Instalação do Visual Studio Code (VS Code)

O Visual Studio Code é um editor de código-fonte desenvolvido pela Microsoft para **Windows**, **Linux** e **macOS**. Ele inclui suporte para **depuração**, controle **Git** incorporado, realce de sintaxe, complementação inteligente de código, *snippets*[1] e refatoração de código. Ele também é customizável, fazendo com que os usuários possam mudar o tema do editor, teclas de atalho e preferências. Ele é um ***software livre*** e de ***código aberto***, apesar do download oficial estar sob uma licença proprietária.

Pré-requisitos

É recomendável instalar antes do VS Code:

1. O Git (controle de versão). Caso não seja instalado, você ficará recebendo uma mensagem de sugestão dizendo "***Git not found. Install it or configure it using the 'git.path' setting.***" (Git não encontrado. Instale-o ou configure-o usando a configuração 'git.path'). Para solucionar este problema basta indicar o caminho até o git.exe instalado em teu computador na variável git.path das configuração do VSCode. Teremos algumas aulas de Git e isto será feito, mas, por enquanto, vamos simplesmente fechar a mensagem no botão "**Close**".

É necessário:

1. Acesso a internet para adquirir o VS Code e instalar as extensões necessárias.
2. Um compilador C/C++ (para este tutorial usaremos **MinGW**)

Como adquirir o VS Code

Acesse ao site https://code.visualstudio.com/download e escolha a opção que corresponda ao teu sistema operacional (Windows, Linux ou macOS).

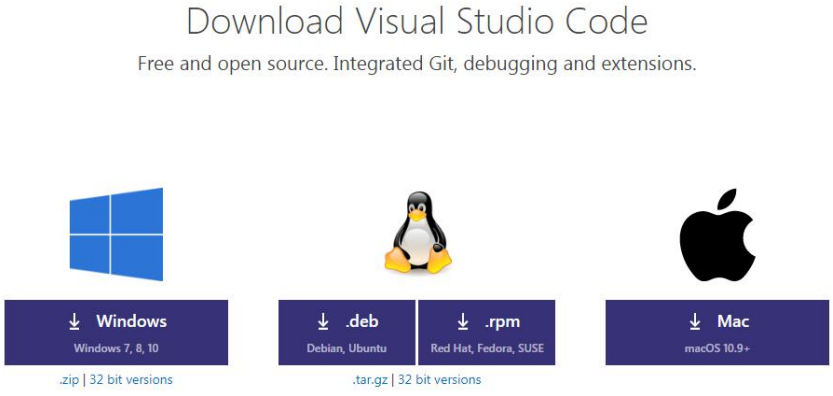

Figura 1 - Opções de download do Visual Code

Neste tutorial, usaremos a opção Windows baixando o arquivo **VSCodeSetup-x64-<VERSÃO>.exe**.

Quando você colocar o programa de instalação para executar, a primeira tela que você verá é a mostrada na figura 2. Esta é a tela de escolha do idioma. Infelizmente, não existe a opção Português. Sendo assim, ficaremos com a opção Inglês (*English*). Clique em ***OK***.

[1] são fragmentos de texto que podem ser reutilizados de forma rápida

Figura 2 - Seleção de idioma

Logo em seguida temos a tela de boas-vindas (Veja a figura 3). Basta clicar em ***Next*** (Próximo).

Figura 3 - Tela de Boas-vindas!

A próxima tela é a tela de aceite da licença. Clique em "*I accept the agreement*" ("Eu aceito o acordo") e, em seguida, ***Next*** (Próximo)

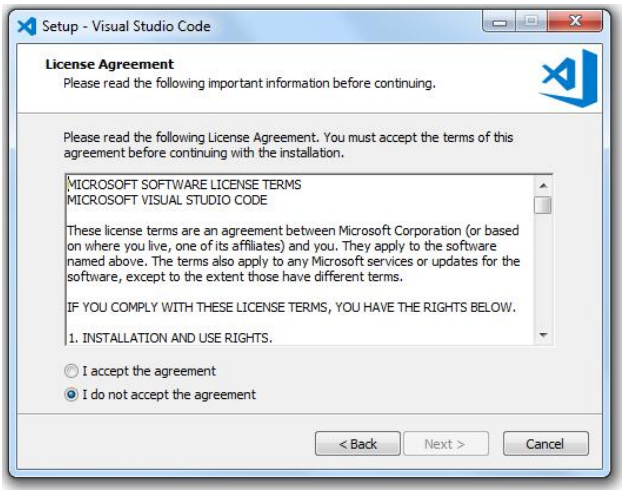

Figura 4 - Tela de Aceite da Licença

A tela seguinte é a tela de escolha da localização da instalação. Vamos optar por escolher o local sugerido pela instalação e simplesmente clicar em ***Next***. Assim como na próxima que é para escolher a pasta do menu iniciar. Vamos deixar com a opção padrão e clicar em ***Next***.

