**EXERCÍCIO PRÁTICO (LABORATÓRIO)**

**Instruções:** O programa deverá ser salvo com o nome "24102017_LabAEDs.c".
Os arquivos (*.c) deverão ser enviados para: kellersullivan@dcc.ufmg.br e para pedro.fortini@dcc.ufmg.br

Atividade:

O CPF (Cadastro de Pessoa Física) é um documento emitido pela Receita Federal que contém 11 dígitos, os quais, os dois últimos são os dígitos verificadores.

Para a validação do CPF, utiliza-se como base o cálculo de seus 9 primeiros dígitos, como pode ser observado a seguir:

Consideremos a geração de um número de CPF válido, que hipoteticamente, tenha o seguinte formato: 234.567.891.XX em que XX serão os dígitos verificadores a serem encontrados.

I) Para encontrar o primeiro digito verificador, calcule a soma dos produtos dos nove dígitos:

$$Soma = \sum_{i=2}^{10} \left( Vetor_{0,10-i} \cdot i \right) \quad \therefore$$
$$Soma = 2 \cdot 10 + 3 \cdot 9 + 4 \cdot 8 + 5 \cdot 7 + 6 \cdot 6 + 7 \cdot 5 + 8 \cdot 4 + 9 \cdot 3 + 1 \cdot 2 = 246 \quad (1)$$

II) O primeiro número verificador será 0, se o resto da divisão de 11 pela soma dos produtos for 0 ou 1; caso contrário, será atribuído ao Vetor[9] a subtração de 11 pelo resto da divisão da soma dos produtos por 11.

Ex.:

O resto da divisão de 246/11 será 4;
Assim, o número verificador será encontrado pela subtração de 11-4=7.

O primeiro dígito verificador do CPF a ser armazenado na décima posição do vetor para este exemplo será 7.

III) Para encontrar o segundo dígito verificador, calcule a soma dos produtos dos 10 primeiros dígitos (incluindo o primeiro dígito verificador)

$$Soma = \sum_{i=2}^{11} \left( Vetor_{0,11-i} \cdot i \right) \quad \therefore$$
$$Soma = 2 \cdot 11 + 3 \cdot 10 + 4 \cdot 9 + 5 \cdot 8 + 6 \cdot 7 + 7 \cdot 6 + 8 \cdot 5 + 9 \cdot 4 + 1 \cdot 3 + 7 \cdot 2 = 305 \quad (2)$$

IV) O segundo número verificador será 0, se o resto da divisão de 11 pela soma dos produtos for 0 ou 1; caso contrário, será atribuído ao Vetor[10] a subtração de 11 pelo resto da divisão da soma dos produtos por 11.

Ex.:

O resto da divisão de 305/11 será 8;
Assim, o segundo número verificador será encontrado pela subtração de 11-8=3.

1) Utilizando as informações sobre verificação do CPF e geração de dígitos verificadores, crie um programa que verifique se um CPF digitado é válido ou não. Para isso, utilize os seguintes passos:

a) O programa deve conter o seguinte fragmento de código:

```
1.  #include <stdio.h>
2.  void leitura(char *);
3.  int verificaCPF(char *);
4.  void imprime(char *);
5.  int main()
6.    {
7.    char CPF[12];
8.    leitura(CPF);
9.    imprime(CPF);
10.   return 0;
11.   }
```

b) O programa deverá solicitar o número de CPF (sem caracteres especiais, somente os números) e a saída do programa deverá apresentar o CPF formatado, caso ele seja válido, da seguinte forma:

```
Digite o numero de CPF (Obs: Somente numeros): 23456789173

CPF Valido: 234.567.891-73
```

c) Material complementar para o desenvolvimento do programa:

