

Convenções para código-fonte

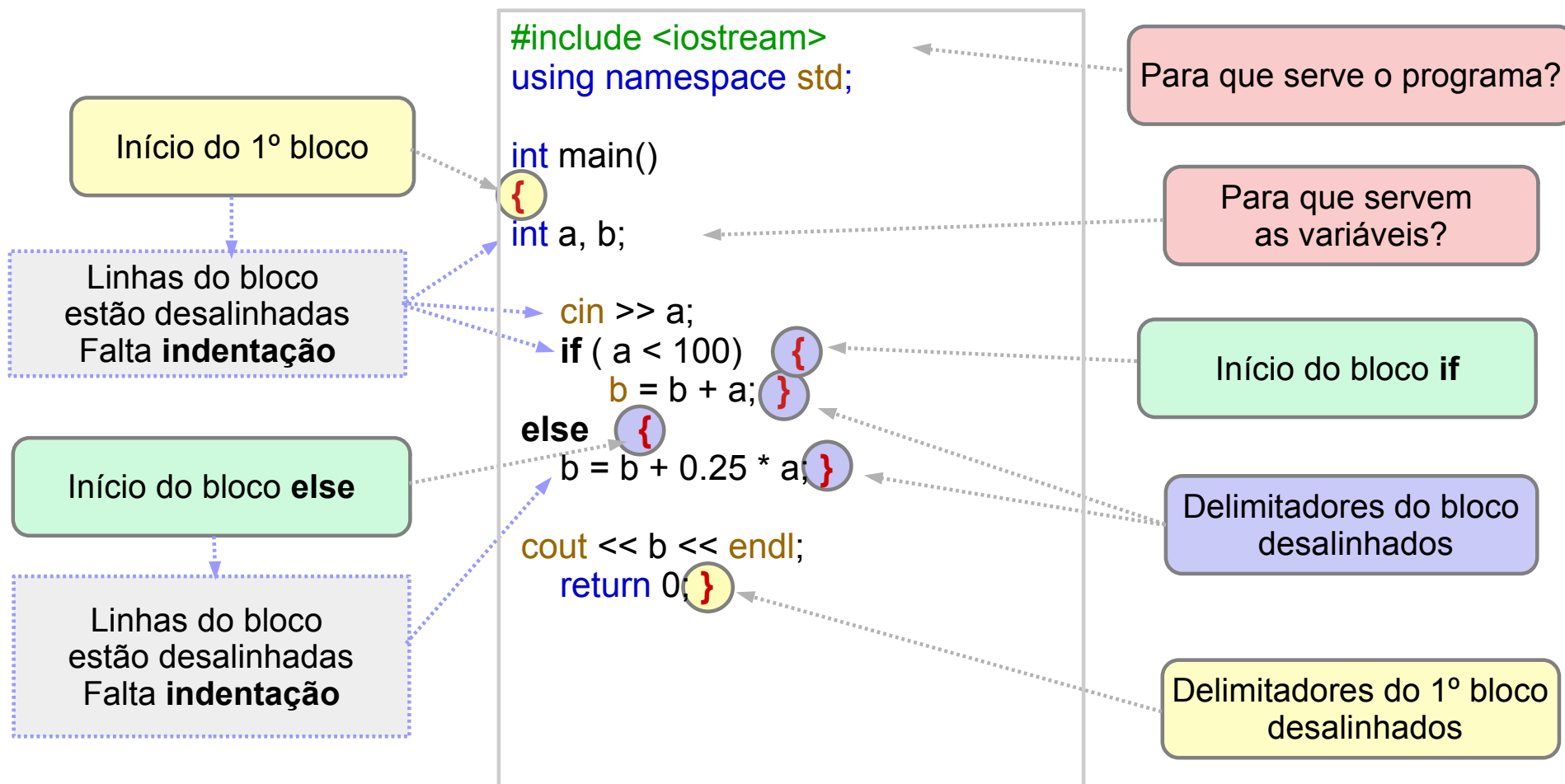
Sumário

- Legibilidade de código-fonte
- Indentação

Aspectos de legibilidade de código

- *Um código-fonte deve ser legível*
 - ▶ *comentários para as variáveis, comandos, repetições e desvios*
 - ▷ *indicam o objetivo dentro da solução do problema*
- *O conteúdo de blocos de comandos (delimitados por { e }) devem ser **INDENTADOS***
 - ▶ *espaços para a direita (geralmente 4 espaços).*
 - ▶ *{ e }* *não participam da indentação do bloco*
 - ▶ *Alguns editores já fazem isto automaticamente*

