

UTILIZANDO O CODE BLOCKS

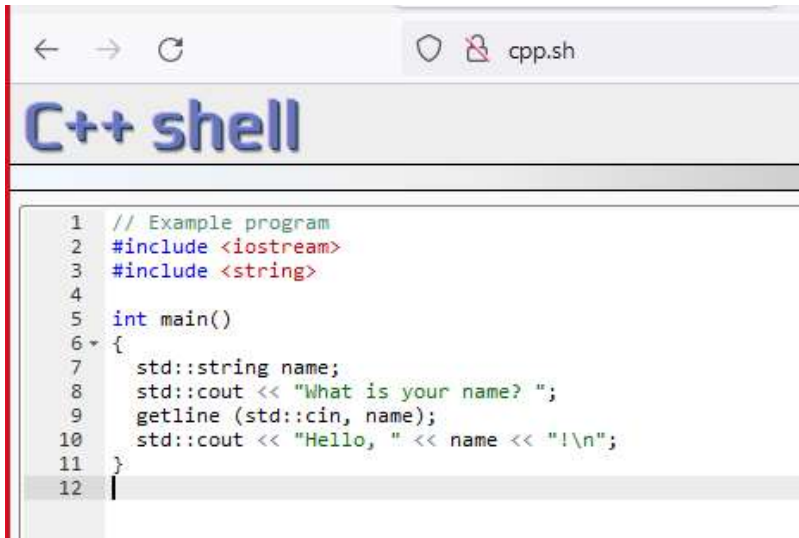
Prof. Humberto Razente

Sala 1B144

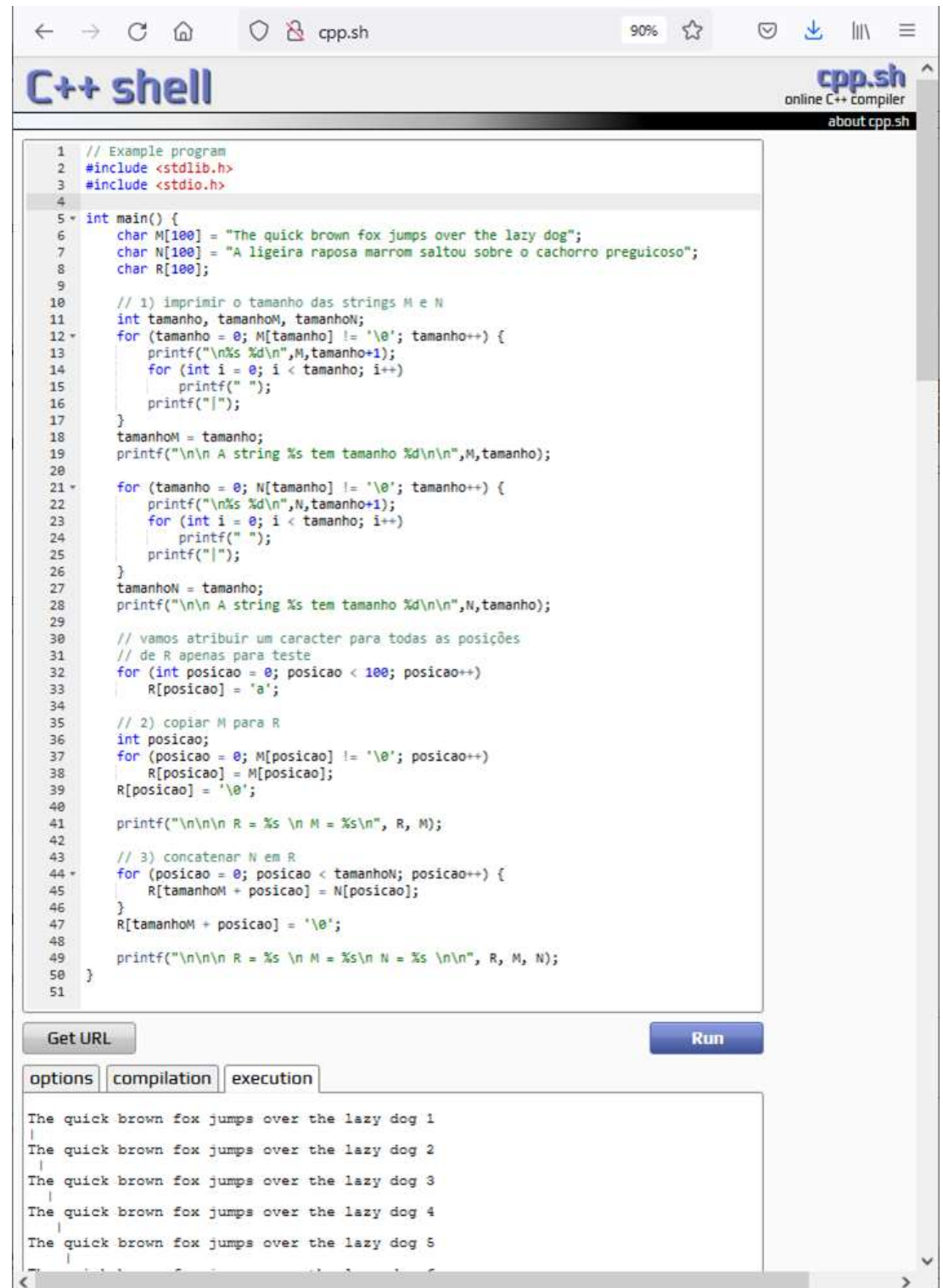
INTRODUÇÃO

- Existem diversos ambientes de desenvolvimento integrado ou IDEs (Integrated Development Environment) que podem ser utilizados para a programação em linguagem C.
- Um deles é o Code::Blocks, uma IDE de código aberto e multiplataforma que suporta múltiplos compiladores.
 - O Code::Blocks pode ser baixado diretamente de seu site www.codeblocks.org
 - Procure baixar a versão que inclui tanto a IDE do Code::Blocks como o compilador GCC e o debugger GDB da MinGW

ONLINE



```
1 // Example program
2 #include <iostream>
3 #include <string>
4
5 int main()
6 {
7     std::string name;
8     std::cout << "What is your name? ";
9     getline (std::cin, name);
10    std::cout << "Hello, " << name << "! \n";
11 }
12
```



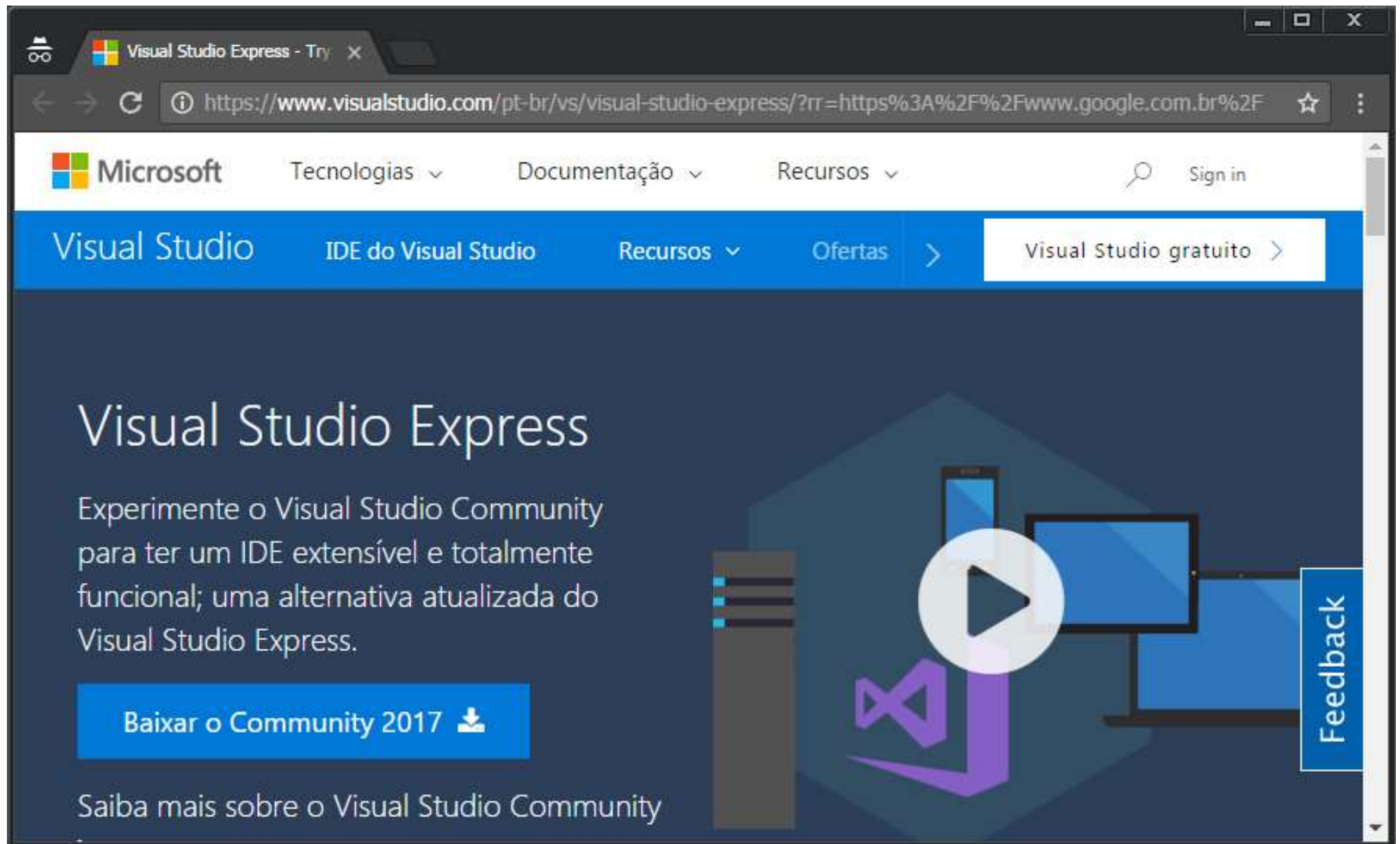
```
1 // Example program
2 #include <stdlib.h>
3 #include <stdio.h>
4
5 int main() {
6     char M[100] = "The quick brown fox jumps over the lazy dog";
7     char N[100] = "A ligeira raposa marrom saltou sobre o cachorro preguiçoso";
8     char R[100];
9
10    // 1) imprimir o tamanho das strings M e N
11    int tamanho, tamanhoM, tamanhoN;