## TABELA ASCII

| Dec | Hex | Oct | Char | Dec | Hex | Oct | Char | Dec | Hex | Oct | Char | Dec | Hex | Oct | Char |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | | 32 | 20 | 40 | [space] | 64 | 40 | 100 | @ | 96 | 60 | 140 | ` |
| 1 | 1 | 1 | | 33 | 21 | 41 | ! | 65 | 41 | 101 | A | 97 | 61 | 141 | a |
| 2 | 2 | 2 | | 34 | 22 | 42 | " | 66 | 42 | 102 | B | 98 | 62 | 142 | b |
| 3 | 3 | 3 | | 35 | 23 | 43 | # | 67 | 43 | 103 | C | 99 | 63 | 143 | c |
| 4 | 4 | 4 | | 36 | 24 | 44 | $ | 68 | 44 | 104 | D | 100 | 64 | 144 | d |
| 5 | 5 | 5 | | 37 | 25 | 45 | % | 69 | 45 | 105 | E | 101 | 65 | 145 | e |
| 6 | 6 | 6 | | 38 | 26 | 46 | & | 70 | 46 | 106 | F | 102 | 66 | 146 | f |
| 7 | 7 | 7 | | 39 | 27 | 47 | ' | 71 | 47 | 107 | G | 103 | 67 | 147 | g |
| 8 | 8 | 10 | | 40 | 28 | 50 | ( | 72 | 48 | 110 | H | 104 | 68 | 150 | h |
| 9 | 9 | 11 | | 41 | 29 | 51 | ) | 73 | 49 | 111 | I | 105 | 69 | 151 | i |
| 10 | A | 12 | | 42 | 2A | 52 | * | 74 | 4A | 112 | J | 106 | 6A | 152 | j |
| 11 | B | 13 | | 43 | 2B | 53 | + | 75 | 4B | 113 | K | 107 | 6B | 153 | k |
| 12 | C | 14 | | 44 | 2C | 54 | , | 76 | 4C | 114 | L | 108 | 6C | 154 | l |
| 13 | D | 15 | | 45 | 2D | 55 | - | 77 | 4D | 115 | M | 109 | 6D | 155 | m |
| 14 | E | 16 | | 46 | 2E | 56 | . | 78 | 4E | 116 | N | 110 | 6E | 156 | n |
| 15 | F | 17 | | 47 | 2F | 57 | / | 79 | 4F | 117 | O | 111 | 6F | 157 | o |
| 16 | 10 | 20 | | 48 | 30 | 60 | 0 | 80 | 50 | 120 | P | 112 | 70 | 160 | p |
| 17 | 11 | 21 | | 49 | 31 | 61 | 1 | 81 | 51 | 121 | Q | 113 | 71 | 161 | q |
| 18 | 12 | 22 | | 50 | 32 | 62 | 2 | 82 | 52 | 122 | R | 114 | 72 | 162 | r |
| 19 | 13 | 23 | | 51 | 33 | 63 | 3 | 83 | 53 | 123 | S | 115 | 73 | 163 | s |
| 20 | 14 | 24 | | 52 | 34 | 64 | 4 | 84 | 54 | 124 | T | 116 | 74 | 164 | t |
| 21 | 15 | 25 | | 53 | 35 | 65 | 5 | 85 | 55 | 125 | U | 117 | 75 | 165 | u |
| 22 | 16 | 26 | | 54 | 36 | 66 | 6 | 86 | 56 | 126 | V | 118 | 76 | 166 | v |
| 23 | 17 | 27 | | 55 | 37 | 67 | 7 | 87 | 57 | 127 | W | 119 | 77 | 167 | w |
| 24 | 18 | 30 | | 56 | 38 | 70 | 8 | 88 | 58 | 130 | X | 120 | 78 | 170 | x |
| 25 | 19 | 31 | | 57 | 39 | 71 | 9 | 89 | 59 | 131 | Y | 121 | 79 | 171 | y |
| 26 | 1A | 32 | | 58 | 3A | 72 | : | 90 | 5A | 132 | Z | 122 | 7A | 172 | z |
| 27 | 1B | 33 | | 59 | 3B | 73 | ; | 91 | 5B | 133 | [ | 123 | 7B | 173 | { |
| 28 | 1C | 34 | | 60 | 3C | 74 | < | 92 | 5C | 134 | \ | 124 | 7C | 174 | \| |
| 29 | 1D | 35 | | 61 | 3D | 75 | = | 93 | 5D | 135 | ] | 125 | 7D | 175 | } |
| 30 | 1E | 36 | | 62 | 3E | 76 | > | 94 | 5E | 136 | ^ | 126 | 7E | 176 | ~ |
| 31 | 1F | 37 | | 63 | 3F | 77 | ? | 95 | 5F | 137 | _ | 127 | 7F | 177 | |

2) Com base no exercício anterior, crie um programa que siga os seguintes passos:
   a. Deve conter o seguinte fragmento de código:

```
1.  #include <stdio.h>
2.  #define TNome 50
3.  #define TClientes 3
4.
5.  struct TipoCadastro
6.    {
7.    char nome[TNome], CPF[12];
8.    };
9.
10. void leitura(struct TipoCadastro *);
11. int verificaCPF(char *);
12. void imprimir(struct TipoCadastro *);
13.
14. int main()
15.   {
16.   struct TipoCadastro Clientes[TClientes];
17.   leitura(Clientes);
18.   imprimir(Clientes);
19.   return 0;
20.   }
```

   b. Para todos os clientes cadastrados, deve-se verificar se o CPF é valido ou não.
   c. A saída deste programa deve ser similar ao apresentado a seguir:

```
Digite o Nome do Cliente 1: Jose
Digite o CPF do Ciente 1:  23456789173
Digite o Nome do Cliente 2: Antonio
Digite o CPF do Ciente 2:  12345678909
Digite o Nome do Cliente 3: Carlos
Digite o CPF do Ciente 3:  13579135759

Cliente 1: Jose
CPF: 234.567.891-73
Cliente 2: Antonio
CPF: 123.456.789-09
Cliente 3: Carlos
CPF: 135.791.357-59
```