

PROJETO INTEGRADOR – CONVERSOR DE BASES EM C

Ana Paula Antunes Araujo

Curso Tecnólogo em Análise e desenvolvimento de Sistema – Universidade do Vale do Itajaí (UNIVALI) – Campus de Itajaí
88302-901– Itajaí – SC– Brasil

Anapaula.antunesaraujo@outlook.com

Abstract. *This meta-article presents the concepts and describes how these concepts were put into practice when solving the proposed problem. The problem situation that must be solved is to program a console application that performs the conversion between numeric bases, in C language.*

Resumo. *Este meta-artigo traz os conceitos e descreve como estes conceitos foram colocados em prática ao resolver o problema proposto. A situação problema que deve ser resolvida é programar uma aplicação console que realize a conversão entre bases numéricas, em linguagem C.*

1. Introdução

Este trabalho foi elaborado com o intuito de praticarmos o que vimos em aula.

Utilizamos alguns dos conceitos que nos foram apresentados como lógica de programação, algoritmo, sistemas de numeração, conversão entre as bases numéricas e sintaxe da linguagem C.

Este trabalho tem como intuito instigar o(a) acadêmico(a) a buscar soluções ao problema proposto, que foi: Conseguir desenvolver utilizando a linguagem C uma aplicação console intuitiva, que o usuário consiga compreender o que deve fazer e conseguir interpretar o que as ações dele podem gerar dentro da aplicação. Essa aplicação deve conseguir realizar conversões entre bases numéricas de uma forma que seja simples e fácil para o usuário.

2. Lógica de programação e algoritmos

A lógica de programação nada mais é do que a capacidade que todo programador precisa ter para conseguir resolver os problemas que aparecem no dia-a-dia. A capacidade de dividir o problema em partes menores é uma etapa essencial da lógica de programação e precisa ser levada em consideração quando nos deparamos com qualquer problema que precisa ser resolvido. É nesse ponto que entra o conceito de algoritmo, que geralmente é descrito como uma sequência lógica de ações capaz de resolver um problema (GASPAROTTO, 2017).

Ressaltando, no entanto, que o conceito de algoritmo vai muito além da programação. Qualquer resolução de problema pode ser organizada em partes como uma receita de bolo, esse é um exemplo simples de algoritmo (GASPAROTTO, 2017).

Com isso entendemos que algoritmos é muito utilizado na programação, porém pode ser aplicado em outras tarefas do dia a dia. É na verdade uma forma de pensar e organizar o que precisa ser feito.

Na programação utilizamos o pensamento lógico ao desenhar e prever cada ação e reação que a aplicação que esta sendo desenvolvida precisa realizar, o que precisa ser executado primeiro, o que precisa ser executado em seguida e assim sucessivamente.

3. Sistemas de numeração

Sistemas de numeração são conjuntos de símbolos e regras para representar o número de elementos de conjuntos quaisquer (GARCIA, 2011).

O número de símbolos utilizados na representação dos números é chamado de base do sistema de numeração (GARCIA, 2011).

As bases que serão vistas neste projeto são:

- Decimal
- Binária
- Octal
- Hexadecimal

3.1. Sistema decimal

Sistema posicional de base 10 em que os seus algarismos indo-arábicos (0,1,2,3,4,5,6,7,8,9) servem para contar unidades, dezenas, centenas, entre outros, da direta para a esquerda (CÓRDOVA JUNIOR, 2018).

A principal regra para escrever quantidades utilizando o sistema decimal é que todo algarismo escrito à esquerda de uma posição possui um peso 10 vezes maior do que se fosse escrito nessa posição. Por isso, o sistema numérico é conhecido como um sistema posicional (GARCIA, 2011).

3.2. Sistema binário

O sistema binário é de base 2 e utiliza os símbolos '0' e '1', sendo que cada um chamado de 'bit'. Uma sequência composta por 8 bits forma um 'byte', esse é o sistema numérico que forma a base da ciência da computação (GARCIA, 2011).

3.3. Sistema octal

O sistema octal, também chamado de sistema de base 8, possui oito algarismos (0,1,2,3,4,5,6,7), possui relação direta com o sistema binário. Foi muito utilizado na computação, na programação em linguagem de máquina, como uma alternativa mais compacta do sistema de base 2 (CÓRDOVA JUNIOR, 2018).

3.4. Sistema hexadecimal

Hexadecimal é um sistema de numeração muito utilizado na programação de microprocessadores, especialmente em equipamentos de estudos e sistemas de desenvolvimento. Utiliza os símbolos do sistema decimal mais as letras A, B, C, D, E e

F, sendo que a letra A equivale a 10, a letra B a 11, a letra C a 12, a letra D a 13, a letra E a 14 e a letra F a 15 (CÓRDOVA JUNIOR, 2018).