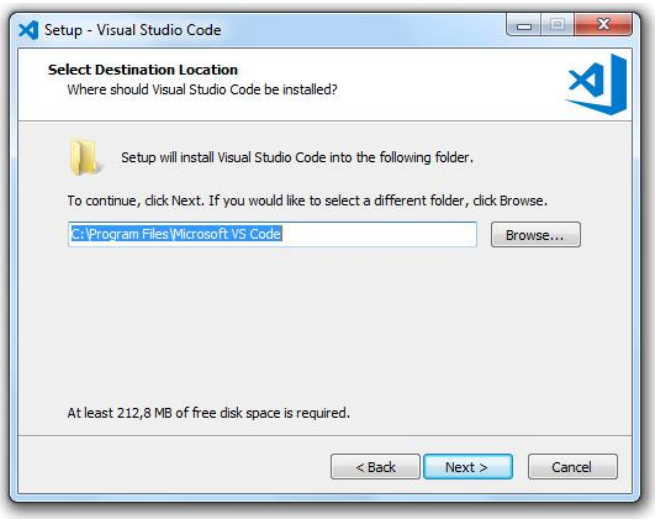

Figura 5 - Tela de Escolha de localização

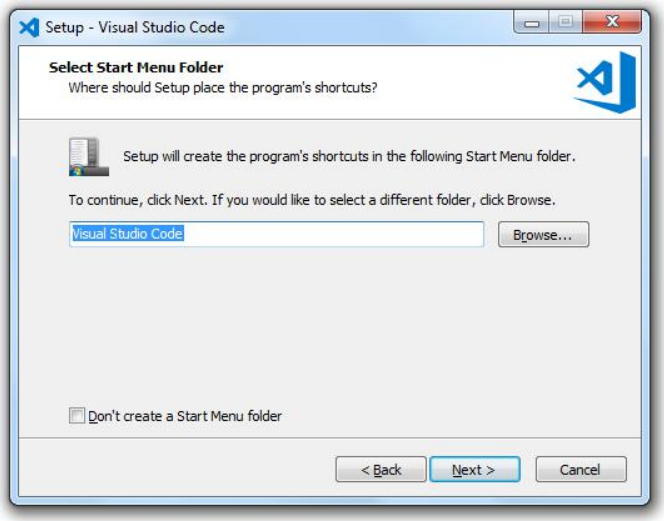

Figura 6 - Tela de escolha da Pasta do Menu Iniciar

A figura 7 mostra a tela de Seleção de tarefas adicionais. Marque as opções "*Create a desktop icon*" (Crie um ícone na Área de Trabalho) e "*Add to PATH (available after restart)*" (Adicionar à PATH (disponível após reinicialização)) e, em seguida, clique em ***Next***.

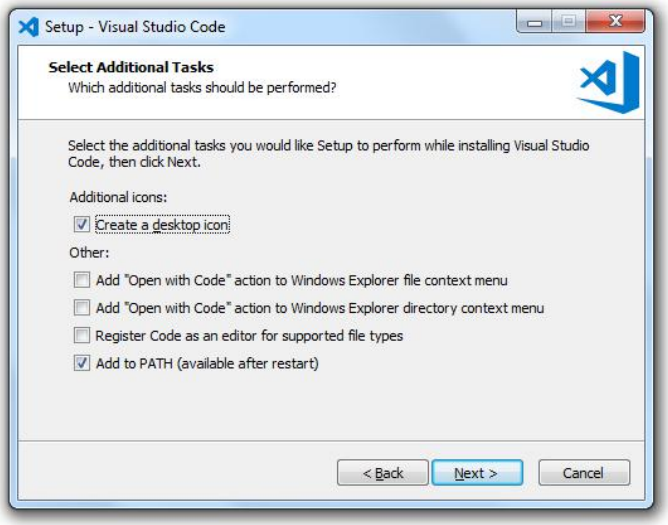

Figura 7 - Tarefas Adicionais

Na tela seguinte (*Ready to Install* (Pronto para instalar)) basta clicar em **Install** e esperar a conclusão da instalação. Veja figura 8.

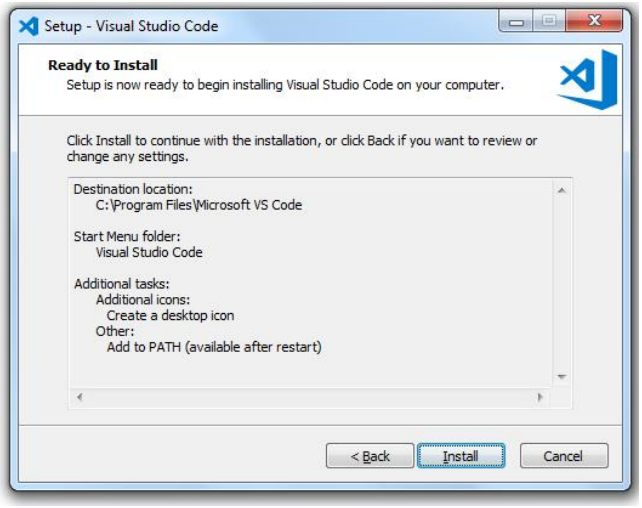

Figura 8 - Pronto para Instalar

Na última tela da instalação vamos deixar marcada a opção "*Launch Visual Studio Code*" ('Abrir'[2] Visual Studio Code) e clicar em ***Finish***. Veja a figura 9.