Exemplo de código-fonte não legível



Exemplo de código-fonte legível

```
/* Programa 'atualizaSaldo' : Ler saldo e valor de depósito e adicionar depósito ao saldo */
#include <iostream>
using namespace std;

int main()
{
    int dep, saldo; // dep: depósito na conta
                  // saldo: saldo bancário

    cin >> dep; // leitura de depósito

    if ( dep < 100) // leitura e soma de 10 valores
    {
        saldo = saldo + dep;
    }
    else
    {
        saldo = saldo + 0.25 * dep;
    }

    cout << saldo << endl; // Saldo final
    return 0;
}
```

Delimitadores de bloco estão alinhados

Linhas de 1º bloco alinhadas para dentro do bloco (INDENTAÇÃO)

Descrição do programa

Comentários descrevem variáveis, operações, desvios e repetições.

Linhas de blocos alinhadas para dentro do bloco (INDENTAÇÃO)

Exemplo de código-fonte legível

```
/* Programa 'atualizaSaldo' : Ler saldo e valor de depósito e adicionar depósito ao saldo */
#include <iostream>
using namespace std;

int main()
{
    int dep, saldo; // dep: depósito na conta
                  // saldo: saldo bancário

    cin >> dep; // leitura de depósito

    if ( dep < 100) { // leitura e soma de 10 valores
        saldo = saldo + dep;
    }
    else {
        saldo = saldo + 0.25 * dep;
    }

    cout << saldo << endl; // Saldo final
    return 0;
}
```