4. Conversão de bases

A conversão de bases nada mais é do que transformar uma representação numérica de um sistema de numeração para outro. Exemplo: O número decimal 50 em binário é 00110010.

Neste capítulo vamos ver de forma breve como realizar estas conversões.

4.1. Conversão de decimal para binário

A conversão de números decimais para números binários é realizada através de divisões consecutivas. Dividimos o número da base decimal por 2 até que não seja mais divisível, ao final, resultado da última divisão ajuntado dos restos das demais divisões "de baixo para cima" formam o número binário (ALVES, 2018).

Exemplo:

The diagram illustrates the conversion of the decimal number 29 to binary. It shows a series of divisions by 2, with the quotient and remainder at each step. The remainders are collected from bottom to top to form the binary number.

29		2
1		14
0		7
1		3
1		1

29₁₀ = 11101₂

Figura 1. Exemplo de conversão de decimal para binário

Fonte: CÓRDOVA JUNIOR, 2018

4.2. Conversão de decimal para octal

A conversão numérica de Decimal para Octal é muito similar a conversão de decimal para binário, a diferença é que para converter para octal é necessário dividir por 8 (ALVES, 2018).

Exemplo:

$$\begin{array}{r} 140 \div 8 = 17 \text{ R } 4 \\ 17 \div 8 = 2 \text{ R } 1 \end{array}$$
$$140_{10} = 214_8$$

Figura 2. Exemplo de conversão de decimal para octal

Fonte: CORDOVA JUNIOR, 2018

4.3. Conversão de decimal para hexadecimal

A conversão de números decimais para a base hexadecimal também é muito similar a conversão de decimal para binário ou octal, porém dividimos por 16. (ALVES, 2018).

Também não podemos esquecer os algarismos do sistema hexadecimal para os números 10, 11, 12, 13, 14, 15, pois no lugar deles devem ser utilizadas as letras A, B, C, D, E e F (ALVES, 2018).

Exemplo:

$$\begin{array}{r} 2834 \div 16 = 177 \text{ R } 2 \\ 177 \div 16 = 11 \text{ R } 1 \end{array}$$

Resultado: B12₁₆

Figura 3. Exemplo de conversão de decimal para hexadecimal

Fonte: ALVES, 2018

4.4. Conversão de octal para decimal

A conversão de números da base octal para a base decimal é muito semelhante a conversão de binário para decimal, porém utilizamos 8 no lugar do número 2. (ALVES, 2018).

Exemplo:

$$\begin{array}{cccc} 1 & 0 & 5 & 6_8 \\ (8^3 \times 1) + (8^2 \times 0) + (8^1 \times 5) + (8^0 \times 6) = 558_{10} \end{array}$$

Figura 4. Exemplo de conversão de octal para decimal

Fonte: CORDOVA JUNIOR, 2018

4.5. Conversão de hexadecimal para decimal

O processo para converter um número de hexadecimal para decimal é similar aos processos de conversão de binário para decimal e de Octal para decimal, porém ao invés de 2 ou 8 utilizamos o 16 e é necessário substituir as letras A, B, C, D, E e F por 10, 11, 12, 13, 14 e 15 (ALVES, 2018).

Exemplo:

$$\begin{array}{cccc} 1 & 0 & A & 6_{16} \\ (16^3 \times 1) + (16^2 \times 0) + (16^1 \times 10) + (16^0 \times 6) = 4262_{10} \end{array}$$

Figura 5. Exemplo de conversão de hexadecimal para decimal

Fonte: CORDOVA JUNIOR, 2018

4.6. Conversão de binário para decimal

A conversão de números binários para números decimais é realizada através de uma somatória dos algarismos binários da direita pra a esquerda onde cada termo da somatória é multiplicado por 2 elevado a um número sequencial iniciado em 0 e vai até 7 (ALVES, 2018).

Exemplo:

$$\begin{array}{ccccccc} 1 & 0 & 0 & 1 & 0 & 1 & 1_2 \\ (2^6 \times 1) + (2^5 \times 0) + (2^4 \times 0) + (2^3 \times 1) + (2^2 \times 0) + (2^1 \times 1) + (2^0 \times 1) = 75_{10} \end{array}$$

Figura 6. Exemplo de conversão de binário para decimal

Fonte: CORDOVA JUNIOR, 2018

4.7. Conversão de binário para octal

A conversão de números da base binária para a base octal, é bem parecida com a conversão binário-decimal, mas antes de iniciar a conversão é preciso separar os dígitos binários de 3 em 3 da direita para a esquerda e converter separadamente cada grupo de 3 dígitos, depois concatenar os resultados (ALVES, 2018).