Figura 9 - Concluindo a instalação

Em seguida você verá a tela inicial do Visual Studio Code. Como mostra a figura 10.

[2] Literalmente 'Lançar'

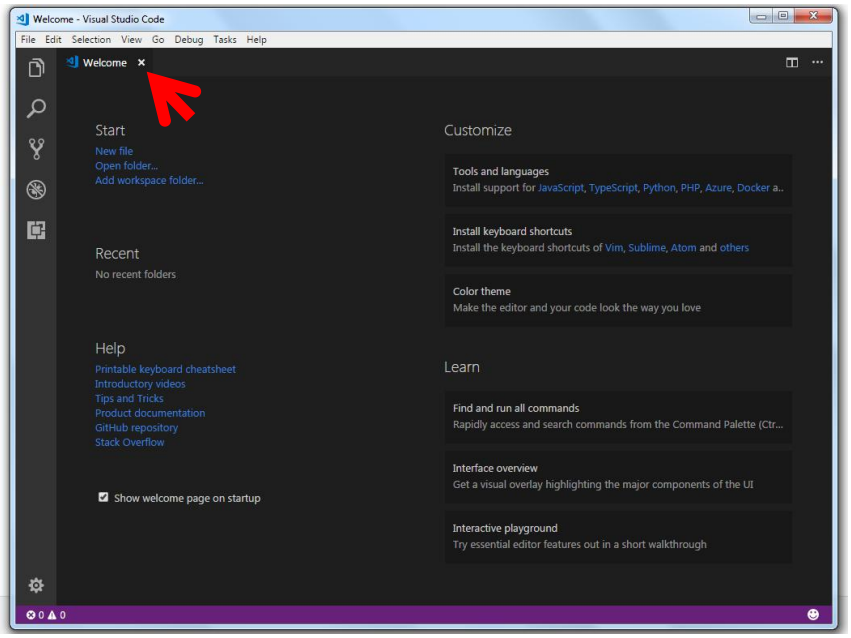

Figura 10 - Tela de Boas-Vindas do VS Code

Vamos fechar a tela de boas vindas clicando no 'x' à direita da palavra "*Welcome*" na parte superior esquerda da tela conforme aponta a figura 10.

Nosso próximo passo é configura um diretório (pasta) de trabalho. Dê um clique no ícone "*Explorer*" apontado pela figura 11.

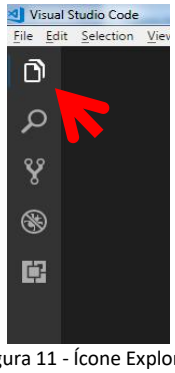

Figura 11 - Ícone Explorer.

O próximo passo é cliclar no botão "***Open Folder***" apontado pela figura 12. Na janela "Open Folder" que será aberta, escolha uma pasta de trabalho. Para este tutorial, escolhemos a pasta **VSCodeFolder** que criamos antes na pasta "Documentos" do usuário logado.

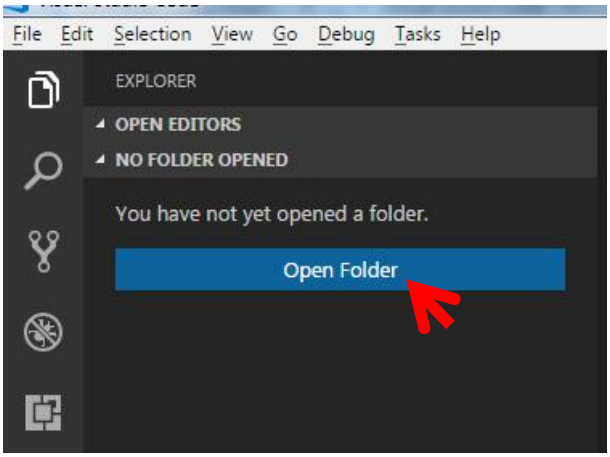

Figura 12 - Botão *Open Folder* (Abrir Pasta)

Tendo feito isto, a janela de boas vindas será aberta novamente com algumas instruções de uso do Visual Studio Code. Feche a aba "Welcome" novamente e observe que agora temos o nome da pasta no Explorer como aponta a figura 13.

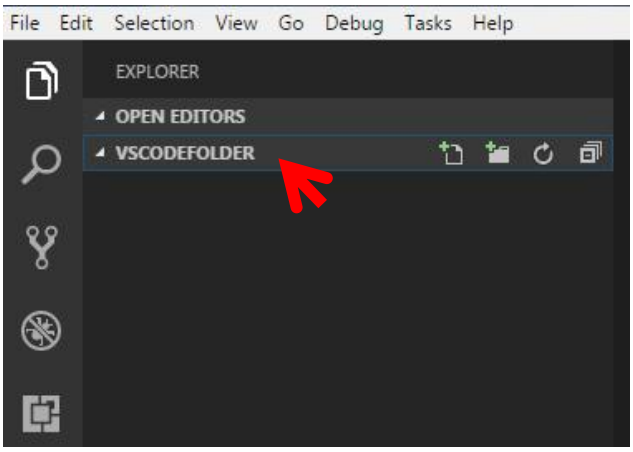

Figura 13 - Pasta de trabalho no Explorer

Ao aproximar o ponteiro do mouse do nome da pasta de trabalho, pequenos ícone aparecem do lado direito do nome da pasta. O primeiro deles é o "New File" (Novo Arquivo) conforme aponta a figura 14. Clique sobre ele e você verá que um pequeno campo será aberto para colocar o nome do arquivo. Coloque **primeiro.cpp** uma vez que este será o nosso primeiro programa em C++ escrito no VS Code. Em seguida confirme com "Enter".