Descrição do programa

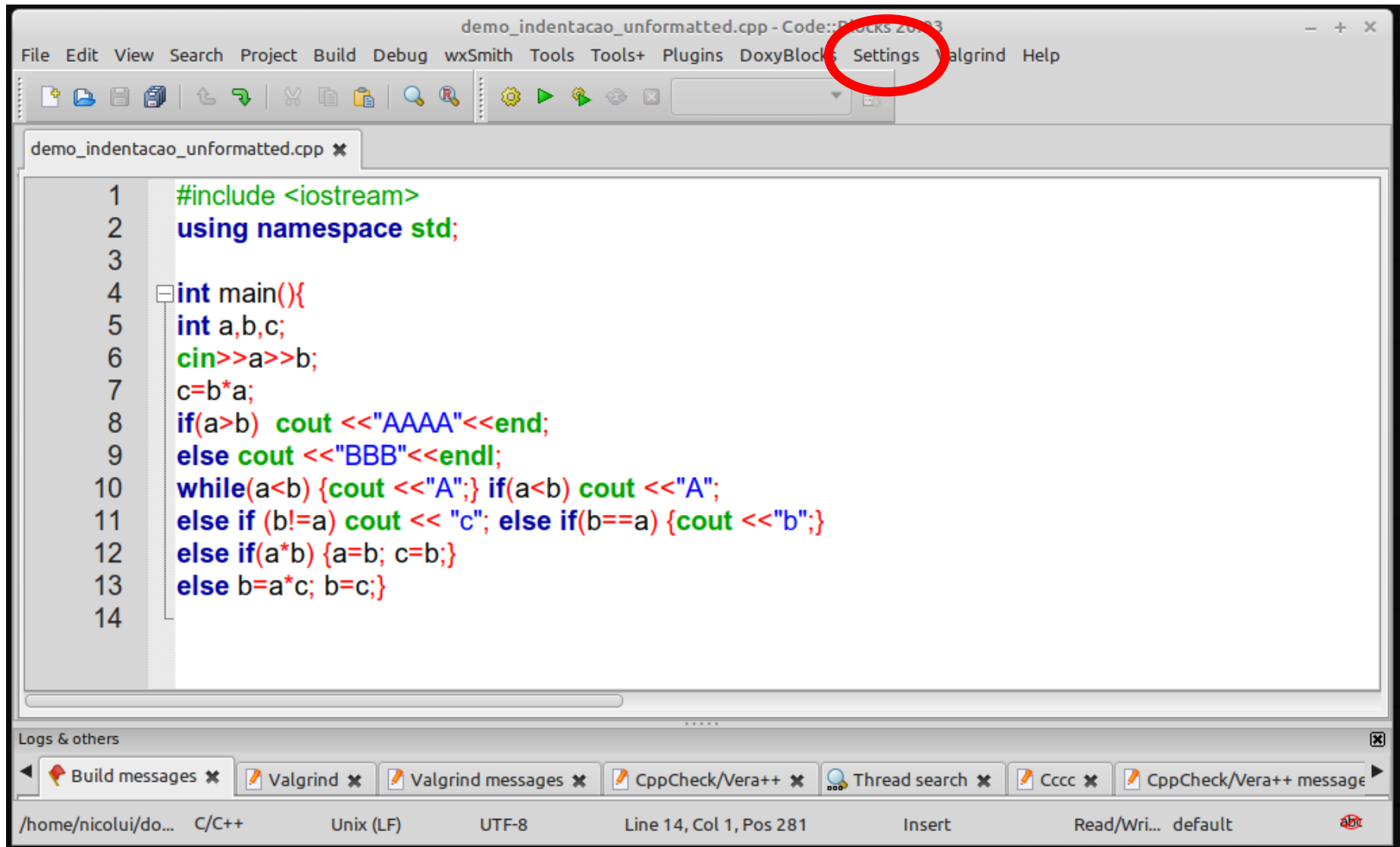
Delimitadores de bloco estão alinhados

Linhas de 1º bloco alinhadas para dentro do bloco (INDENTAÇÃO)

Comentários descrevem variáveis, operações, desvios e repetições.

Linhas de blocos alinhadas para dentro do bloco (INDENTAÇÃO)