Exemplo:

$11010111_2 = 327_8$		
11	010	111
011	010	111
421	421	421
$0+2+1$	$0+2+0$	$4+2+1$
3	2	7

↓
Procedimento

Figura 7. Exemplo de conversão de binário para octal

Fonte: CORDOVA JUNIOR, 2018

4.8. Conversão de binário para hexadecimal

A conversão de números da base binária para a base hexadecimal é quase idêntica à conversão de binário para octal, só que agora separamos os dígitos binários de 4 em 4 da direita para a esquerda e antes de unir os dígitos ao final, trocamos os números 10, 11, 12, 13, 14 e 15 por A, B, C, D, E e F (ALVES, 2018).

Exemplo

$1011010111_2 = 2D7_{16}$		
10	1101	0111
0010	1101	0111
8421	8421	8421
$0+0+2+0$	$8+4+0+1$	$0+4+2+1$
2	D	7

↓
Procedimento

Figura 8. Exemplo de conversão de binário para hexadecimal

Fonte: CORDOVA JUNIOR, 2018

5. Linguagem C

A linguagem C é uma LP bastante popular, criada por volta de 1970, por Dennis Ritchie (SANTOS, 2018).

Algumas das características da linguagem C é a portabilidade, poder e variedade dos operadores, sintaxe elegante, estruturada e flexível, acesso facilitado à memória e a todo o hardware, uso de procedimentos e funções para desenvolver sistemas desacoplados (SANTOS, 2018).

Usar a linguagem C é muito simples, é possível escrever um código fonte em C até mesmo em um bloco de notas. Para isso podemos escrever um algoritmo, depois traduzimos o mesmo para a sintaxe em C e salvamos o arquivo com um nome + .c, exemplo programa.c (SANTOS, 2018).

Com o código escrito precisamos de um compilador para gerar o programa e por fim precisamos apenas executar o sistema. (SANTOS, 2018).

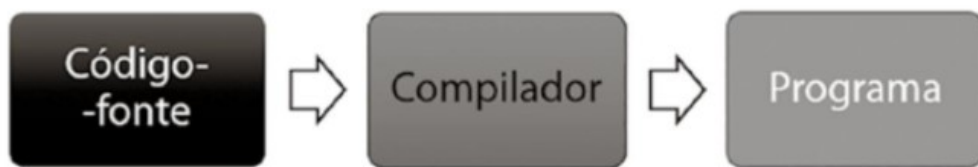


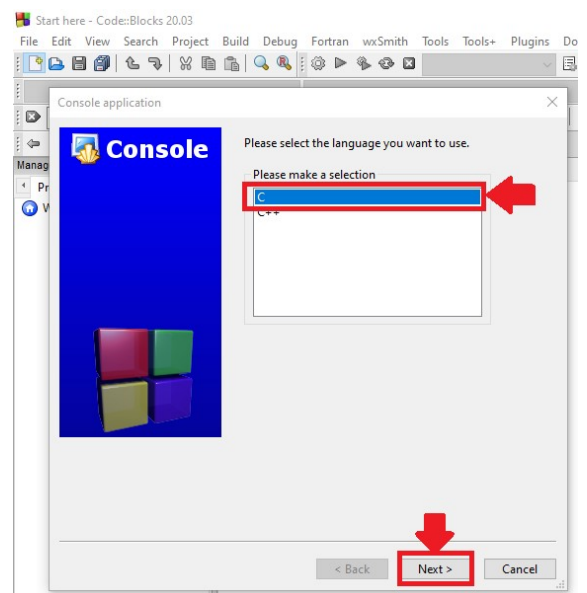
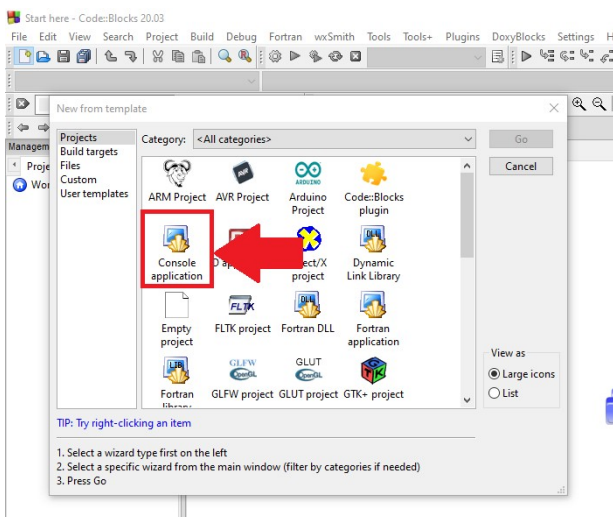
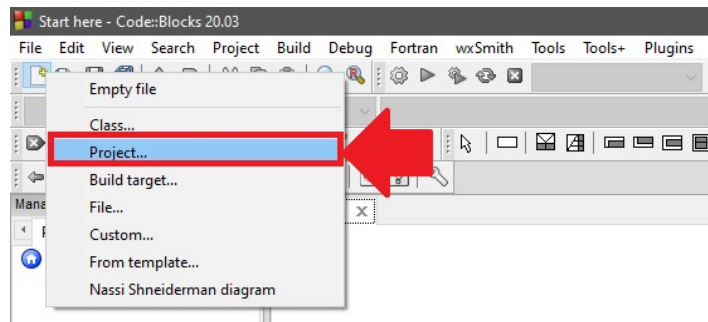
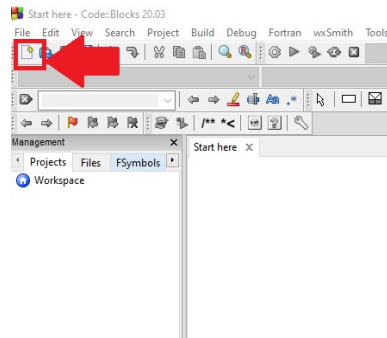
Figura 9. Como um programa compilado é criado?

