Algoritmos e Estruturas de Dados I (DCC/003)
Lista de Exercícios 04 – Estruturas de Controle e de Repetição / função

Nome: Guilherme de Abreu Lima Buitrago Miranda
Matrícula: 2018054788

```
#include <stdio.h>
#include <math.h>

int fatorial (int numero, int resultado){

        if (numero == 1){
                return resultado;
        }else{
                resultado = resultado * numero;
                numero--;
                fatorial(numero, resultado);
        }

}

//mdc3 = mdc(c(mdc(a,b)))

int mdc (int numeroA, int numeroB){

        //algoritmo de Euclides (recursivo)

        if (numeroB == 0)
      return numeroA;
  else
      return mdc(numeroB, numeroA % numeroB);
}

int fib (int numero){
        int a = 0;
        int b = 1;
        int aux;
        for (int i = 0; i < numero; ++i){
                //printf("%d ", b);
                aux = b;
                b = a + b;
                a = aux;
        }
        return b;
}

void primo (int numero){
```

```
	//crivo de erastostenes.
	int raiz = floor(sqrt(numero));
	int boo = 0;
	for (int i = 2; i < raiz; ++i)	{
		if (numero % i == 0){
			boo = 1;
		}
	}
	if (boo == 0)
		printf("O numero %d é primo\n", numero);
	else
		printf("O numero %d não é primo\n", numero);
}

void decrescente (int numero){
	for (int i = numero-1; i > 0; --i){
		printf("%d ", i);
	}
}

int res (int a, int b){

	float divReal = (float) a/b;
	float parteFracionada = divReal - (int)(a/b);
	float resto = parteFracionada * b;

	return round(resto); //round para evitar erros na divisao, pois o programa pode considerar 0,99999999 != 1.

}

int form (int numero){
	int soma = 0;
	for(int i = 1; i <= numero; i++)
		soma += i*i;

	return soma;
}

int mmc (int a, int b){

	return ((a*b)/mdc(a, b));
}

int div (int a, int b){
	int divisaoInteira = (int)a/b;
	return divisaoInteira;
```

```
}

float sqrt_(int numero){
	//metodo babilonico
	//https://en.wikipedia.org/wiki/Methods_of_computing_square_roots#Babylonian_method
	float precisao = 0.001;
	//um valor arbitrario é atribuido para x0. Poderia ser otimizado utilizando outros procedimentos, conforme artigo linkado

	float x0 = numero;
	float xn = 1;
	while(x0 - xn > precisao){
		x0 = (x0+xn)/2;
		xn = numero/x0;
	}

	return x0;
}

int dig(int numero){
	int soma = 0;
	while(numero != 0){
		int resto = numero % 10; // 10 = uma dezena = um dígito no sistema decimal.
		soma += resto;
		numero = numero / 10;
	}
	return soma;
}

int exp_ (int numK, int numN){

	int numKOriginal = numK;

	for (int i = 1; i < numN; ++i){
		numK *= numKOriginal;
	}

	return numK;
}

void crescente(int numero){
	for (int i = 1; i <= numero; ++i){
		printf("%d ", i);
	}
}
```

```
int main (){

	int numero;
	printf("Entre com o número \n");
	scanf("%d", &numero);

	//printf("%d \n", fatorial(numero, 1));

	/* MDC: Raciocínio análogo para 2 e para 3 números. mdc 3 numeros =
mdc(c(mdc(a,b))).
	int numeroA, numeroB, numeroC;
	printf("Entre com o número A, o número B e o numero C \n");
	scanf("%d %d %d", &numeroA, &numeroB, &numeroC);
	int resultadomdc = mdc(numeroA, numeroB);
	int resultadoFinal = mdc (resultadomdc, numeroC);
	printf("O mdc de %d, %d e %d é igual a %d\n", numeroA, numeroB, numeroC,
resultadoFinal);*/

	//printf("%d\n", fib(numero-1));
	//primo(numero);
	//decrescente (numero);

	/* Resto da divisao (mod). Comando round no retorno para evitar erros.
	int numeroA, numeroB;
	printf("Entre com o número A e com o numero B \n");
	scanf("%d %d", &numeroA, &numeroB);

	printf("O resto da divisão entre %d e %d é %d\n", numeroA, numeroB,
res(numeroA, numeroB));*/

	//printf("%d \n", form(numero));

	/*int numeroA, numeroB;
	printf("Entre com o número A e com o numero B \n");
	scanf("%d %d", &numeroA, &numeroB);*/

	//printf("%d \n", mmc(numeroA, numeroB));
	//printf("%d \n", div(numeroA, numeroB));
	//printf("%f\n", sqrt_(numero));
	//printf("A soma dos dígitos do numero %d é: %d\n", numero, dig(numero));
	//printf("%d \n", exp_(numeroA, numeroB));
	crescente(numero);


	return 0;
}
```