Formatação de código-fonte no Code::Blocks

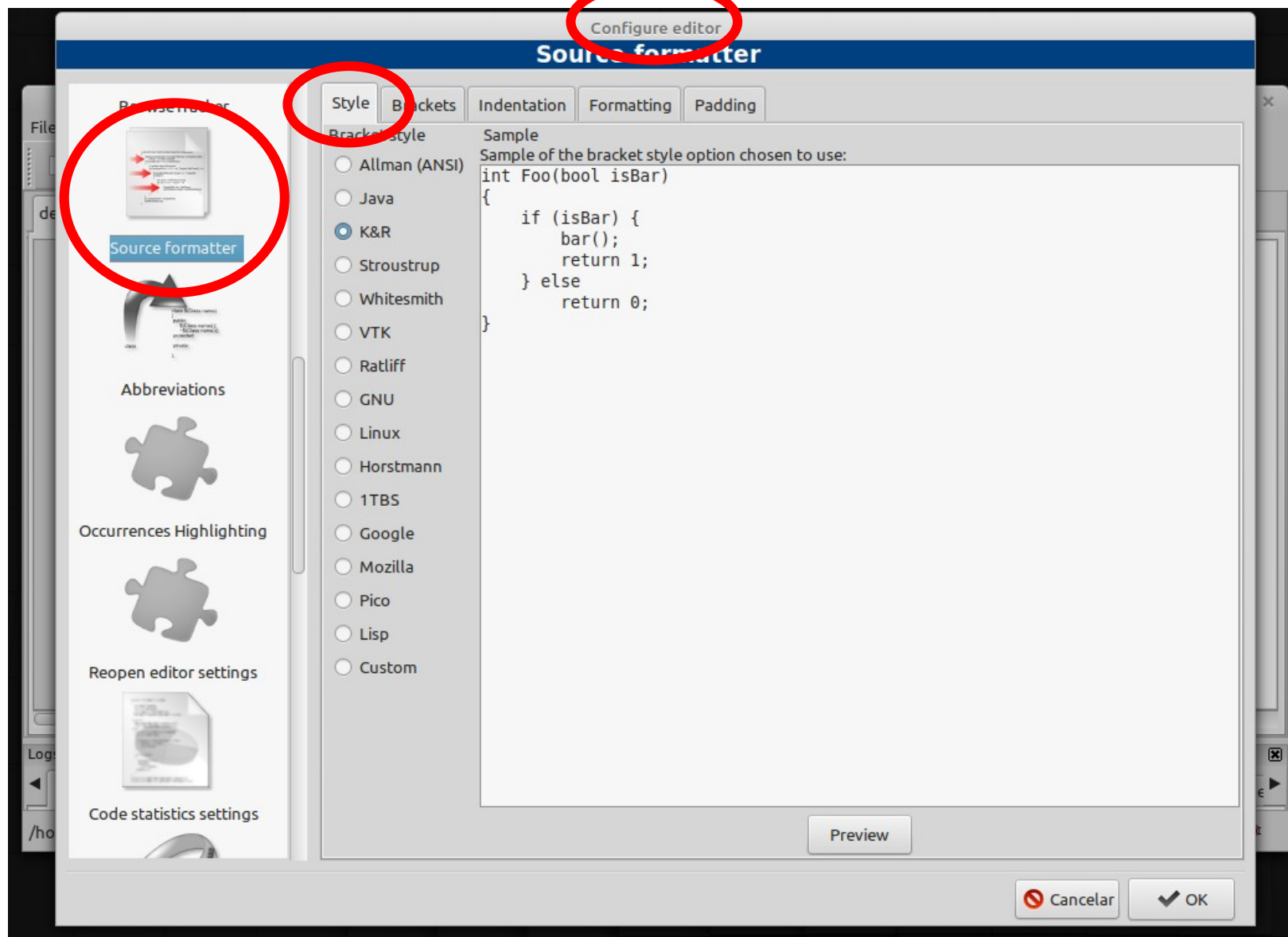


The screenshot shows the Code::Blocks IDE interface. The title bar reads "demo_indentacao_unformatted.cpp - Code::Blocks 20.03". The menu bar includes "File", "Edit", "View", "Search", "Project", "Build", "Debug", "wxSmith", "Tools", "Tools+", "Plugins", "DoxyBlocks", "Settings", "Valgrind", and "Help". The "Settings" menu is circled in red. The toolbar contains icons for file operations, search, and execution. The main editor window shows a C++ file named "demo_indentacao_unformatted.cpp" with the following code:

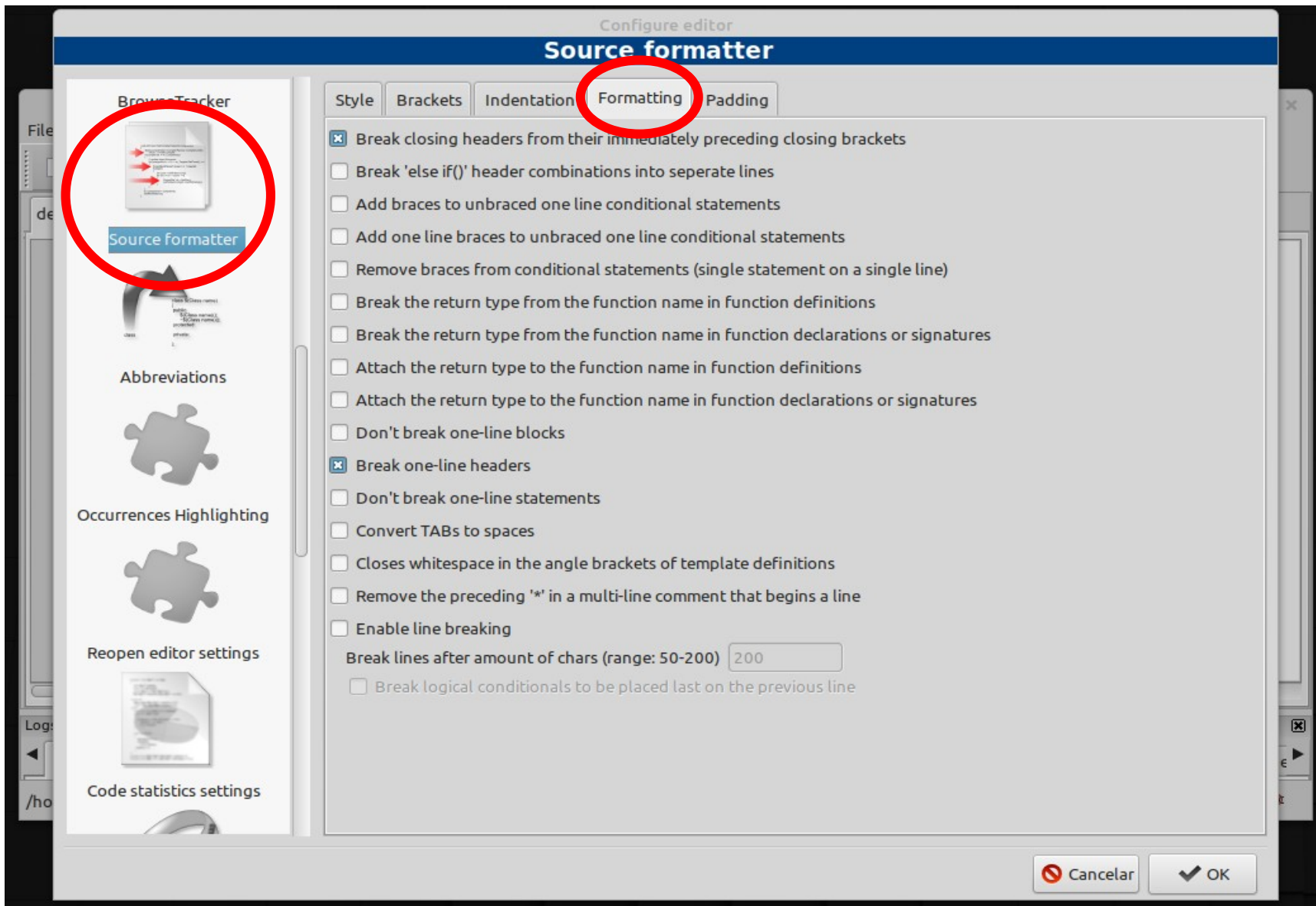
```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5  int a,b,c;
6  cin>>a>>b;
7  c=b*a;
8  if(a>b) cout <<"AAAA"<<end;
9  else cout <<"BBB"<<endl;
10 while(a<b) {cout <<"A";} if(a<b) cout <<"A";
11 else if (b!=a) cout << "c"; else if(b==a) {cout <<"b";}
12 else if(a*b) {a=b; c=b;}
13 else b=a*c; b=c;}
14
```

The bottom status bar shows the current file path, encoding (UTF-8), and cursor position (Line 14, Col 1, Pos 281). The "Logs & others" panel at the bottom contains tabs for "Build messages", "Valgrind", "Valgrind messages", "CppCheck/Vera++", "Thread search", "Cccc", and "CppCheck/Vera++ message".