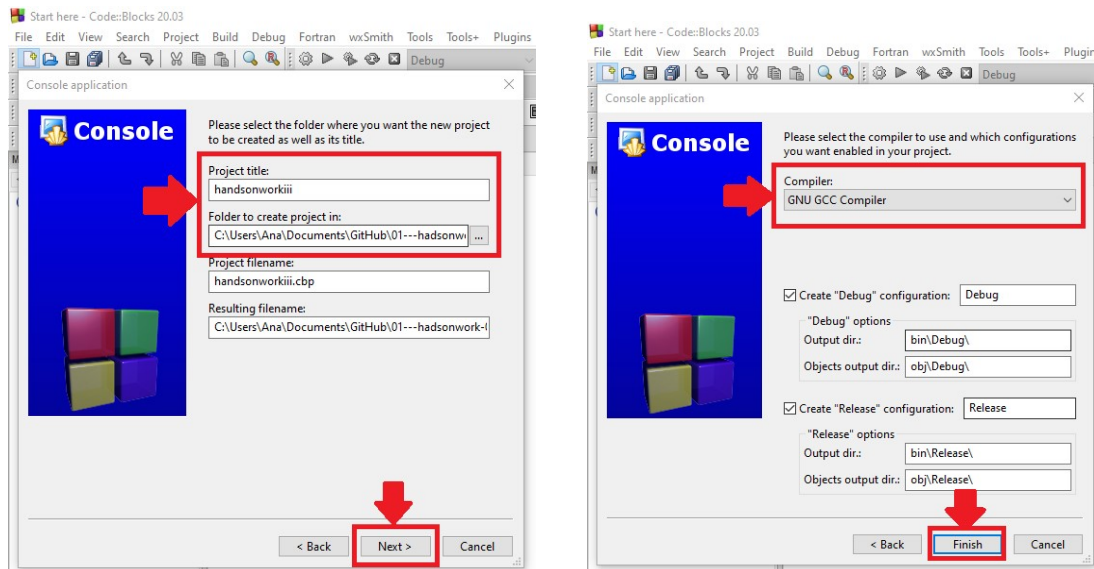
Fonte: SANTOS, 2018

6. Projeto prático

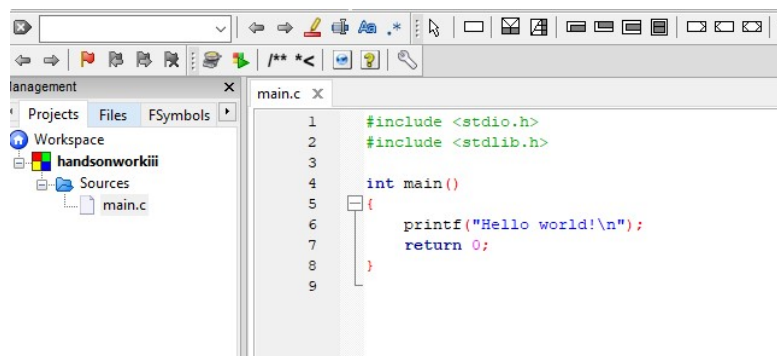
Para a aplicação dos conteúdos vistos em sala foi utilizado o sistema Code::Block e o compilador no Code::Block, o GNU GCC Compiler.

Foi criado um projeto de nome “handsonworkiii” como mostra as imagens a seguir:





Por padrão o Code::Blocks cria um projeto que imprime na tela a frase “Hello world!”.