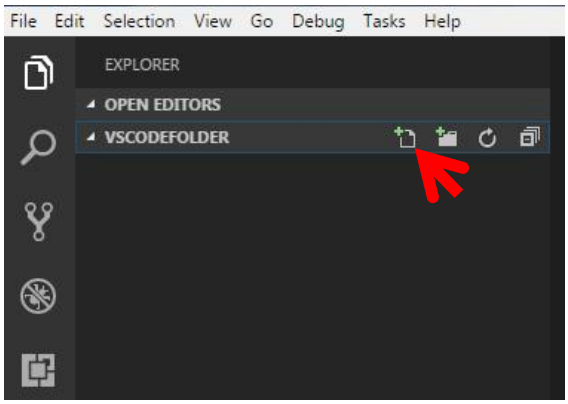

Figura 14 - New file (novo arquivo)

Neste momento a sugestão de instalar extensões de C/C++ lhe será dada. Veja a figura 15. Clique em "Install" para que o que for necessário instalar para uso do C++ no VS Code seja instalado.

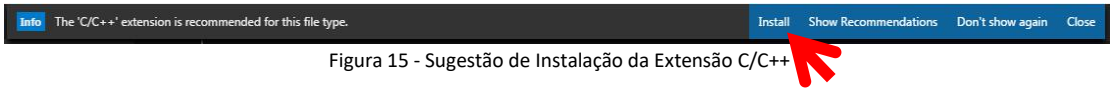

Figura 15 - Sugestão de Instalação da Extensão C/C++

Fazendo isto, um lista de opções de extensão será aberta conforme mostra a figura 16. A primeira será instalada automaticamente. Clique no pequeno botão verde "Install" à direita da extensão C++ IntelliSense conforme aponta a figura 16.

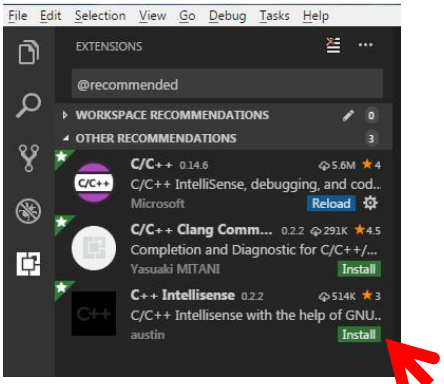

Figura 16 - Lista de opções de extensão para C/C++

O próximo passo é configurar o VSCode para usar o conjunto de ferramentas de compilação. Vamos fazer isto em forma de receita. Vamos dar o passo a passo e você vai seguindo:

A. Configurar um novo terminal padrão

1. Dê um "**ctrl + J**" para abrir o painel e em seguite clique na aba "**Terminal**". Você verá que o terminal foi aberto direto na pasta de trabalho conforme mostra a figura 17. Digite ***dir*** e pressione "Enter" e você verá o arquivo **primeiro.cpp** na lista apresentada.

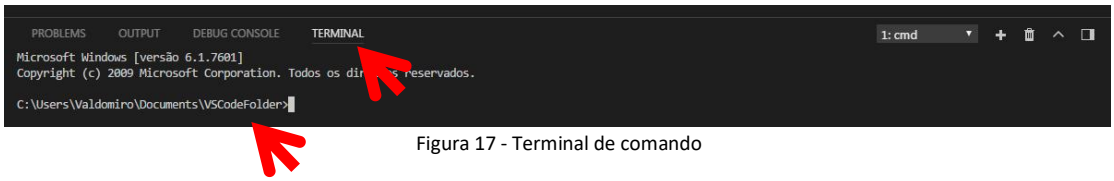

Figura 17 - Terminal de comando

2. Agora vamos definir o nosso terminal como sendo o PowerShell. Clique no sinal de adição **"+"** do lado direito do painel como apontado pela figura 18.

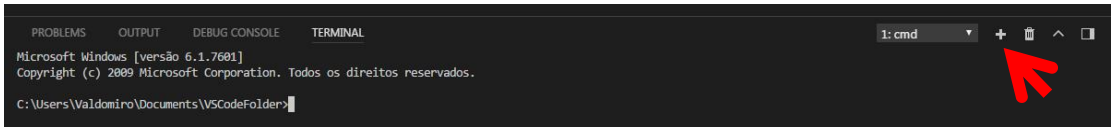

Figura 18 - Adicionando um novo painel

Quando você fizer isto uma pergunta será feita pelo VS Code: "Você pode mudar o terminal padrão selecionando o botão Customize". E é isto que faremos: veja a figura 19

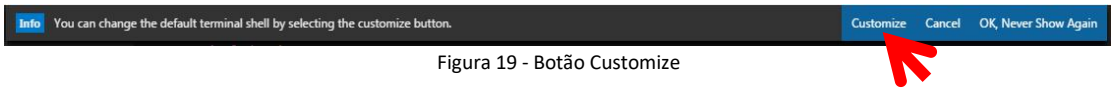

Figura 19 - Botão Customize

Quando você clicar no botão "***Customize***" uma pequena lista de opções de terminal semelhante à da figura 20 aparecerá. Escolha a opção PowerShell como apontado pela figura 20.

