

Parte 2 – Variáveis e Controles de Loops

Observe atentamente o programa a seguir pois o exemplo2.c envolve vários conceitos fundamentais, dentre eles loops (repetição), comentários, declaração de variáveis, inicialização de variáveis com valores primitivos e formatação de dados.

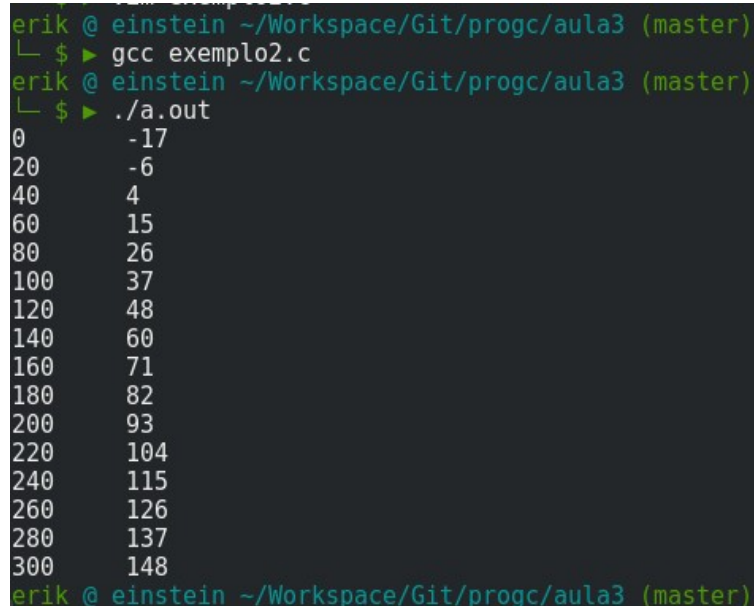
O programa a seguir usa a formula abaixo para conversão de medidas em Fahrenheit-Celsius. Após editar o código será apresentada a seguinte saída (vide Figura 1 abaixo)

$$^{\circ}C = (5/9)(^{\circ}F - 32)$$

```
#include <stdio.h>
/*
    Imprime os valores de Fahrenheit-Celsius de 0, 20,..., 300
*/
main(){
    int fahr, celsius;
    int lower, upper, step;

    lower = 0;
    upper = 300;
    step = 20;

    fahr = lower;
    while (fahr <= upper) {
        celsius = 5 * (fahr-32) / 9;
        printf("%d\t%d\n", fahr, celsius);
        fahr = fahr + step;
    }
}
```



```
erik @ einstein ~/Workspace/Git/progc/aula3 (master)
└─ $ ▶ gcc exemplo2.c
erik @ einstein ~/Workspace/Git/progc/aula3 (master)
└─ $ ▶ ./a.out
0      -17
20     -6
40      4
60     15
80     26
100    37
120    48
140    60
160    71
180    82
200    93
220   104
240   115
260   126
280   137
300   148
erik @ einstein ~/Workspace/Git/progc/aula3 (master)
```

Figura 1 – Compilando e executando o programa exemplo2.c

A partir do código acima pode-se observar os seguintes pontos de interesse:

- Comentários iniciados como

```
/*  
    Imprime os valores de Fahrenheit-Celsius de 0, 20,..., 300  
*/
```

Nesse caso, as instruções dentro desse bloco de código serão desconsideradas pelo compilador. Portanto. Não serão interpretadas.

- Declaração de variáveis

```
int fahr, celsius;  
int lower, upper, step;
```

Todas as variáveis devem ser declaradas em C antes de poder usa-las. Nesse caso são declaradas 4 variáveis do tipo Inteiros. Cada variável do tipo int possui um tamanho de que depende da arquitetura do hardware. Nesse caso, se a CPU é de 8, 16, 32 ou 64 bits. Por exemplo, se uma variável está sendo declarada em uma máquina com 16-bits de CPU então o tamanho (faixa de inteiros) vão de **-32768 até +32767** (pois é limitado por 2^{16}).

Tipo Primitivo	Descrição
char	Um caracter simple – um byte (8 bits)
short	Um inteiro curto (metade de um Inteiro) – short integer
int	Um inteiro
long	Um inteiro longo
float	Um número real de precisão simples
double	Um número real de precisão dupla

Após a criação das variáveis o programa acima inicializa as variáveis com valores para serem usadas no loop em seguida.

```
lower = 0;  
upper = 300;  
step = 20;  
fahr = lower;
```

Dessa forma é possível realizar a computação com o Loop **while**

```
while (fahr <= upper){  
    ...  
}
```

O loop while funciona da seguinte forma: A condição em parenteses é testada. Se ela é verdadeira (**fahr** é menor ou igual que **upper**) então o corpo do loop é executado. (nesse caso se houver abre e fecha chaves). No término, a condição é re-testada para verificar se ainda é verdadeira. Se for o loop é executado novamente. Em caso contrário, se a condição falhar o loop não é executado.

A parte central da computação do loop – while é realizada na seguinte equação (com a atribuição)

```
celsius = 5 * (fahr-32) / 9;
```

Desa forma, ao ser computado o valor da direita para a esquerda o resultado é acumulado na variável celsius.

Outro detalhe apresentado no exemplo, é o uso do comando **printf()**

```
printf("%d\t%d\n", fahr, celsius);
```

Observe atentamente que os formatadores de inteiros **%d** que estão relacionados com cada variável a sua direita.

Atividades

1) Modifique o programa anterior para formatar a saída com alinhamento justificado à direita e não a esquerda como está. Para isso, use o seguinte comando.

```
printf("%3d %6d\n", fahr, celsius);
```

```
      0      -17
     20      -6
     40       4
     60      15
     80      26
    100      37
    ...
```

2) Modifique o programa para apresentar um resultado mais preciso para o cálculo da temperatura. Nesse caso, observe que a conversão 0 F é -17.8 C e não -17 como apresentado. Isso acontece por conta do truncamento do resultado. Para isso use **float** no lugar de **int**

```
float fahr, celsius;
```

(*) será preciso ajustar a formatação do comando **printf()**

```
printf("%3.0f %6.1f\n", fahr, celsius);
```

%d	print as decimal integer
%6d	print as decimal integer, at least 6 characters wide
%f	print as floating point
%6f	print as floating point, at least 6 characters wide
%.2f	print as floating point, 2 characters after decimal point
%6.2f	print as floating point, at least 6 wide and 2 after decimal point

3) Modifique o programa para imprimir um cabeçalho descritivo na tabela acima.