Apaguei este código e iniciei a digitação do código para o conversor de bases.

A princípio foram declaradas as Variáveis que seriam utilizadas. Também foi montado o menu da aplicação utilizando o comando *printf* e foi configurado para pegar o que for digitado e salvar na variável *opcao* utilizando o comando *scanf*.

```

//VARIÁVEIS UTILIZADAS
char opcao = ' ';
int valor_octal;
long long valor_binario;
int valor_decimal, cociente;
int sequencial, valor_temp, tamanho_string;
char valor_hexa[100];

//ENQUANTO FOR A OPCAO 1 EXECUTA NOVAMENTE, SE FOR DIFERENTE DE 1 SISTEMA É FINALIZADO
while (opcao2 = 1) {
    //MOSTRA O MENU PRINCIPAL
    printf("\n-----\n");
    printf("##BEM VINDO AO CONVERSOR DE BASES!##\n");
    printf("\nDigite a Opcao de conversao desejada\n");
    printf("\n1 - Binario para Decimal\n");
    printf("\n2 - Decimal para Binario\n");
    printf("\n3 - Decimal para Octal\n");
    printf("\n4 - Octal para Decimal\n");
    printf("\n0 - Sair\n");
    printf("\nResposta: ");
    scanf("%c", &opcao);
    getchar();
}

```

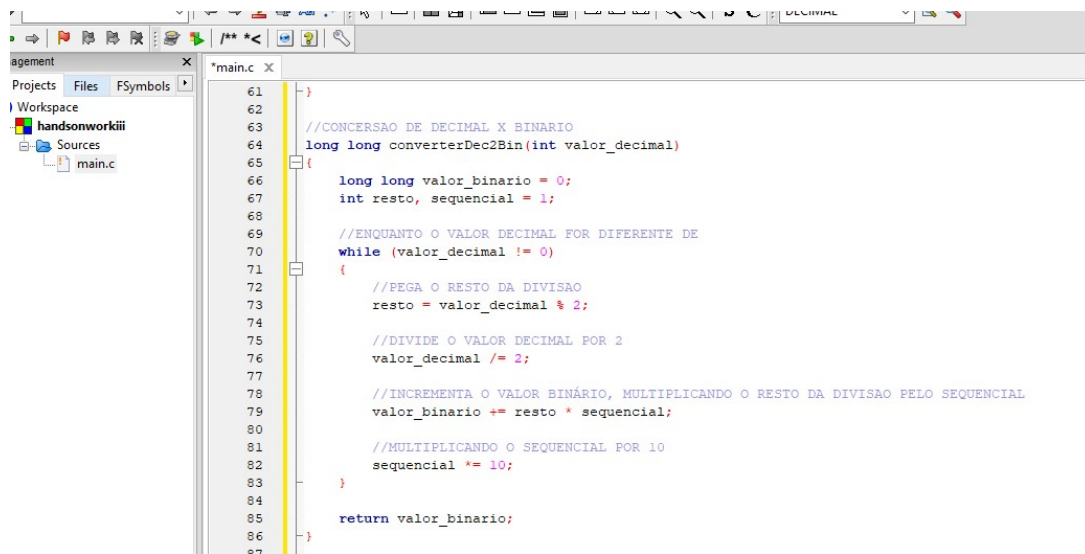
Feito isso foi defini que utilizaria funções e que cada função realizaria um calculo de conversão.

Para criar essas funções foi feita a chamada das mesmas no início do código, fora da função main().

