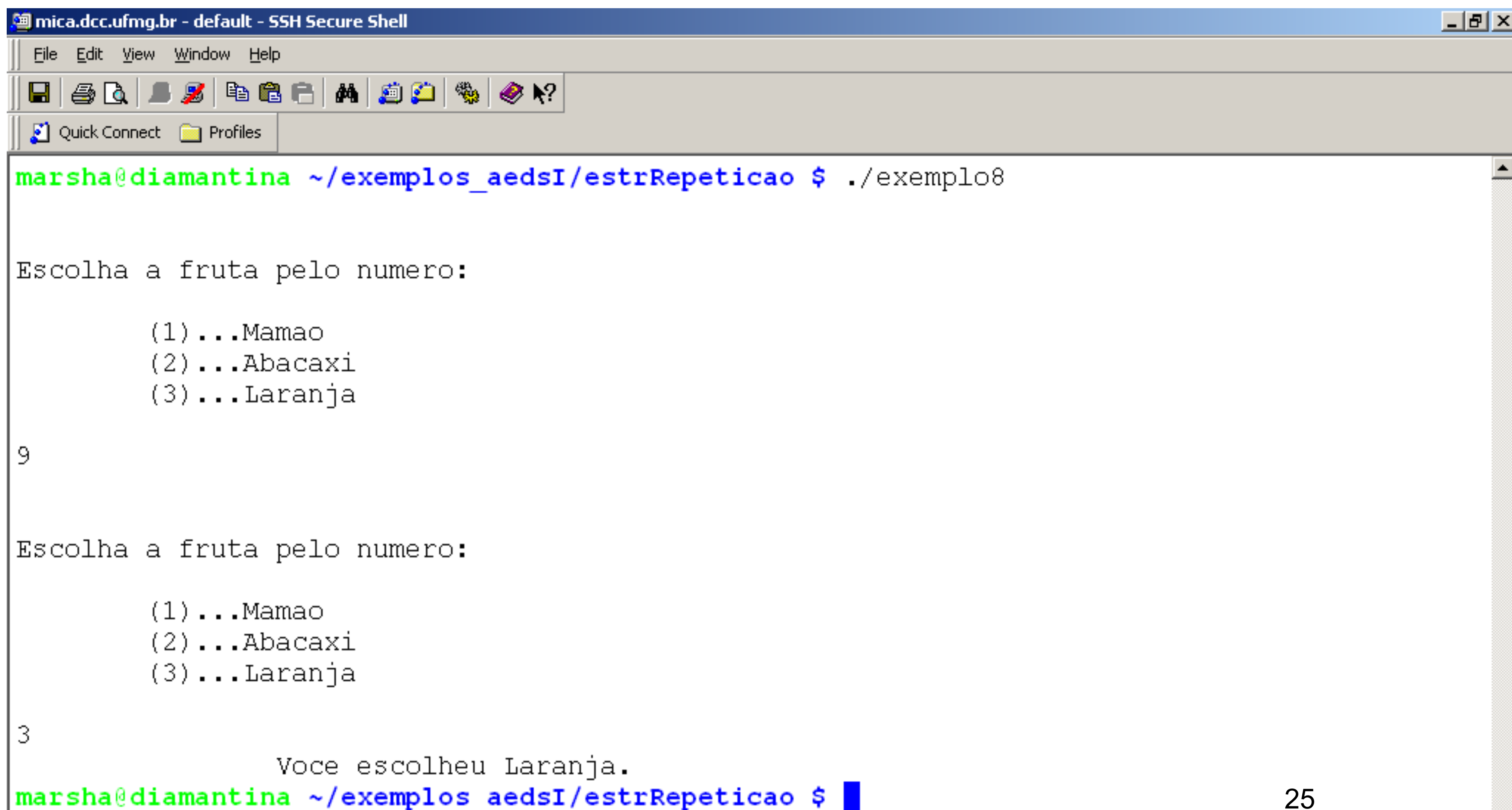
```
    }
```

```
    return(0);
```

```
}
```

Exemplo do-while – menu de frutas

Execução do programa com do-while



The screenshot shows an SSH terminal window titled "mica.dcc.ufmg.br - default - SSH Secure Shell". The terminal displays the execution of a program named "exemplo8" in the directory "~/exemplos_aedsI/estrRepeticao". The program prompts the user to "Escolha a fruta pelo numero:" and lists three options: (1) ...Mamão, (2) ...Abacaxi, and (3) ...Laranja. The user enters "9" in the first iteration and "3" in the second iteration. After the second iteration, the program outputs "Voce escolheu Laranja." and the terminal returns to the prompt "marsha@diamantina ~/exemplos_aedsI/estrRepeticao \$".

```
mica.dcc.ufmg.br - default - SSH Secure Shell
File Edit View Window Help
[Icons: Save, Print, Find, etc.]
Quick Connect Profiles

marsha@diamantina ~/exemplos_aedsI/estrRepeticao $ ./exemplo8

Escolha a fruta pelo numero:

    (1) ...Mamão
    (2) ...Abacaxi
    (3) ...Laranja

9

Escolha a fruta pelo numero:

    (1) ...Mamão
    (2) ...Abacaxi
    (3) ...Laranja

3

Voce escolheu Laranja.
marsha@diamantina ~/exemplos_aedsI/estrRepeticao $
```

WHILE X DO-WHILE

```
int num = 1;

while (num != 0)
{
    scanf("%d", &num);
}
```

```
int num, sum;
num = sum = 0;

do
{
    scanf("%d", &num);
    sum += num;
} while (sum < 50);
```