12    for (tamanho = 0; M[tamanho] != '\0'; tamanho++) {
13        printf("\n%s %d\n", M, tamanho+1);
14        for (int i = 0; i < tamanho; i++)
15            printf(" ");
16        printf("|");
17    }
18    tamanhoM = tamanho;
19    printf("\n\n A string %s tem tamanho %d\n\n", M, tamanho);
20
21    for (tamanho = 0; N[tamanho] != '\0'; tamanho++) {
22        printf("\n%s %d\n", N, tamanho+1);
23        for (int i = 0; i < tamanho; i++)
24            printf(" ");
25        printf("|");
26    }
27    tamanhoN = tamanho;
28    printf("\n\n A string %s tem tamanho %d\n\n", N, tamanho);
29
30    // vamos atribuir um caracter para todas as posições
31    // de R apenas para teste
32    for (int posicao = 0; posicao < 100; posicao++)
33        R[posicao] = 'a';
34
35    // 2) copiar M para R
36    int posicao;
37    for (posicao = 0; M[posicao] != '\0'; posicao++)
38        R[posicao] = M[posicao];
39    R[posicao] = '\0';
40
41    printf("\n\n R = %s \n M = %s\n", R, M);
42
43    // 3) concatenar N em R
44    for (posicao = 0; posicao < tamanhoN; posicao++) {
45        R[tamanhoM + posicao] = N[posicao];
46    }
47    R[tamanhoM + posicao] = '\0';
48
49    printf("\n\n R = %s \n M = %s \n N = %s \n\n", R, M, N);
50 }
51
```