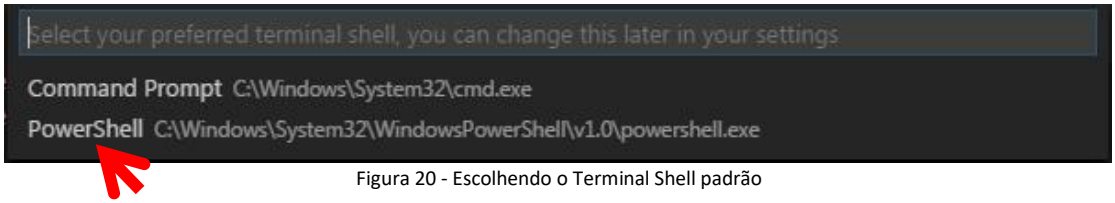

Figura 20 - Escolhendo o Terminal Shell padrão

Se você observar agora no painel, verá que o PowerShell está configurado como sendo o terminal padrão. Veja figura 21.

Figura 21 - Novo terminal padrão

Com o PowerShell alguns comandos do Linux (Unix) podem ser utilizados como, por exemplo, ***ls*** para listas o conteúdo do diretório vigente. Experimente!!

B. Configurar as opções de auto completar inteligente.

1. Antes de continuar, feche o VSCode e em seguida reabra-o. Reabra o arquivo **primeiro.cpp**.
2. Digite o seguinte código C++ no arquivo **primeiro.cpp**:

```
#include <iostream>

using namespace std;

int main(){
    for(int i = 0; i < 10; ++i){
        cout << i << endl;
    }
    return 0;
}
```

3. Provavelmente você verá uma linha ondulada abaixo do texto da primeira linha do nosso código. A figura 22 aponta para o que estamos dizendo:

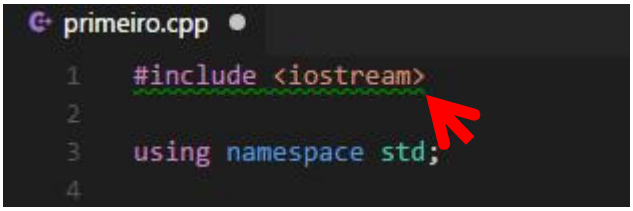

Figura 22 - Indicação de erro

4. Se você aproximar o ponteiro do mouse verá uma dica do tipo de erro encontrado. Veja a figura 23. Este erro informa que o arquivo de cabeçalho não pode ser aberto. Isto se dá porque o VSCode não sabe onde estão os arquivos de cabeçalho do C++ em teu computador. É preciso configurar a variável de ambiente do VSCode chamada **includePath** e informar onde estão os arquivos de cabeçalho.

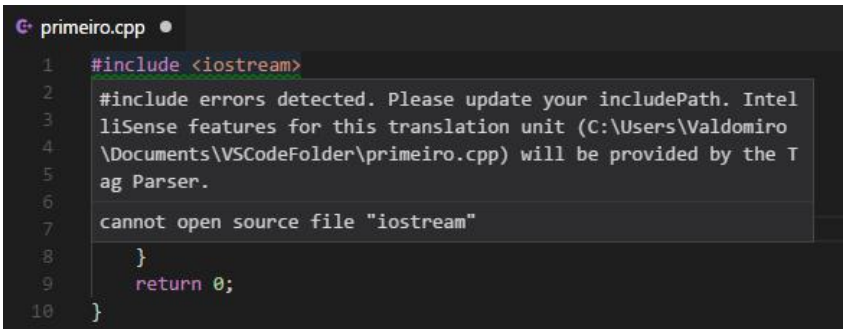

Figura 23 - Dica de erro

5. Dê um clique na primeira linha do nosso código e você verá uma pequena lâmpada aparecer do lado esquerdo como mostra a figura 24.

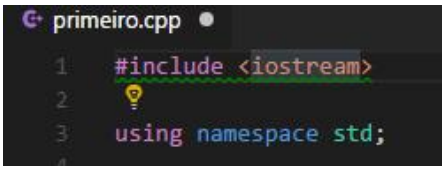

Figura 24 - Lâmpada de sugestão de correção de erro

6. Dando um clique na lâmpada de sugestões de correção de erro aparecerá uma lista de sugestões. Clique em "*Edit "includePath" setting*" (Edite as configurações de "includePath") como aponta a figura 25.

