Programação Procedimental
GBC014 – 2015/1
Prof. Renan Cattelan – www.facom.ufu.br/~renan

Prática 12

Revisão e exercícios avançados

Exercício 6 (Lista 7)

6. Os dois dígitos de verificação do CPF (constituído de 9 dígitos) são calculados através do algoritmo abaixo:

   (a) Etapa 1: cálculo de DV1
   - Soma 1: soma dos produtos de cada dígito por um peso de 2 a 10, na ordem inversa (do nono para o primeiro).
   - Multiplique a soma 1 por 10 e calcule o resto da divisão do resultado por 11. Se der 10, DV1 é zero, caso contrário o DV1 é o próprio resto.

   (b) Etapa 2: cálculo de DV2
   - Soma 2: soma dos produtos de cada dígito por um peso de 3 a 11, também na ordem inversa.
   - Adicione a Soma 2 ao dobro do DV1, multiplique por 10 e calcule o resto da divisão do resultado por 11. Se der 10, DV2 é zero, caso contrário o DV2 é o próprio resto.

   (c) Etapa 3: Multiplique DV1 por 10, some com DV2 e você tem o número de controle do CPF.

```c
#include<stdio.h>

int dv1(int n) {
    int i, vn[9], soma1=0;
    for (i = 0; i<9; i++) {
        vn[i] = n % 10;
        n = n / 10;
        soma1 = soma1 + vn[i]*(i+2);
    }
    soma1 = soma1*10;
    if ((soma1%11) == 10)
        return 0;
    else return soma1%11;
}

int dv2(int n, int dv1) {
    int  i, vn[9], soma2=0;
    for (i = 0; i<9; i++) {
        vn[i] = n % 10;
        n = n / 10;
        soma2 = soma2 + vn[i]*(i+3);
    }
    soma2 = (soma2 + (2*dv1)) * 10;
    if ((soma2%11) == 10)
        return 0;
    else return soma2%11;
}
```

```c
int main(void)
{
    int cpf, d1, d2;
    printf("Digite os primeiros 9 digitos do CPF: ");
    scanf("%d", &cpf);

    d1 = dv1(cpf);
    d2 = dv2(cpf, d1);
    printf("O numero de controle eh: %d%d\n", d1, d2);

    system("pause");
    return 0;
}
```

Exercício 2

- A função de Fibonacci é definida por:
  - F(0)= 0,
  - F(1)= 1, e
  - F(n) = F(n-1)+F(n-2) para n > 1.
- Descreva a função F em linguagem C. Faça uma versão iterativa e uma recursiva.

0	1	1	2	3	5	8	13	21
F(0)	F(1)	F(2)	F(3)	F(4)	F(5)	F(6)	F(7)	F(8)

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

int F(int n) {
  if (n == 0) {
    return 0;
  }
  int a = 1;
  int b = 1;
  int i;
  for (i = 3; i <= n; i++) {
    int c = a + b;
    a = b;
    b = c;
  }
  return b;
}

int main(void) {
    int n;

    printf("\nDigite o valor de n: ");
    scanf("%d", &n);

    printf("\nO %d-esimo termo de Fibonacci eh: %d\n", n, F(n));

    system("pause");
    return 0;
}
```

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

int F(int n) {
    if (n == 0) return 0;
    if (n == 1) return 1;
    if (n > 1) return (F(n-1) + F(n-2));
    return -1; /* erro no valor fornecido (n < 0) */
}

int main(void) {
    int n;

    printf("\nDigite o valor de n: ");
    scanf("%d", &n);

    printf("\nO %d-esimo termo de Fibonacci eh: %d\n", n, F(n));

    system("pause");
    return 0;
}
```

Exercício 3

- Modifique a função de Fibonacci do exercício 3 para que os valores calculados, à medida que a recursão progrida, sejam armazenados num vetor.

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

int vetor[46]; /* O 46o numero da sequencia de Fib eh o maximo
                  que pode ser armazenado em um int */
int F(int n) {
    if (n == 0) return vetor[0] = 0;
    if (n == 1) return vetor[1] = 1;
    if (n > 1) {
        if(vetor[n] !=0)
            return vetor[n];
        return vetor[n] = F(n-1) + F(n-2);
    }
}

int main(void)
{
    int n;

    printf("\nDigite o valor de n: ");
    scanf("%d", &n);

    printf("\nO %d-esimo termo de Fibonacci eh: %d\n", n, F(n));

    system("pause");
    return 0;
}
```