Get URL Run

options compilation execution

```
The quick brown fox jumps over the lazy dog 1
|
The quick brown fox jumps over the lazy dog 2
|
The quick brown fox jumps over the lazy dog 3
|
The quick brown fox jumps over the lazy dog 4
|
The quick brown fox jumps over the lazy dog 5
|
```


OUTRAS IDEs



The image shows a web browser window displaying the Visual Studio Express website. The browser's address bar shows the URL: <https://www.visualstudio.com/pt-br/vs/visual-studio-express/?rr=https%3A%2F%2Fwww.google.com.br%2F>. The page features the Microsoft logo and navigation links for 'Tecnologias', 'Documentação', and 'Recursos'. A blue header bar contains 'Visual Studio', 'IDE do Visual Studio', 'Recursos', 'Ofertas', and a button for 'Visual Studio gratuito'. The main content area has a dark blue background with the text 'Visual Studio Express' and a description: 'Experimente o Visual Studio Community para ter um IDE extensível e totalmente funcional; uma alternativa atualizada do Visual Studio Express.' Below this is a blue button labeled 'Baixar o Community 2017' with a download icon. At the bottom, it says 'Saiba mais sobre o Visual Studio Community'. On the right side, there is a large play button icon over a graphic of a computer setup, and a vertical 'Feedback' button.

Visual Studio Express

Experimente o Visual Studio Community para ter um IDE extensível e totalmente funcional; uma alternativa atualizada do Visual Studio Express.

Baixar o Community 2017

Saiba mais sobre o Visual Studio Community

Feedback

OUTRAS IDEs

Visual Studio Express - Try


Dev-C++ - Download


Secure | <https://bloodshed-dev-c.en.softonic.com>

softonic.com

APPS GAMES FEATURES VIDEOS


Search for apps, articles...

 Google Cloud Platform



US\$ 300 de crédito para experimentar

TESTE GRÁTIS



Dev-C++


beta 9.2 (4.9...

Development software

C/C++


Free Download

Safe download




Community Created Design Studio

Enter the world of C and C++ programming with Bloodshed Dev-C++ a widely used and very efficient editor and compiler in the C and C++ languages. Download it free today and start



1164 votes

Rate it!



We use own and third party cookies to improve our services and your experience. This includes to personalise ads, to provide social media features and to analyse our traffic. We also share information about your use of our site with our social media, advertising and analytics partners. If you continue browsing, you are considered to have accepted such use. You may change your cookie preferences and obtain more information [here](#).