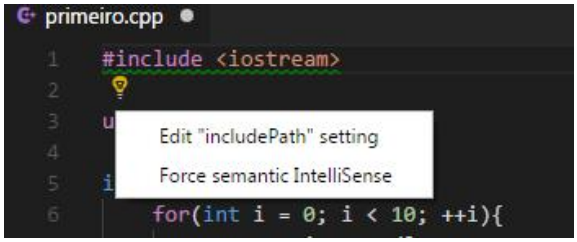

Figura 25 - Edite configurações da "includePath"

7. O arquivo de configurações **c_cpp_properties.json** será aberto no editor. Ele é um arquivo de dados em formato ***JSON***[3] que mantém a configuração das propriedades do ambiente de programação C/C++. Este aquivo contém configurações para Windows, Mac e Linux. Como a nossa proposta é trabalhar em ambiente Windows, procure na lista de configurações as configurações que dizem respeito ao ambiente Windows. Para ambiente Linux, provavelmente a situação atual é a ideal. A figura 26 mostra a situação original destas configurações:

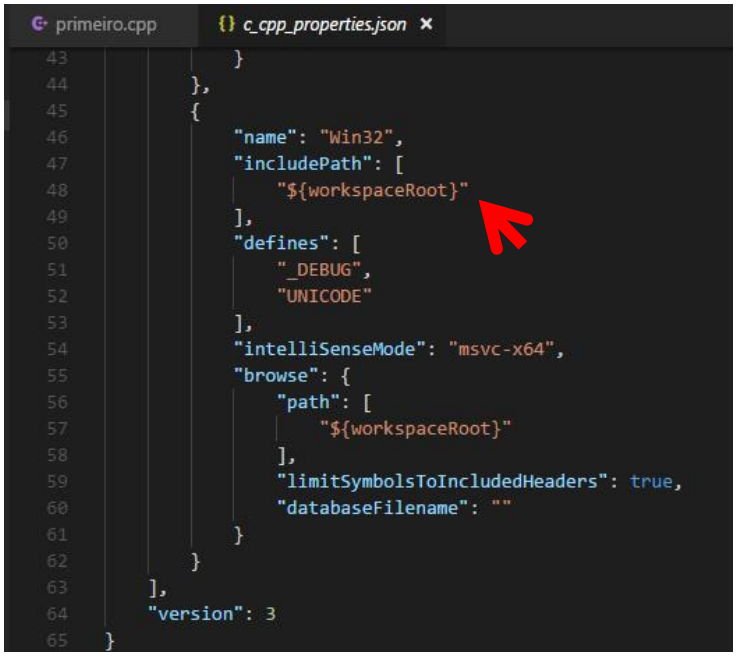

Figura 26 - Configuração do ambiente Windows

8. Para configurar a variável **includePath** alguns passos devem ser seguidos:
   a) Vá ao terminal (ctrl+J) e digite o seguinte comando:
      `g++ -v primeiro.cpp`
   b) Como resultado, aparecerá uma lista muito grande de coisas, mas você deve procurar algo parecido com o conteúdo abaixo. **<MinGW_Path>** é o caminho até onde o teu compilador MinGW está instalado:

```
#include "..." search starts here:
```

[3] Em computação, JSON (com a pronúncia ['dʒejzən], J-son em inglês), um acrônimo para "JavaScript Object Notation", é um formato de padrão aberto que utiliza texto legível a humanos para transmitir objetos de dados consistindo de pares atributo-valor. O JSON é um formato de dados independente de linguagem. Deriva do JavaScript,[1][2] mas a partir de 2017 muitas linguagens de programação incluem código para gerar e analisar sintaticamente dados em formato JSON. O tipo de mídia da Internet oficial para o JSON é application/json. Nomes de arquivos JSON usam a extensão ***.json***.

```
#include <...> search starts here:
<MinGW_Path>/bin/../lib/gcc/i686-w64-mingw32/5.3.0/include
<MinGW_Path>/bin/../lib/gcc/i686-w64-mingw32/5.3.0/include-fixed
<MinGW_Path>/bin/../lib/gcc/i686-w64-mingw32/5.3.0/../../../../i686-w
64-mingw32/include
<MinGW_Path>/lib/gcc/../../i686-w64-mingw32/include/c++
<MinGW_Path>/lib/gcc/../../i686-w64-mingw32/include/c++/i686-w64-ming
w32
<MinGW_Path>/lib/gcc/../../i686-w64-mingw32/include/c++/backward
```

c) Copie todo os caminhos oferecido por esta lista iniciando no primeiro `<MinGW_Path>` até o último selecionando-os e fazendo um **ctrl+C**.

d) Cole-os na variável "**includePath**" após "**${workspaceRoot}**", colando-os entre aspas e separados por vírgulas. A figura 27 mostra um exemplo.