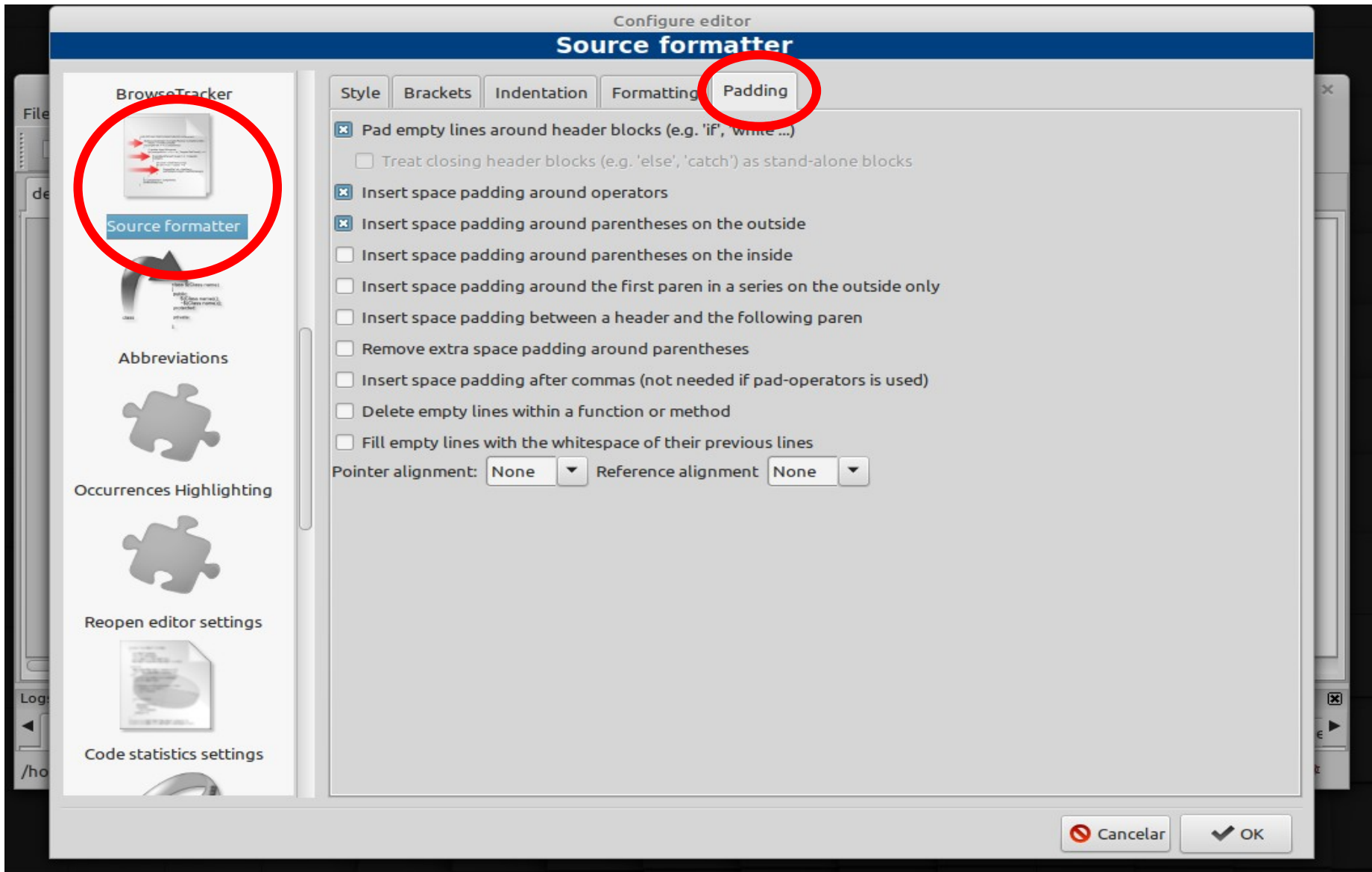
Formatação de código-fonte no Code::Blocks