Comando for

- o comando **for** é usado para repetir um comando, ou bloco de comandos, diversas vezes, de maneira que se possa ter um bom controle sobre o laço.
- Sua forma geral é:

```
for (inicialização; condição; incremento) {  
    declaração;  
}
```

Funcionamento do for

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

Exemplo

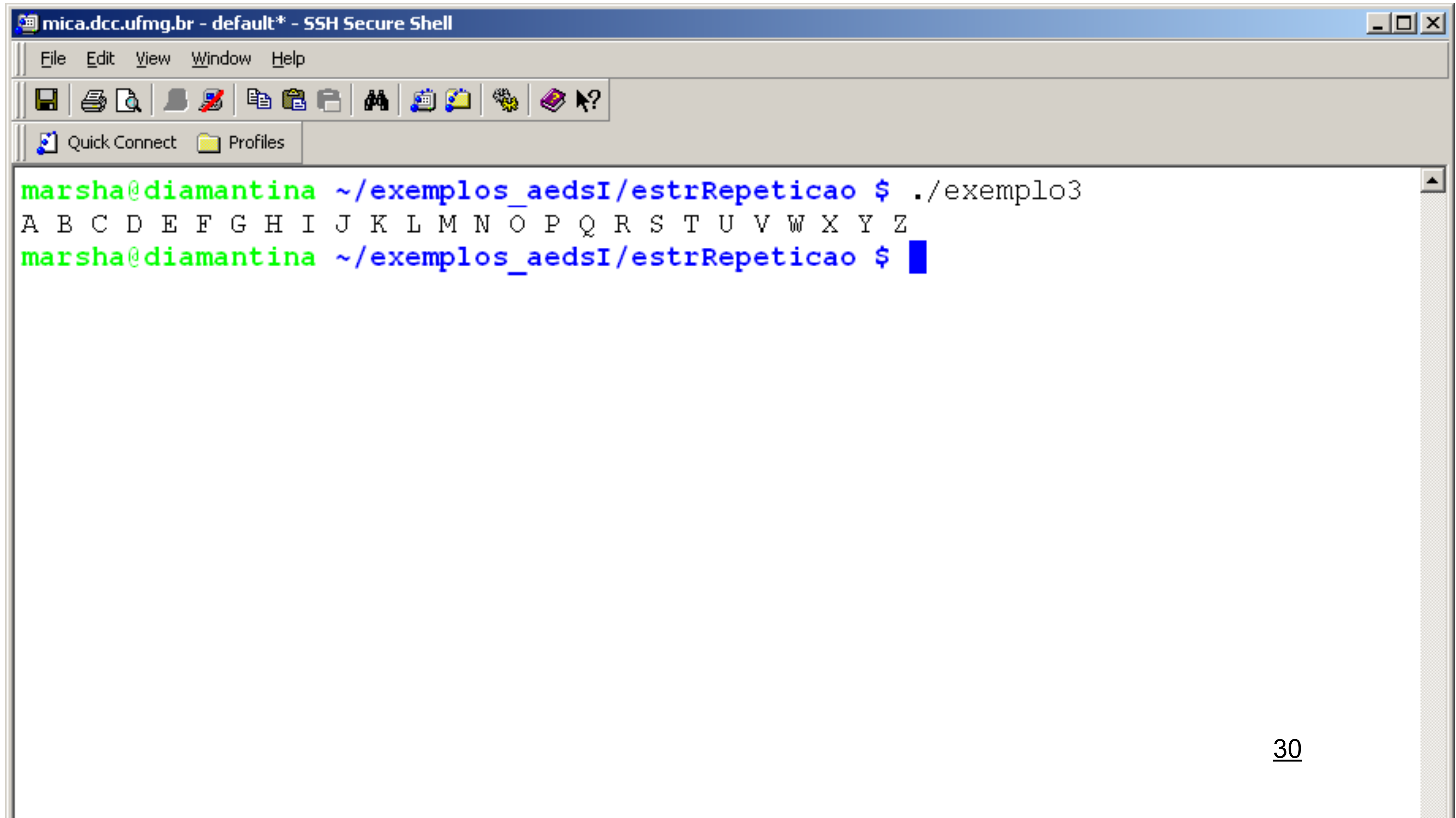
```
#include <stdio.h>

int main()
{
    char letra;

    for(letra = 'A' ; letra <= 'Z'; letra++)
        printf("%c ", letra);

    printf("\n");
    return(0);
}
```


Execução do Exemplo



The screenshot shows a terminal window titled "mica.dcc.ufmg.br - default* - SSH Secure Shell". The window has a menu bar with "File", "Edit", "View", "Window", and "Help". Below the menu bar is a toolbar with various icons for file operations and system functions. The terminal content shows a user named "marsha" at a host named "diamantina" in the directory "~/exemplos_aedsI/estrRepeticao". The user has executed the command "./exemplo3", which has produced the output "A B C D E F G H I J K L M N O P Q R S T U V W X Y Z". The prompt for the next command is shown as "marsha@diamantina ~/exemplos_aedsI/estrRepeticao \$" followed by a blue cursor.

```
mica.dcc.ufmg.br - default* - SSH Secure Shell
File Edit View Window Help
[Icons]
Quick Connect Profiles

marsha@diamantina ~/exemplos_aedsI/estrRepeticao $ ./exemplo3
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
marsha@diamantina ~/exemplos_aedsI/estrRepeticao $
```




OBRIGADO!

RAFAELVC2@GMAIL.COM

<https://books.goalkicker.com>