X

OUTRAS IDEs



The image shows a screenshot of the NetBeans website homepage as seen in a web browser. The browser's address bar shows the URL <https://netbeans.org>. The page features a dark blue header with the NetBeans logo and navigation links: NetBeans IDE, NetBeans Platform, Enterprise, Plugins, Docs & Support, and Community. A search bar is located on the right side of the header. The main content area has a blue background with the text "NetBeans IDE Fits the Pieces Together". Below this, it states: "Quickly and easily develop desktop, mobile and web applications with Java, JavaScript, HTML5, PHP, C/C++ and more." and "NetBeans IDE is FREE, open source, and has a worldwide community of users and developers." A large blue starburst graphic with the word "NEW!" is positioned next to the text "NetBeans IDE 8.2". Two orange buttons, "Learn More" and "Download", are located to the right of the starburst. Below the main content area, there is a section titled "Featured News:" with the text "7 February 2017: NetBeans Day France" and a link "See All News". At the bottom, there are three columns of content: "Rapid User Interface Development" with icons for desktop and web, "Write Bug Free Code" with an icon of a bug, and "Powerful Tools for JavaScript, HTML5, and CS" with icons for HTML5 and Java.

Borland C++ Compiler ver: x Welcome to NetBeans x

Secure | <https://netbeans.org>

Choose page language ▶

NetBeans NetBeans IDE NetBeans Platform Enterprise Plugins Docs & Support Community

Search

NetBeans IDE

Fits the Pieces Together

Quickly and easily develop desktop, mobile and web applications with Java, JavaScript, HTML5, PHP, C/C++ and more.

NetBeans IDE is FREE, open source, and has a worldwide community of users and developers.

NEW! **NetBeans IDE 8.2**

Learn More Download

Featured News: 7 February 2017: NetBeans Day France [See All News](#)

Rapid User Interface Development

desktop web

Write Bug Free Code

Powerful Tools for JavaScript, HTML5, and CS

HTML5 Java

OUTRAS IDEs



The screenshot shows the Eclipse website homepage in a web browser. The browser's address bar displays "Secure | https://eclipse.org". The page features the Eclipse logo, a navigation menu with links for "GETTING STARTED", "MEMBERS", "PROJECTS", and "MORE", and a "DOWNLOAD" button. A search bar with "Google Custom Search" is also present. The main content area has a dark blue background with a crowd of people and the text "Eclipse Is..." followed by a description of the community. At the bottom, there are three circular icons representing "IDE & Tools", "Community of Projects", and "Collaborative Working Groups".

Eclipse - The Eclipse Found

Secure | https://eclipse.org

Create account Log in

eclipse

Google Custom Search

DOWNLOAD

GETTING STARTED MEMBERS PROJECTS MORE

Eclipse Is...

An amazing open source community of **Tools, Projects** and **Collaborative Working Groups**. Discover what we have to offer and join us.

