Lógica e Técnica de Programação – Introdução à Linguagem C - Resumo

por Salatiel S.

1: As funções de entrada e saída SCANF e PRINTF

```c
float Nota1, Nota2; //define as váriaveis com o tipo float
printf("Digite a primeira nota\n"); //mostra uma mensagem na tela
scanf("%f", &Nota1); //lê o valor int ou float que o usuário digitar...
printf("Digite a segunda nota\n");
scanf("%f", &Nota2); //o E comercial (&) salvará o valor na variável
printf("A média das duas notas é %f", (Nota1 + Nota2) / 2);
```

Objetivo:
Mostrar duas mensagens na tela pedindo por dois valores de tipo float e retornar a média desses dois valores.

Explicação:
O usuário define o valor das variáveis Nota1 e Nota2, a média é a soma dos dois valores dividida por 2, a soma é feita com o operador + e a divisão é feita com o operador /.

Por partes:

```c
float Nota1, Nota2;
```
Define as variáveis com o tipo "Real" (float)

```c
printf("Digite a primeira nota\n");
```
Escreve uma mensagem na tela e pula uma linha com o símbolo \n

```c
scanf("%f", &Nota1);
```

```c
printf("Digite a segunda nota\n");
```

```c
scanf("%f", &Nota2);
```
Recebe o valor float (%f) digitado pelo usuário e salva na variável escolhida pelo ponteiro (E comercial &)

```c
printf("A média das duas notas é %f", (Nota1 + Nota2) / 2);
```
Recebe o resultado float final da operação de soma e divisão

Soma os valores das duas variáveis e divide o resultado por 2

2: A função de repetição FOR

```c
for (int contador = 0; contador <= 5; contador++){
    printf(" %i \n ", contador); //o símbolo \n pula para uma nova linha
}
```

Objetivo:
Mostrar na tela os números do 0 até o 5 em diferentes linhas.

Explicação:
O valor inicial da variável int contador é 0, enquanto contador estiver menor ou igual à 5 (ou seja, até 5) ele vai se repetir e receber +1.

Por partes:

```c
for (int contador = 0; contador <= 5; contador++){
```
Define a variável com o tipo inteiro e o valor inicial

Verifica se contador é menor ou igual à 5

Soma contador com +1

```c
    printf(" %i \n ", contador);
```
Mostra o valor atual do contador, pulando uma linha sempre que é repetido

O próximo só será executado se o anterior for verdadeiro

```c
}
```

2.1: A função de repetição WHILE

```c
int contador = 0;
```
Define a variável com o tipo inteiro e o valor inicial

```c
while (contador <= 5){
```
Enquanto variável for menor ou igual à 5, repita.

```c
    printf(" %i \n ", contador);
    contador++; // adiciona +1 ao contador
```
Mostra o valor atual do contador, pulando uma linha sempre que é repetido

```c
}
```

3: As funções condicionais IF e ELSE

```c
//variáveis constantes, recebem um valor logo na definição do tipo (nesse caso int)
int idadeMinima = 18;
int idadeAtual = 15;
if (idadeAtual < idadeMinima){ //IF significa "se"
    printf(" Erro: a idade mínima é %i. ", idadeMinima);
} else { //ELSE significa "se não"
    printf(" Sucesso: idade %i suficiente. ", idadeAtual);
}
```

Objetivo:
Mostrar na tela uma mensagem que indica se a idade é suficiente ou não.

Explicação:
O valor da variável int idadeAtual é comparado ao valor da variável int idadeMinima, se for menor, uma mensagem de erro. Se não, uma mensagem de sucesso.

Por partes:

```c
//variáveis constantes, recebem um valor logo na definição do tipo (nesse caso int)
```
Linhas que se iniciam com // são comentários e são ignoradas pelo compilador

```c
int idadeMinima = 18;
int idadeAtual = 15;
```
Variáveis que recebem um valor assim que são definidas são chamadas de Constantes

```c
if (idadeAtual < idadeMinima){
```
Verifica se o valor de idadeAtual é menor que o valor de idadeMinima

Lembre-se de manter o seu código legível, utilize TAB para criar espaçamentos automáticos.

```c
    printf(" Erro: a idade mínima é %i. ", idadeMinima);
```
Se a condição é verdadeira, executa esta função.

```c
} else {
```

```c
    printf(" Sucesso: idade %i suficiente. ", idadeAtual);
}
```
Se a condição é falsa, executa esta função.

3.1: Os operadores lógicos AND, OR e NOT

```c
int a = 1;
int b = 2;
int c = 3;

if (a > b && a > c){
    //O símbolo && significa "AND", ou seja, executa apenas se A é maior do que B e C.
    printf("\n A é maior do que B E C");
};
if (a > b || a > c){
    //O símbolo || significa "OR", ou seja, executa se A for maior que B ou C.
    printf("\n A é maior do que B OU C");
};
if (!(a > b) && !(a > c)){
    //O símbolo ! significa "NOT", ou seja, executa se A não for maior que B e nem maior que C.
    printf("\n A NÃO é maior que B e nem C");
};
```

4: Espaçamentos e estrutura familiar

À medida em que seu código vai ficando grande e complexo, é necessário organizá-lo e deixá-lo mais legível para que você ou outros não tenham problemas para entender o objetivo do programa e corrigir possíveis erros.

Número de linhas

```c
01 int main(){
02     funcaoA1(){
03         funcaoA2(){
04             //filho A do filho A da função PAI
05         };
06     };
07
08     funcaoB1(){
09         funcaoB2(){
10             //filho A do filho B da função PAI
11         };
12     };
13 }
```

Função PAI principal

Função FILHO

Função NETO

Função FILHO

Função NETO

Para adicionar um espaço automático utilize a tecla TAB no inicio da linha.

As funções "FILHO" ficam 1 espaçamento à frente do próprio "PAI" Os irmãos possuem espaçamentos iguais.