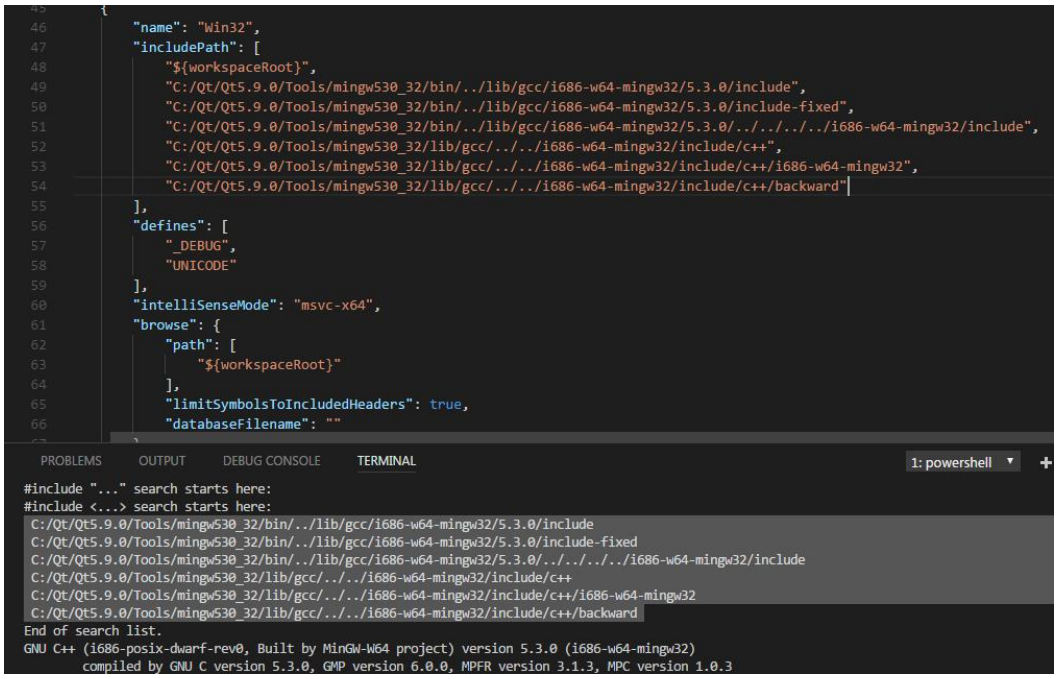

Figura 27 - Exemplo de Configuração da variável includePath

e) Tendo configurado a variável **includePath** copie o conteúdo da variável desta e cole na variável "**browse.path**" como mostra a figura 28.

Figura 28 - Configurando a variável **browse.path**

f) Como último passo vamos vamos configurar o IntelliSense Engine. Vá até o menu File → Preferences → Settings com aponta a figura 29.

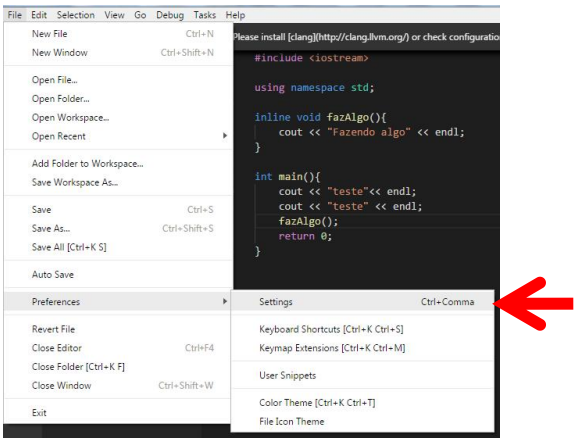

Figura 29 - Menu de configurações de usuário

g) No campo de Busca digite **IntelliSenseEngine** como aponta a figura 30.

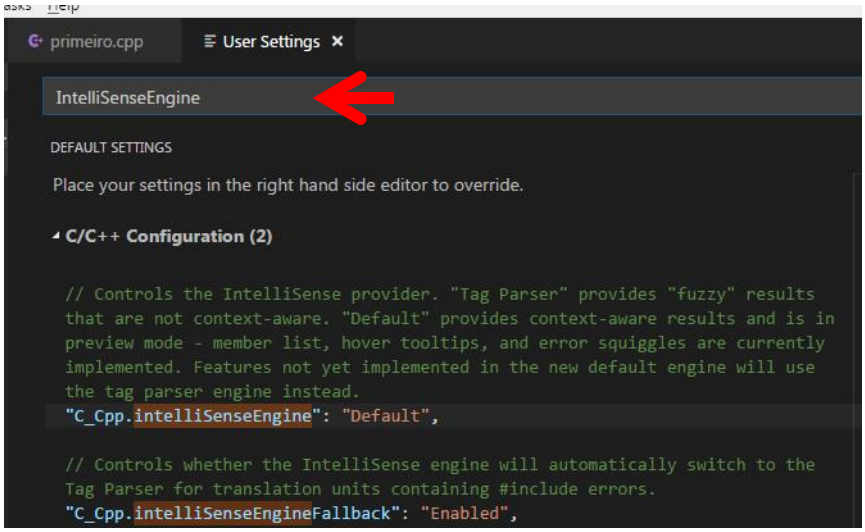

Figura 30 - Procurando a propriedade **intelliSenseEngine**

h) Em seguida, aproxime o mouse da propriedade **C_Cpp.intelliSenseEngine: "Default"**. Do lado esquerdo você verá um lápis de edição. Clique no referido lápis e escolha a opção "**Tag Parser**" como aponta a figura 31.