DISCOVER

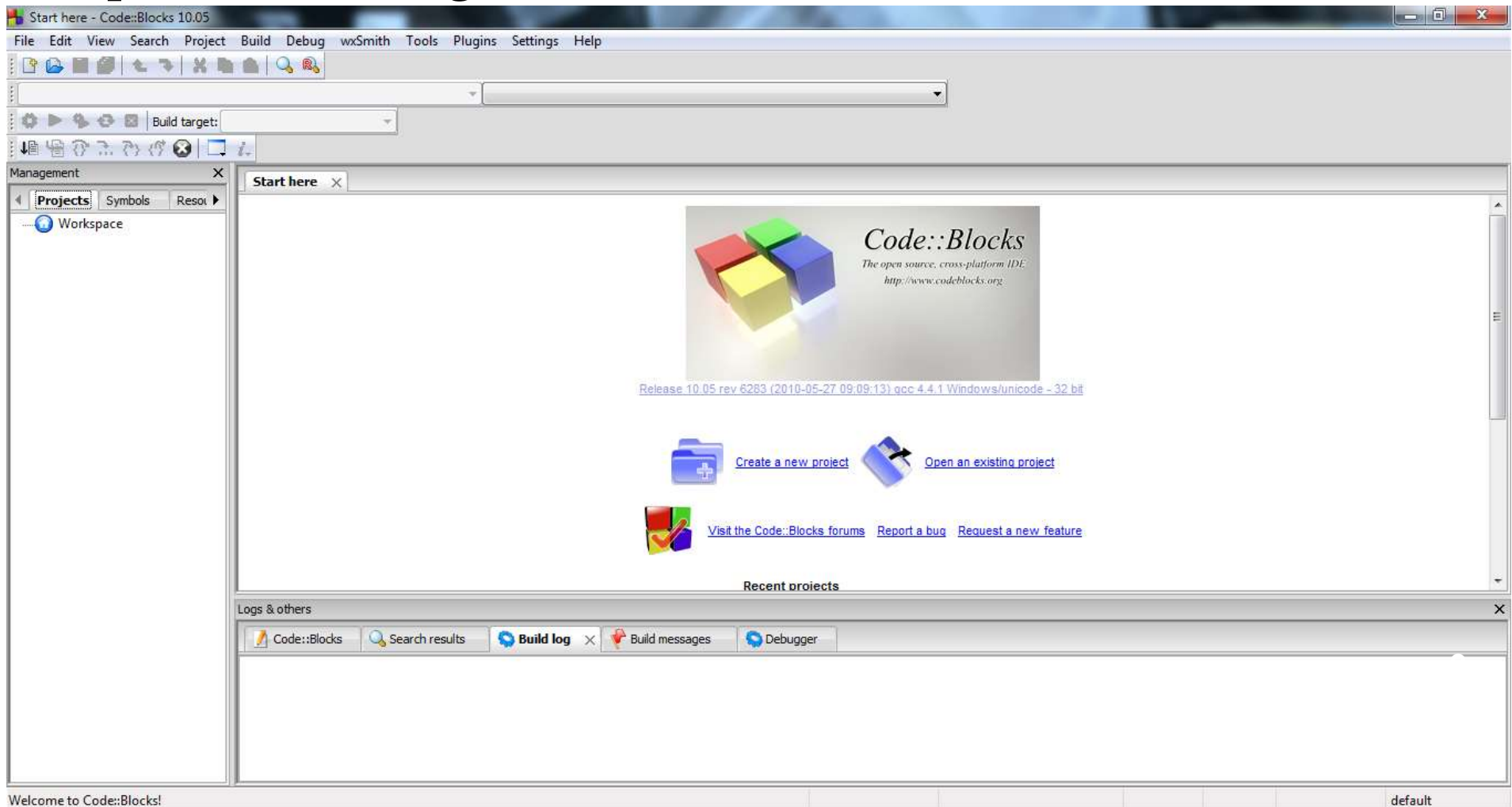
IDE & Tools

Community of Projects

Collaborative Working Groups

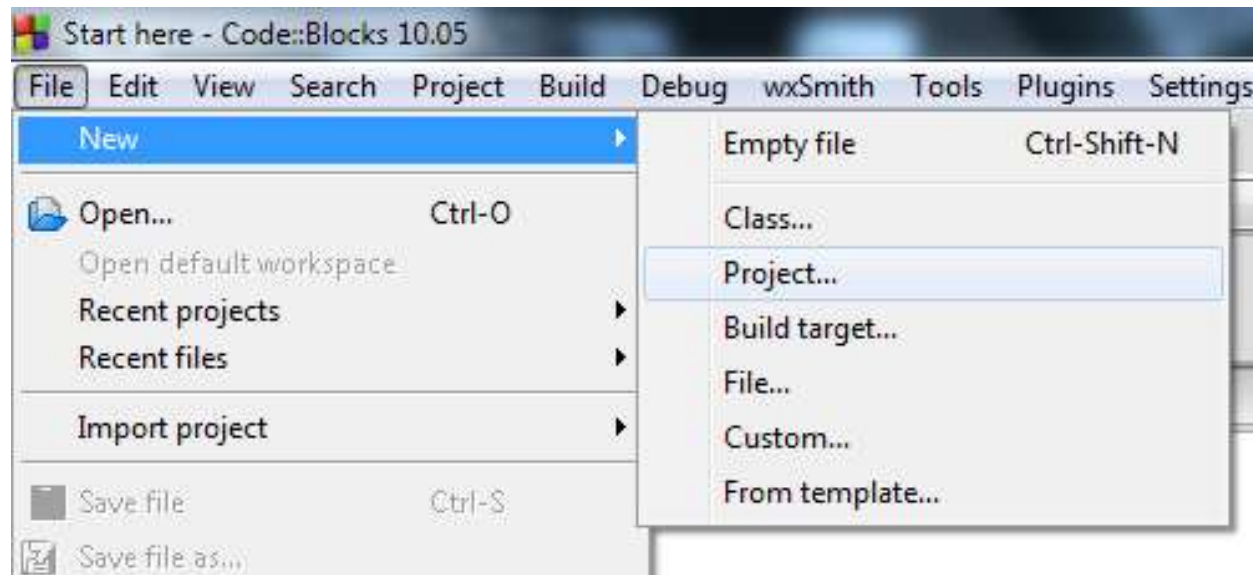
CRIANDO UM NOVO PROJETO NO CODE::BLOCKS