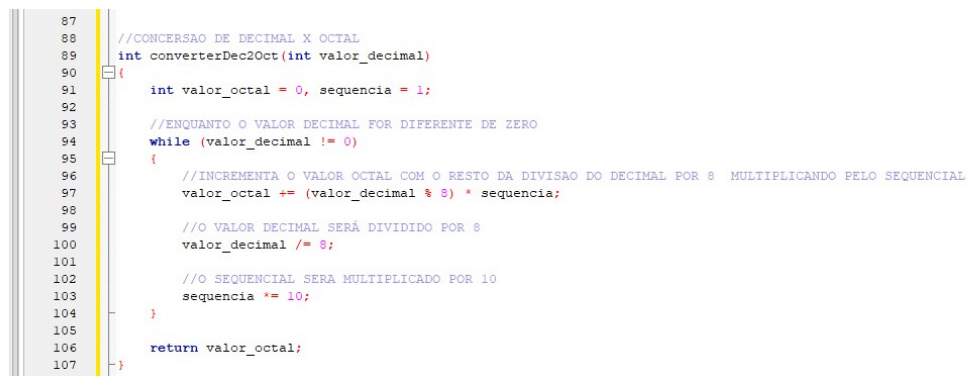
```
*main.c X
1  #include <stdio.h>
2  #include <math.h>
3  #include <string.h>
4
5  //CHAMADA DAS FUNÇÕES UTILIZADAS
6  int converterBin2Dec(long long valor_binario);
7  long long converterDec2Bin(int valor_decimal);
8  int converterDec2Oct(int valor_decimal);
9  long long converterOct2Dec(int valor_octal);
10
11 int main()
12 {
13
14     //VARIÁVEIS UTILIZADAS
15     char opcao = ' ';
16     int valor_octal;
17     long long valor_binario;
18     int valor_decimal, cociente;
19     int sequencial, valor_temp, tamanho_string;
20     char valor_hexa[100];
21
22     //MOSTA O MENU PRINCIPAL
23     printf("\n-----\n");
24     printf("##BEM VINDO AO CONVERSOR DE BASES!#\n");
25     printf("\nDigite a Opção de conversão desejada\n");
26     printf("\n1 - Binário para Decimal\n");
27     printf("\n2 - Decimal para Binário\n");
28     printf("\n3 - Decimal para Octal\n");
29     printf("\n4 - Octal para Decimal\n");
30     printf("\n0 - Sair\n");
31     printf("\nResposta: ");
32     scanf("%c", &opcao);
33     getchar();
34
```

Depois de fazer a chamada das funções elas foram criadas dentro da função main() como mostra as imagens a seguir:

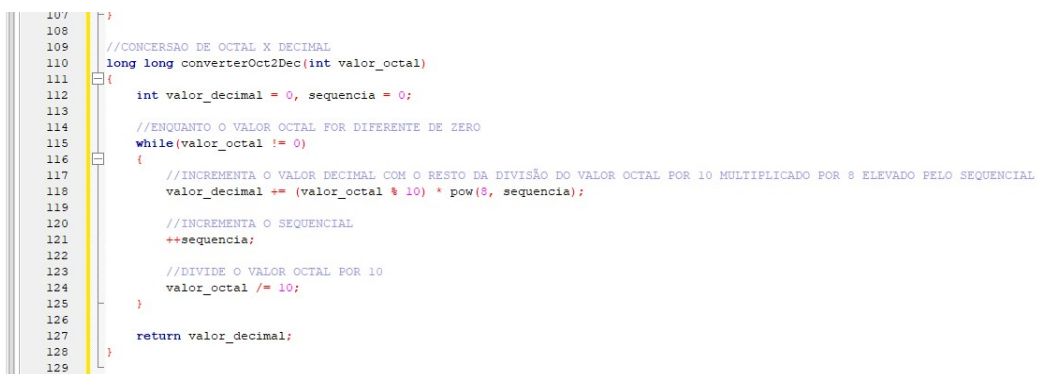
```
36
37 //FUNCOES PARA CONVERSOES
38
39 //CONVERSAO DE BINARIO X DECIMAL
40 int converterBin2Dec(long long valor_binario)
41 {
42     int valor_decimal = 0, sequencial = 0, resto;
43
44     //ENQUANTO EXISTIR VALOR NO BINÁRIO
45     while (valor_binario != 0)
46     {
47         //PEGA O RESTO DA DIVISÃO DO VALOR POR 10
48         resto = valor_binario % 10;
49
50         //DIVIDE O VALOR BINARIO POR 10
51         valor_binario /= 10;
52
53         //INCREMENTA O VALOR DECIMAL COM O RESTO DA DIVISÃO MULTIPLICANDO POR 2 ELEVADO AO SEQUENCIAL
54         valor_decimal += resto * pow(2, sequencial);
55
56         //INCREMENTA A SEQUENCIAL
57         ++sequencial;
58     }
59     return valor_decimal;
60 }
61
62
```



```
61 }
62
63 //CONVERSAO DE DECIMAL X BINARIO
64 long long converterDec2Bin(int valor_decimal)
65 {
66     long long valor_binario = 0;
67     int resto, sequencial = 1;
68
69     //ENQUANTO O VALOR DECIMAL FOR DIFERENTE DE
70     while (valor_decimal != 0)
71     {
72         //PEGA O RESTO DA DIVISAO
73         resto = valor_decimal % 2;
74
75         //DIVIDE O VALOR DECIMAL POR 2
76         valor_decimal /= 2;
77
78         //INCREMENTA O VALOR BINARIO, MULTIPLICANDO O RESTO DA DIVISAO PELO SEQUENCIAL
79         valor_binario += resto * sequencial;
80
81         //MULTIPLICANDO O SEQUENCIAL POR 10
82         sequencial *= 10;
83     }
84
85     return valor_binario;
86 }
87
```



```
87
88 //CONVERSAO DE DECIMAL X OCTAL
89 int converterDec2Oct(int valor_decimal)
90 {
91     int valor_octal = 0, sequencia = 1;
92
93     //ENQUANTO O VALOR DECIMAL FOR DIFERENTE DE ZERO
94     while (valor_decimal != 0)
95     {
96         //INCREMENTA O VALOR OCTAL COM O RESTO DA DIVISAO DO DECIMAL POR 8 MULTIPLICANDO PELO SEQUENCIAL
97         valor_octal += (valor_decimal % 8) * sequencia;
98
99         //O VALOR DECIMAL SERA DIVIDIDO POR 8
100        valor_decimal /= 8;
101
102        //O SEQUENCIAL SERA MULTIPLICADO POR 10
103        sequencia *= 10;
104    }
105
106    return valor_octal;
107 }
108
```



```
107 }
108
109 //CONVERSAO DE OCTAL X DECIMAL
110 long long converterOct2Dec(int valor_octal)
111 {
112     int valor_decimal = 0, sequencia = 0;
113
114     //ENQUANTO O VALOR OCTAL FOR DIFERENTE DE ZERO
115     while(valor_octal != 0)
116     {
117         //INCREMENTA O VALOR DECIMAL COM O RESTO DA DIVISAO DO VALOR OCTAL POR 10 MULTIPLICADO POR 8 ELEVADO PELO SEQUENCIAL
118         valor_decimal += (valor_octal % 10) * pow(8, sequencia);
119
120         //INCREMENTA O SEQUENCIAL
121         ++sequencia;
122
123         //DIVIDE O VALOR OCTAL POR 10
124         valor_octal /= 10;
125     }
126
127     return valor_decimal;
128 }
129
```