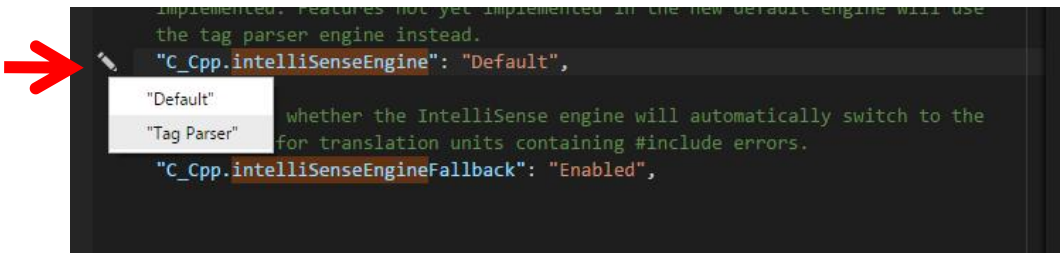

Figura 31 - Alterando a propriedade **intelliSenseEngine**

9. Reinicie o VSCode.

Compilando seu código

Se você deseja compilar sua aplicação a partir do **VS Code**, você terá necessidade de gerar um arquivo tasks.json:

1. Abra a Paleta de Comando (**Command Palette**) (Ctrl+Shift+P).
2. Selecione o comando **Tasks: Configure Tasks...** (Tarefa: Configurar Tarefas...), clique em **Create tasks.json from templates**, (Criar **tasks.json** a partir de modelo), e você irá ver uma lista de modelos de tarefas.
3. Selecione **Others** (Outros) para criar uma tarefa que rode um comando externo.
4. Escolha o **command** (comando) para a expressão de linha de comando que você usa para criar tua aplicação (por exemplo g++).
5. Adicione qualquer argumento necessário (por exemplo **-g** para construir para depuração)
6. Você pode também mudar a **label** para ser mais descritivo.

Você deverá agora ver um arquivo **tasks.json** em seu workspace **.vscode** que se assemelha a este:

```
{
    "version": "2.0.0",
    "tasks": [
        {
            "label": "build hello world",
            "type": "shell",
            "command": "g++",
            "args": [
                "-g", "helloworld.cpp"
            ]
        }
    ]
}
```

Se você quiser construir seu aplicativo com **Tasks: Run Build Task** (Ctrl+Shift+B) (Tarefas: Executar tarefa de compilação), você pode adicioná-lo ao grupo **build**.

```
{
    "version": "2.0.0",
    "tasks": [
        {
            "label": "build hello world",
            "type": "shell",
            "command": "g++",
            "args": [
                "-g", "helloworld.cpp"
            ],
            "group": {
                "kind": "build",
                "isDefault": true
            }
        }
    ]
}
```

Alterar o terminal integrado

1. Clique em File →Preferences →Settings (Ctrl+ ‘,’) ou clique no ícone engrenagem no canto inferior esquerdo da tela principal.
2. No campo de pesquisa digite **terminal.integrated.shell**, em seguida procure a opção para o teu sistema operacional. Aproxime o mouse e clique no lápis de edição à esquerda. Depois clique em

"**Replace in Settings**" como aponta a figura 32 para adicional esta configuração à configurações de usuário. (Vamos fazer para o Windows).

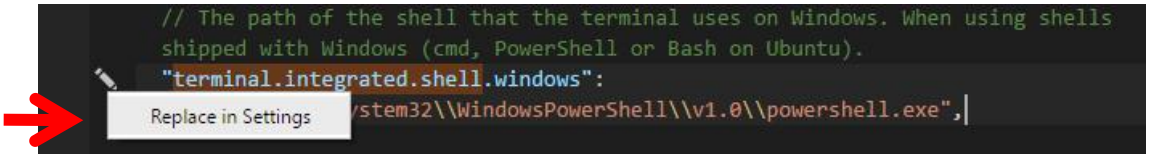

Figura 32 - Alterando o terminal integrado

3. Nas configurações de usuário, informe o caminho do terminal que você gostaria de usar. No nosso caso estamos usando o PowerShell, mas um excelente escolha seria o bash do git como mostra a figura 33.

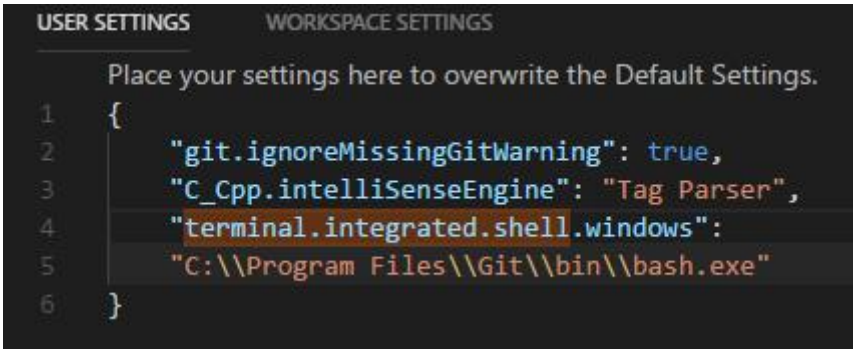

Figura 33 - Terminal do Git