- Primeiramente, inicie o software Code::Blocks. Aparecerá a seguinte tela



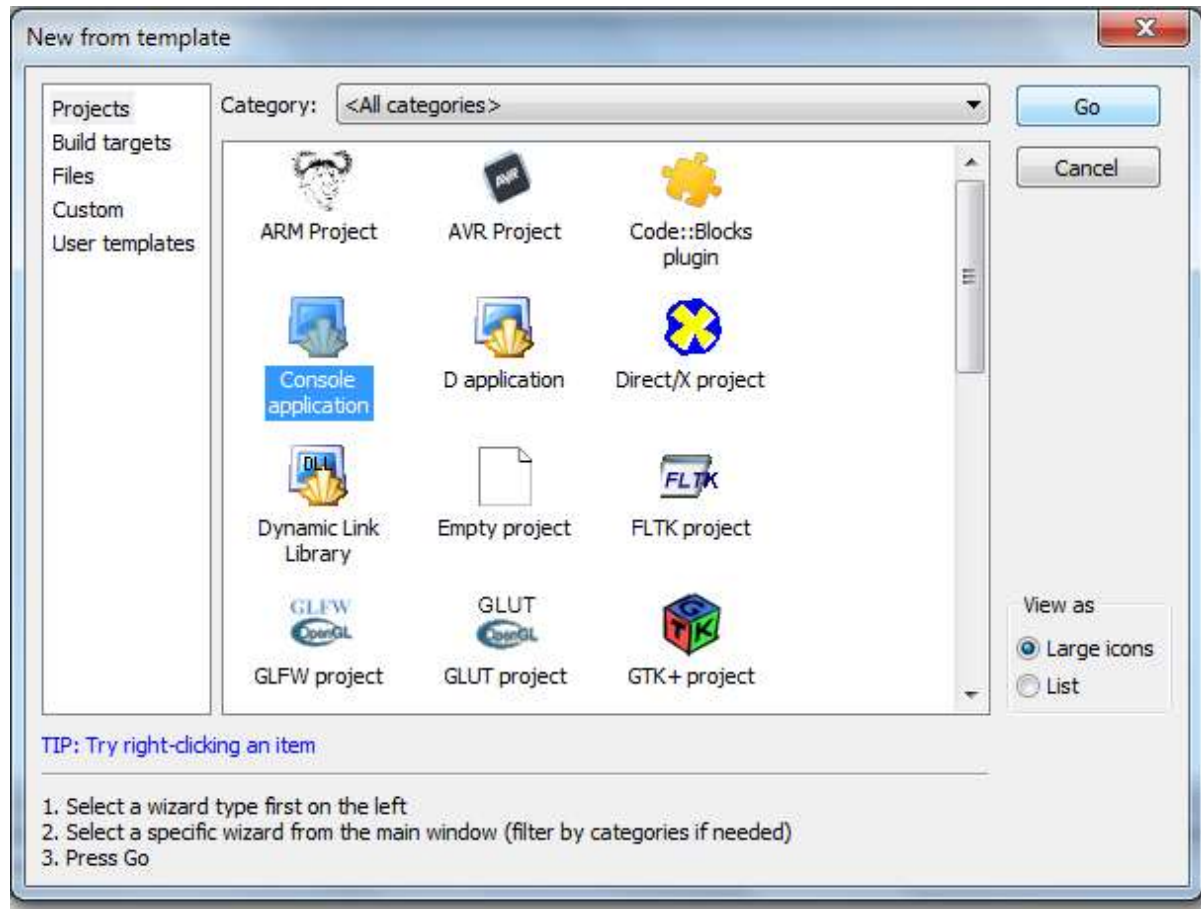
CRIANDO UM NOVO PROJETO NO CODE::BLOCKS

- Em seguida clique em **File**, escolha **New** e depois **Project...