Com as funções prontas comecei a montar as condicionais para cada opção que o usuário pode selecionar.

```

34
35 //OPÇÃO 1 - BINÁRIO -> DECIMAL
36 if (opcao == '1') {
37     printf("Digite o número binário: ");
38     scanf("%lld", &valor_binario);
39     printf("[%lld] em binário = [%d] em decimal\n", valor_binario, converterBin2Dec(valor_binario));
40 }
41
42 //OPÇÃO 2 - DECIMAL -> BINÁRIO
43 else if (opcao == '2') {
44     printf("Digite o número decimal: ");
45     scanf("%d", &valor_decimal);
46     printf("[%d] em decimal = [%lld] em binário\n", valor_decimal, converterDec2Bin(valor_decimal));
47 }
48
49 //OPÇÃO 3 - DECIMAL -> OCTAL
50 else if (opcao == '3') {
51     printf("Digite o número decimal: ");
52     scanf("%d", &valor_decimal);
53     printf("[%d] em decimal = [%d] em octal\n", valor_decimal, converterDec2Oct(valor_decimal));
54 }
55
56 //OPÇÃO 4 - OCTAL -> DECIMAL
57 else if (opcao == '4') {
58     printf("Digite o número octal: ");
59     scanf("%d", &valor_octal);
60     printf("[%d] em octal = [%lld] em decimal\n", valor_octal, converterOct2Dec(valor_octal));
61 }
62 //OPÇÃO DESCONHECIDA
63 else {
64     printf("Opção desconhecida [%c]\n", opcao);
65 }
66
67 ///FUNCOES PARA CONVERSOES
68

```

Com as condicionais criadas a aplicação já começa a funcionar e converter as bases, porém preferi fazer um laço de repetição para poder voltar ao menu quando uma conversão for executada e para fechar quando clicar em “0”.

Para isso adicionei mais uma variável:

```

*main.c X
1  #include <stdio.h>
2  #include <math.h>
3  #include <string.h>
4
5  //CHAMADA DAS FUNÇÕES UTILIZADAS
6  int converterBin2Dec(long long valor_binario);
7  long long converterDec2Bin(int valor_decimal);
8  int converterDec2Oct(int valor_decimal);
9  long long converterOct2Dec(int valor_octal);
10
11 int main()
12 {
13
14     //VARIÁVEIS UTILIZADAS
15     char opcao = ' ';
16     int valor_octal;
17     int opcao2 = 1;
18     long long valor_binario;
19     int valor_decimal, cociente;
20     int sequencial, valor_temp, tamanho_string;
21     char valor_hexa[100];
22

