Universidade de Brasília UnB Gama
O novo endereço da Tecnologia.

Técnicas de Programação
202194 - Turma A

Documento de Estilo e Design

Grupo 03
Daniel
João Paulo
Simião
Victor Fellipe

Link para repositório: https://github.com/victorcabeceira/TECPROG_2-2014

1. Bibliotecas

As bibliotecas serão escritas separadas por espaçamento entre os termos.

2. Indentação

2.1 Comprimento de cada Linha

Linha de no máximo 140 caracteres, incluindo espaços em branco

2.2 Quebra de Linha

Caso a expressão seja maior que a linha, o critério de quebra de linha será:
- Quebrar linha depois da vírgula;
- Quebrar linha antes do operador;
- A nova linha deve começar no mesmo nível da linha da expressão de origem.

2.3 Início do Texto

- Cada nova função/método deve ser iniciada junto ao canto da tela;
- Para cada nova estrutura dentro de uma função/método, essa deve possuir um (1) espaçamento de tabulação a mais que o elemento anterior a ele.

3. Comentário

Antes de todo método/função deve se ter um comentário explicando o que o método/função faz. Os comentários devem ser feitos de acordo com a quantidade de linhas:
- 1 linha: espaço após '//' e ponto final, '.', no final do comentário (caso seja somente de descrição);
- 2 ou mais Linhas: espaçõ após '/*' e '*' seguido de espaço a cada linha de comentário. Ponto final '.' no final do comentário (caso seja somente descritivo) seguido de espaço e '*/'.

4. Declarações

4.1 Variáveis/Constantes

4.1.1 Quantidade por linha

Somente uma variável/constante por linha.

4.1.2 Localização

As variáveis/constantes serão definidas todas no começo do arquivo.

4.1.3 Inicialização

Inicializar as variáveis/constantes aonde forem declaradas, exceto as que forem resultado de operações.

4.1.4 Nome

As variáveis deverão ter nome condizente para o que será utilizada. Todas deverão iniciar com letra minúscula. Variáveis com mais de 1 palavra deverão obrigatoriamente começar com letra minúscula e as demais palavras com letra maiúscula. Já as constantes serão em letra maiúscula e caso seja formada por mais de palavra, estas serão separadas por undeline.

4.1.5 Variáveis Globais

Serão inicializadas após as bibliotecas, com o tipo e valor, caso tenha.

```
1. int primeiroElemento;
2. float segundoElemento;
3. int contador = 0;
```

5. Funções / Métodos

Regras a serem seguidas:
- Com espaço entre o nome da função/método ou estrutura e os parâmetros;
- Sem espaço entre o nome da função/método e os parênteses que iniciam sua lista de parâmetros;
- Abrir chaves ‘{‘ deve ser colocado após o parênteses que finaliza os parâmetros da função/método, ou seja, no final da mesma linha de declaração, sem utilização de espaço para separá-los;

- Fechar chaves ‘}’ deve ser colocado numa linha específica, abaixo da última declaração de tal função/método e sempre no mesmo alinhamento do método que esta pertence;
- Caso o método/função não possua declaração (nulo), deve-se abrir e fechar chaves após o nome do método, ‘{‘ seguido de ‘}’, sem espaçamento;
- É necessário sempre pular uma (1) linha em branco após a linha do nome da função e outra linha antes da chave que encerra a função/método.
- Cada método deve ser separado por uma (1) linha em branco;
- Caso o método só possui uma linha de declaração, pode-se não utilizar chaves, bastando apenas com que sua declaração siga a indentação.

6. Declaração de parâmetros

6.1 Declarações Simples

Uma nova declaração por linha.

6.2 Declarações Complexas

Uma declaração complexa é uma declaração que contém instruções com chaves ou outras estruturas para a exerção da função/método. Deve-se seguir o padrão de funções/métodos e estruturas, para não gerar erros e inconsistências no código, facilitando o entendimento.

6.3 Declaração de retorno

As declarações de retorno devem mostrar o elemento a ser retornado sem parênteses, a não ser quer o retorno seja uma expressão;

6.4 Operadores

Os operadores de parâmetro (como por exemplo ‘a **>=** b’), deverão sempre ter um espaço antes e outro espaço depois de serem utilizados.

6.5 Ponteiro

Os operadores para acesso de endereço/valor de determinado elemento de um ponteiro não deverão ser separados por espaço.