Formatação de código-fonte no Code::Blocks

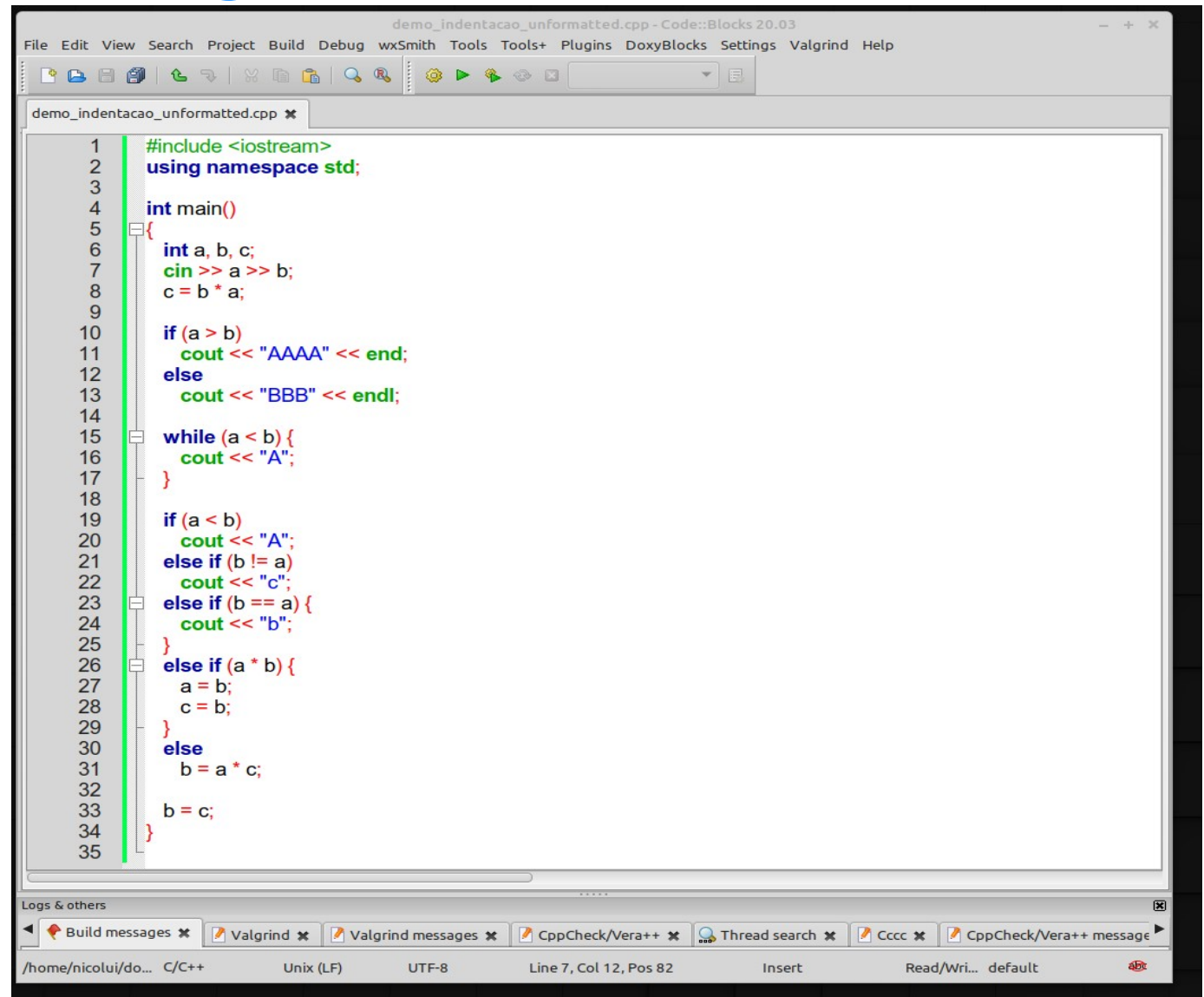


Formatação de código-fonte no Code::Blocks



Formatação de código-fonte no Code::Blocks

Resultado após ativar menu na área do código (botão direito) e escolher opção “**Format use AStyle**”



The screenshot shows the Code::Blocks IDE interface. The main window displays a C++ source file named 'demo_indentacao_unformatted.cpp'. The code is as follows:

```
1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6     int a, b, c;
7     cin >> a >> b;
8     c = b * a;
9
10    if (a > b)
11        cout << "AAAA" << endl;
12    else
13        cout << "BBB" << endl;
14
15    while (a < b) {
16        cout << "A";
17    }
18
19    if (a < b)
20        cout << "A";
21    else if (b != a)
22        cout << "c";
23    else if (b == a) {
24        cout << "b";
25    }
26    else if (a * b) {
27        a = b;
28        c = b;
29    }
30    else
31        b = a * c;
32
33    b = c;
34 }
35
```

The IDE interface includes a menu bar (File, Edit, View, Search, Project, Build, Debug, wxSmith, Tools, Tools+, Plugins, DoxyBlocks, Settings, Valgrind, Help), a toolbar, and a status bar at the bottom showing the current file path, encoding (UTF-8), and cursor position (Line 7, Col 12, Pos 82). A 'Logs & others' panel is visible at the bottom, containing tabs for Build messages, Valgrind, Valgrind messages, CppCheck/Vera++, Thread search, Cccc, and CppCheck/Vera++ message.

Créditos: O conteúdo deste documento é de autoria do Prof. Armando L.N. Delgado (DINF/ET) para uso na disciplina *Programação de Computadores* (CI208, CI180, CI183)

Compartilhe este documento de acordo com a licença abaixo



Este documento está licenciado com uma Licença *Creative Commons* **Atribuição-NãoComercial-SemDerivações** 4.0 Internacional.
<https://creativecommons.org/licenses/by-nc-sa/4.0/>