```

Também adicionei uma função While antes do menu do Sistema e nas condicionais adicionei a opção de digitar 0 que preencher a variavel opcao2 com o valor zero e fecha o sistema:

```
//VARIABLES UTILIZADAS
char opcao = ' ';
int valor_octal;
int opcao2 = 1;
long long valor_binario;
int valor_decimal, cociente;
int sequencial, valor_temp, tamanho_string;
char valor_hexa[100];

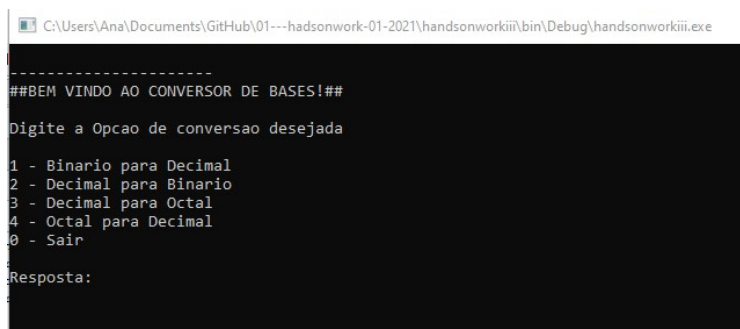
//ENQUANTO FOR A OPCAO 1 EXECUTA NOVAMENTE, SE FOR DIFERENTE DE 1 SISTEMA É FINALIZADO
while (opcao2 == 1) {
    //MOSTRA O MENU PRINCIPAL
    printf("\n-----\n");
    printf("##BEM VINDO AO CONVERSOR DE BASES!##\n");
    printf("\nDigite a Opcao de conversao desejada\n");
    printf("1 - Binario para Decimal\n");
    printf("2 - Decimal para Binario\n");
    printf("3 - Decimal para Octal\n");
    printf("4 - Octal para Decimal\n");
    printf("0 - Sair\n");
    printf("\nResposta: ");
    scanf("%c", &opcao);
    getchar();

    //OPÇÃO 0 - FECHA O SISTEMA
    if (opcao == '0') {
        printf("Clique em ENTER e o sistema sera finalizado\n");
        opcao2 = 0;
        exit(0);
    }
}
```

Link para Código: <https://github.com/anapaulaantunesaraujo/ConversorDeBases>

6.1. Como ficou?

Ao final a aplicação ficou com a aparência conforme imagem a seguir:



```
#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada
1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair
Resposta:
```

Vejamos um exemplo, testamos a conversão do valor decimal 80 entre as bases disponíveis na aplicação.

Primeiro de binário para decimal, no caso 80 em binário é 01010000:


```
C:\Users\Ana\Documents\GitHub\01---hadsonwork-01-2021>

#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada

1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair

Resposta: 1
```

```
C:\Users\Ana\Documents\GitHub\01---hadsonwork-01-2021>

#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada

1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair

Resposta: 1
Digite numero binario:
```

```
C:\Users\Ana\Documents\GitHub\01---hadsonwork-01-2021>

#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada

1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair

Resposta: 1
Digite numero binario: 01010000
```

```
C:\Users\Ana\Documents\GitHub\01---hadsonwork-01-21-2021>

#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada

1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair

Resposta: 1
Digite numero binario: 01010000
[01010000] em binario = [80] em decimal

#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada

1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair

Resposta:
```

Agora de decimal para binário:

```
C:\Users\Ana\Documents\GitHub\01---hadsonwork-01-2021\

#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada

1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair

Resposta: 1
Digite numero binario: 01010000
[1010000] em binario = [80] em decimal

#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada

1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair

Resposta: 2
```

```
#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada

1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair

Resposta: Digite o numero decimal: 80
```

```
#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada

1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair

Resposta: Digite o numero decimal: 80
[80] em decimal = [1010000] em binario

#####
##BEM VINDO AO CONVERSOR DE BASES!##
Digite a Opcao de conversao desejada

1 - Binario para Decimal
2 - Decimal para Binario
3 - Decimal para Octal
4 - Octal para Decimal
0 - Sair

Resposta:
```

Referências

GASPAROTTO, Henrique Machado. Lógica de programação: introdução a algoritmos e pseudocódigo. 2017. Disponível em: <<https://www.devmedia.com.br/logica-de-programacao-introducao-a-algoritmos-e-pseudocodigo/37918>>. Acesso em: 13 mar. 2021.

GARCIA, Ney. Lógica De Programação De Computadores: Introdução à Construção de Algoritmos. Joinville: Clube de Autores (managed), 2011. 163 p.

CÓRDOVA JUNIOR, Ramiro Sebastião. FUNDAMENTOS COMPUTACIONAIS. Porto Alegre: SAGAH, 2018. 194 p.

ALVES, Gustavo Furtado de Oliveira. As 10 conversões numéricas mais utilizadas na computação. 2018. Disponível em: <<https://dicasdeprogramacao.com.br/as-10-conversoes-numericas-mais-utilizadas-na-computacao/>>. Acesso em: 13 mar. 2021.