UNIVERSIDADE FEDERAL DE MATO GROSSO
CAMPUS UNIVERSITÁRIO DO ARAGUAIA
Instituto de Ciências Exatas e da Terra
Curso de Bacharelado em Ciência da Computação

Professor: Ivairton M. Santos

Boas práticas de codificação

1) Use um editor adequado, que use de cores para identificar comentários, variáveis e palavras reservadas da linguagem, além de enumerar as linhas e identar eficientemente.
Ex.:

```
1  /* Exemplo de codificação utilizando um editor adequado de código em C++ */
2
3  #include <iostream>
4
5  classe Teste {
6  private:
7          int variavel1;
8  public:
9          //Método construtor
10         Teste() {
11                 variavel1 = 0;
12         }
13
14         //Método para incrementar variavel1
15         void incrementa() {
16                 if (variavel1 == 0) {
17                         variavel1++;
18                 } else {
19                         for (int i=0; i<=0; i++)
20                                 variavel1++/
21                 }
22         }
23 }
```

Linha: 10 Coluna: 1 INS LINHA C++ temp.cpp

2) Documente o arquivo, coloque a data de criação, o autor e a descrição daquele módulo/classe. Descreva também as funções e métodos e quando for adequado também variáveis/atributos relevantes.
Ex:

```
1  /* Universidade Federal de Mato Grosso
2   * Disciplina de Programação
3   * Data: 01/01/2000
4   * Professor: Ivairton
5   * Aluno: 'Chico'
6   *
7   * Descrição: Exemplo de módulo documentado adequadamente.
8   */
9
10 #include <stdlib.h>
11
12 /*
13  * Função para a soma de dois números inteiros
14  * val1: Inteiro com o primeiro valor a ser somado
15  * val2: Inteiro com o segundo valor a ser somado
16  * Retorno: Valor inteiro resultante da soma
17  */
18 int somaInteiros ( int val1, int val2 ) {
19         int aux;        // Variável auxiliar p/ registro da soma
20         aux = val1 + val2;
21         return aux;
22 }
```

Linha: 3 Coluna: 20 INS LINHA C++ temp.cpp

4) Se programar com nomes de variáveis/atributos em português, então use sempre português, evite misturar inglês+português. Mas prefira programar estritamente em inglês.

5) Seja criterioso na identação do código. Isso é muito importante para uma boa organização e leitura do código.

6) Modularize o melhor possível o código em funções/métodos. Avalie procedimentos que podem vir a ser utilizadas mais de uma vez e então defina uma função/método para isso.

7) Organize o sistema em arquivos separados. Se está programando orientado a objeto, então crie um documento para cada classe. Se está programando procedural, então crie um arquivo por estrutura de dados ou função macro.

8) Use espaço entre atribuições, chamada de funções, parâmetros, entre outros. Isso auxilia na leitura do código.
Ex.:

```
if ( x == y )
x = removeElemento( lista );
estrutura.vetor[ estrutura.posicao ] = x;
```

9) Evite rebuscar o código. Seja objetivo e claro. Pense que haverá uma outra pessoa que irá precisar estudar seu código, ou você mesmo no futuro.

10) Sempre que possível, escreva um arquivo “readme.txt” ou “leiame.txt” que descre como utilizar o sistema, quais são os parâmetros necessários e/ou opcionais para executar o sistema, e características relevantes inerente ao uso.