Módulo 1

Recursividade

Métodos e Algoritmos Computacionais

C++

(Rone Ilídio)

Pilha de Execução

- Quando uma função é chamada, cria-se uma área de memória para receber os valores de suas variáveis
- A função main é sempre a primeira
- As outras são empilhadas umas sobre as outras
- A superior sempre é a primeira a ser executada

```
#include <iostream.h>
int G (int a, int b) {
    int x;
    x = a + b;
    return x;
  }
  int F (int i, int j, int k) {
    int x;
    x = G (i, j);
    x = x + k;
    return x;
  }
  int main () {
    int i, j, k, y;
    i = 111; j = 222; k = 444;
    y = F (i, j, k);
    printf ("%d\n", y);
    getchar();
  }
```

Pilha de Execução

3º G → a=111 b=222 x= a+b return x

2º F → i=111 j=222 k=444 x = G() return x

1º Main → i=111 j=222 k=444 y = F()

Resultado: imprime 777

Recursividade

- Funções recursivas são aquelas que chamam a se próprias
- Importante definir um critério de parada para evitar loop infinito
- A cada chamada da função, cria-se uma área de memória na pílha de execução

```
#include <iostream.h>
long fat(long n){
    if (n==0)
       return 1;
    return n * fat(n-1);
}
int main(){
    long num;
    cout << "Informe um numero:";
    cin >> num;
    long fatorial = fat(num);
    cout << "\nO fatorial de" << num << " e " <<
                                          fatorial << "\n";
    system("pause");
}
```

Pilha de Execução

5º fat(0)→ n=1 return 1

4º fat(1)→ n=1 return 1*fat(0)

3º fat(2)→ n=2 return 2*fat(1)

2º fat(3)→ n=3 return 3*fat(2)

1º main→num=3 fatorial=fat(3)

```
#include <conio.h>
#include <iostream.h>
int fib(int n){
    if (n==0)
       return 0;
    if (n==1)
       return 1;
    return fib(n-1)+ fib(n-2);
}
int main(){
    int n;
    cout << "Informe um numero:";
    cin >> n;
    int f = fib(n);
    cout << "\nO " << n <<"o numero da serie de fibonacci e " << f;
    getch();
}
```

Pilha de execução

- fib(0) n=0 return 0
- fib(1)→n=1 return 1
- fib(2) →n=2 return fib(1)+fib(0)
- fib(3) →n=3 return fib(2)+fib(1)
- fib(4) →n=4 return fib(3)+fib(2)
- 1º main→ n=4 fib(4)

```
#include <iostream.h>
#include <conio.h>
int maior(int v[], int n){
 if (n==1) return v[0];
 else{
        int x;
        int y = maior(v,n-1);
        if(y > v[n-1])
                x = y;
        else
                x = v[n-1];
        return x;
 }
}
int main(){
      int v[4] = {4,5,9,3};
      int m = maior(v,4);
      cout << "m:"<< m;
      getch();
}
```

Pilha de execução

- 5º maior→n=1 return v[0]=4
- 4º maior→n=2 return (maior(v,1) ou v[1]=5)
- 3º maior→n=3 return (maior(v,2) ou v[2]=9)
- 2º maior→n=4 return (maior(v,3) ou v[3]=3)
- 1º main→ v={4,5,9,3} m=maior(v,4)