7. Estruturas

7.1 Estruturas de seleção

7.1.1 Estrutura if-else

A declaração das estruturas que envolvem a estrutura if devem ser feitas de acordo com os tópicos: 7.1.1.1,7.1.1.2 e 7.1.1.3.

7.1.1.1 if

```
1. if (parâmetro){
2.
3.          declaração;
4.
5. }
```

7.1.1.2 if-else

```
1. if (parâmetro){
2.
3.          declaração;
4.
5. }
6. else{
7.
8.          declaração;
9.
10. }
```

7.1.1.3 if-if-else-else if

```
1. if (parâmetro){
2.
3.          declaração;
4.
5.          if (parâmetro2){
6.
7.          declaração2;
8.
9.          }
```

```
10.         else{
11.
12.         declaração3;
13.
14.         }
15. }
16. else if (parâmetro3){
17.
18.         declaração4;
19.
20. }
```

7.1.2 Estrutura switch

A declaração da estrutura switch deve seguir a seguinte forma, onde caso houver algum ‘case’ onde ele passe através, deve-se adicionar um comentário indicando onde a instrução ‘break’ deveria estar:

```
1.  switch (parâmetro/condição){
2.
3.  case A:
4.
5.          declaraçãoA;
6.
7.          // Sem o ‘break;’.
8.
9.  case B:
10.
11.         declaraçãoB;
12.         break;
13. default:
14.
15.         declaraçãoDefault;
16.         break;
17.
18. }
```

7.2 Estruturas de repetição

7.2.1 Estrutura For

A declaração do ‘For’ deve ser feita da seguinte maneira:

```
1.  for (inicialização ; parâmetro/condição ; atualização){
2.
3.          declaração;
4.
5.  }
```

7.2.2 Estrutura While

A declaração das estruturas que envolvem a estrutura ‘while’ devem ser feitas de acordo com os tópicos: 7.2.2.1 e 7.2.2.2.

7.2.2.1 While

```
1.  while (parâmetro/condição){
2.
3.          declaração;
4.
5.   }
```

7.2.2.2 Do-while

```
1.  do {
2.
3.          declaração;
4.
5.  } while (parâmetro/condição);
```

8. Espaço em Branco

8.1 Linha em Branco

Deve-se utilizar uma linha em Branco nas seguinte situações:
- Após a primeira linha de cada função/método e antes da última linha de cada função/método;
- Antes de cada comentário de cada função/método;
- Antes e depois de todo comentário que não seja de descrição de algum método/função;
- Entre seções lógicas e trechos com encadeamento de estruturas ou funções/métodos (exemplo if-else).

8.2 Espaço em Branco

Deve-se utilizar um espaço em Branco nas seguintes situações:
- Entre qualquer palavra chave (estrutura) e seu parâmetro;
- Entre os operadores e os termos de parâmetro/expressão (parâmetro do if ou ‘;’ do For);
- Após cada termo de argumento (em métodos/funções);
Obs.: Não se deve utilizar espaço em branco entre os operadores binários e o termo que eles iterarem (contador) e nem entre o nome e parâmetro(s) de uma função/método.

9. Tabela de Nomes

O código será escrito em Inglês.

Identificador	Regra de Nomenclatura	Exemplo
Bibliotecas	Bibliotecas padrão serão separadas somente por espaço. Bibliotecas feitas para o projeto terão primeira letra maiúscula.	#include <iostream> #include “Example.h”

Comentário	Todo comentário que não for de uma função/método/estrutura/palavra reservada deve terminar com ponto final.	// Example.
		
		/*
		* Example.
		*/
		
		/* if (parameter){
		*
		* operation;
		*
		* }
		*/

Variáveis	Iniciar com letra minúscula e caso seja composta por mais de 1 palavra, as subsequentes devem ser escritas sem espaçamento e com a primeira letra maiúscula.	int exampleInteger;
Constantes	Ser Composta por letras maiúsculas e caso seja composta por mais de 1 palavra, as subsequentes devem ser separadas por underline.	int EXAMPLE_MIN;
Métodos/Funções	Primeira letra minúscula e caso seja composta por mais de 1	Animation(int x_, bool loop_);

	palavra, as subsequentes devem ser escritas sem espaçamento e com a primeira letra maiúscula. Caso tenha mais de 1 parâmetro, os subsequentes devem ser separados por vírgula e 1 espaçamento.	