**



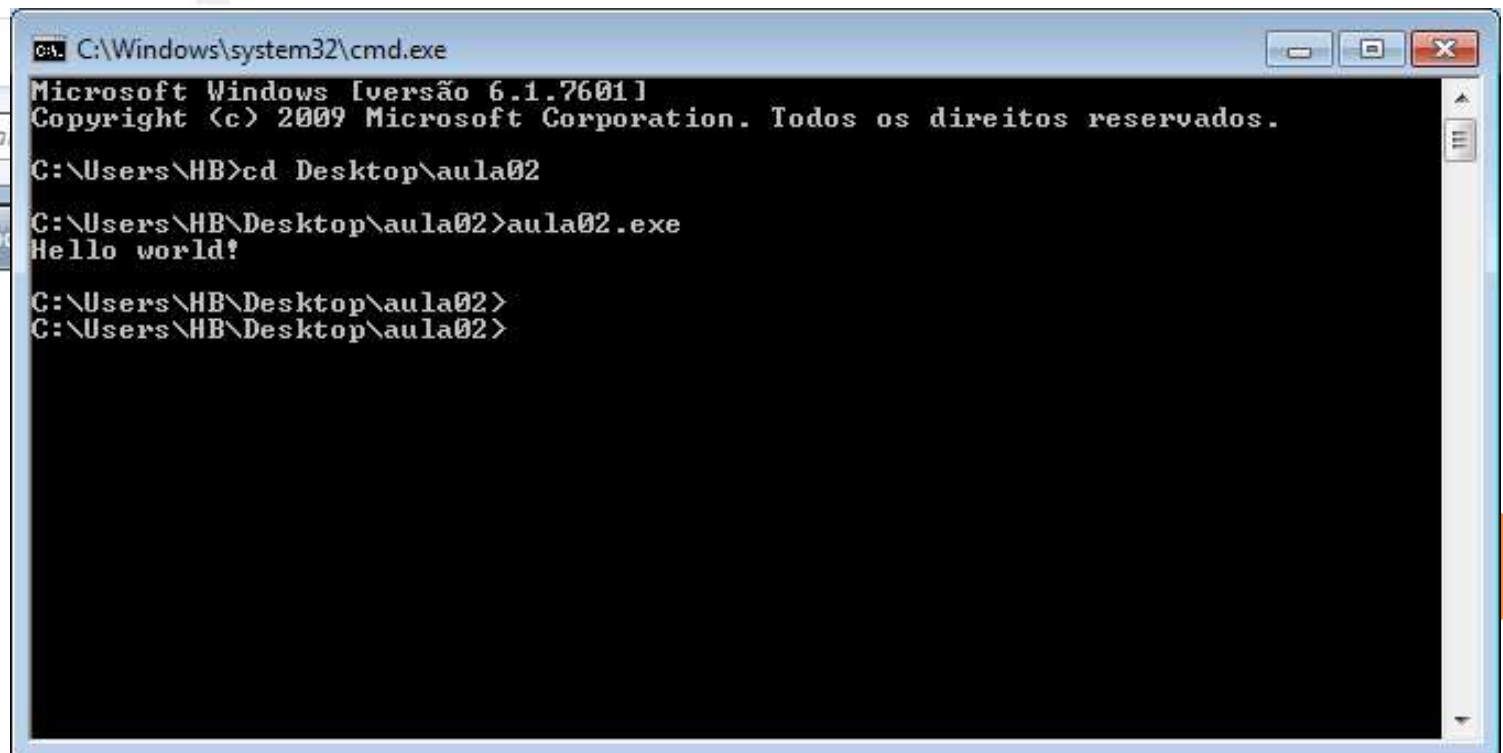
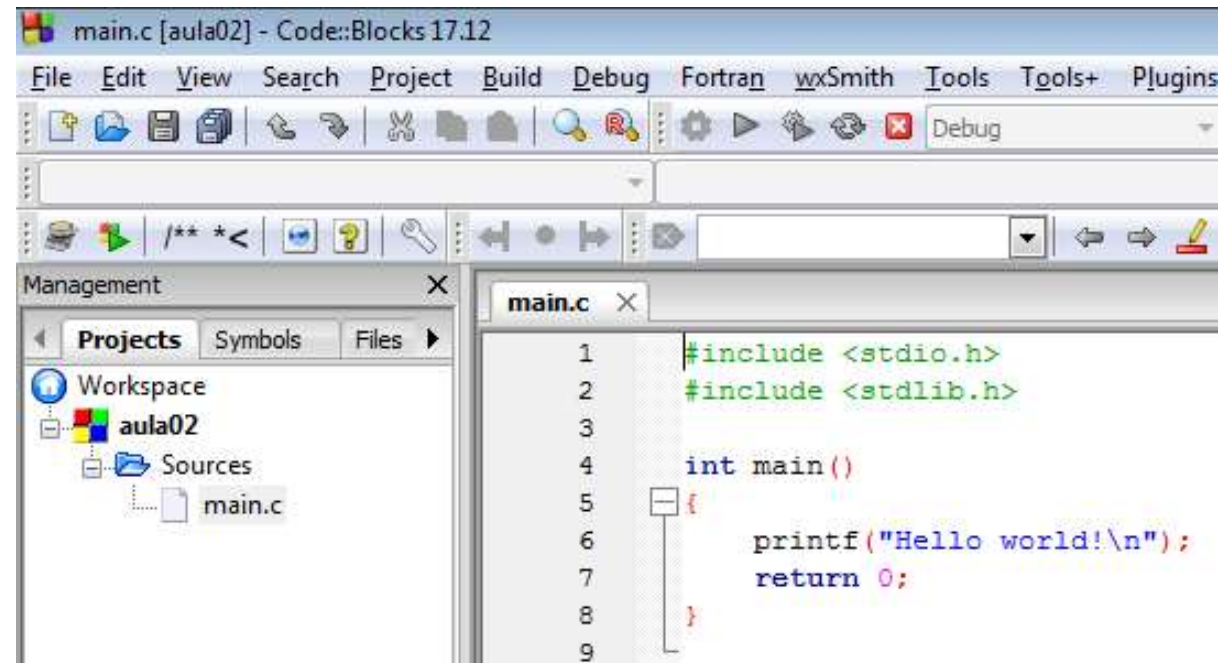
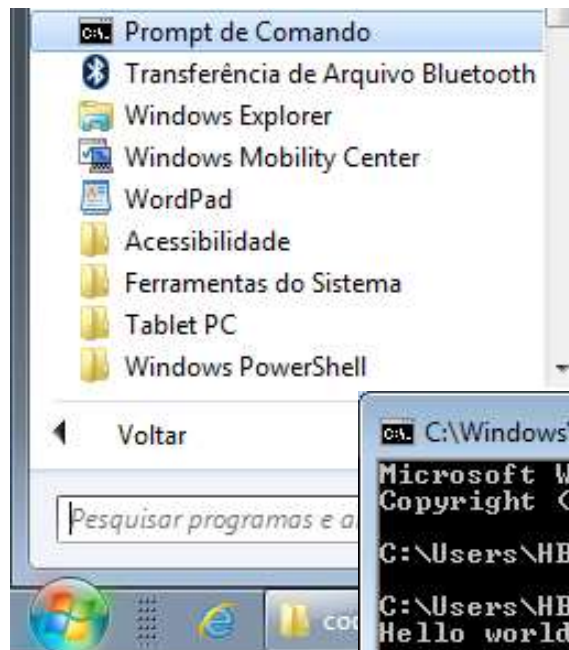
CRIANDO UM NOVO PROJETO NO CODE::BLOCKS

- Uma lista de modelos (templates) de projetos vai aparecer. Escolha **Console application**



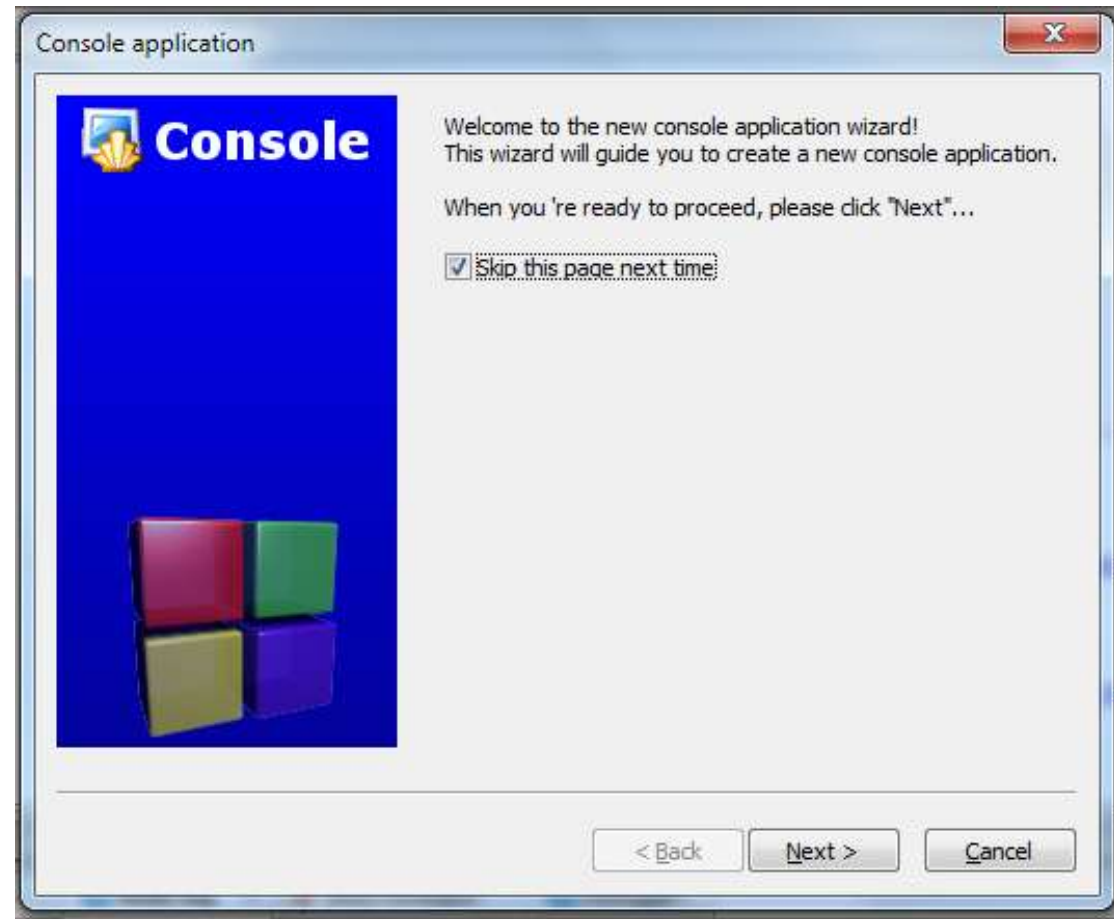
Um Projeto **Console application** gera um programa executável cuja entrada e saída se dá em um terminal de texto (Prompt de comandos)

Um Projeto **Console application** gera um programa executável cuja entrada e saída se dá em um terminal de texto (Prompt de comandos)



CRIANDO UM NOVO PROJETO NO CODE::BLOCKS

- Caso esteja criando um projeto pela primeira vez, a tela a seguir vai aparecer.
 - Se marcarmos a opção **Skip this page next time**, essa tela de boas-vindas não será mais exibida da próxima vez que criarmos um projeto.
- Em seguida, clique em **Next**



CRIANDO UM NOVO PROJETO NO CODE::BLOCKS

- Escolha a opção **C** e clique em **Next**



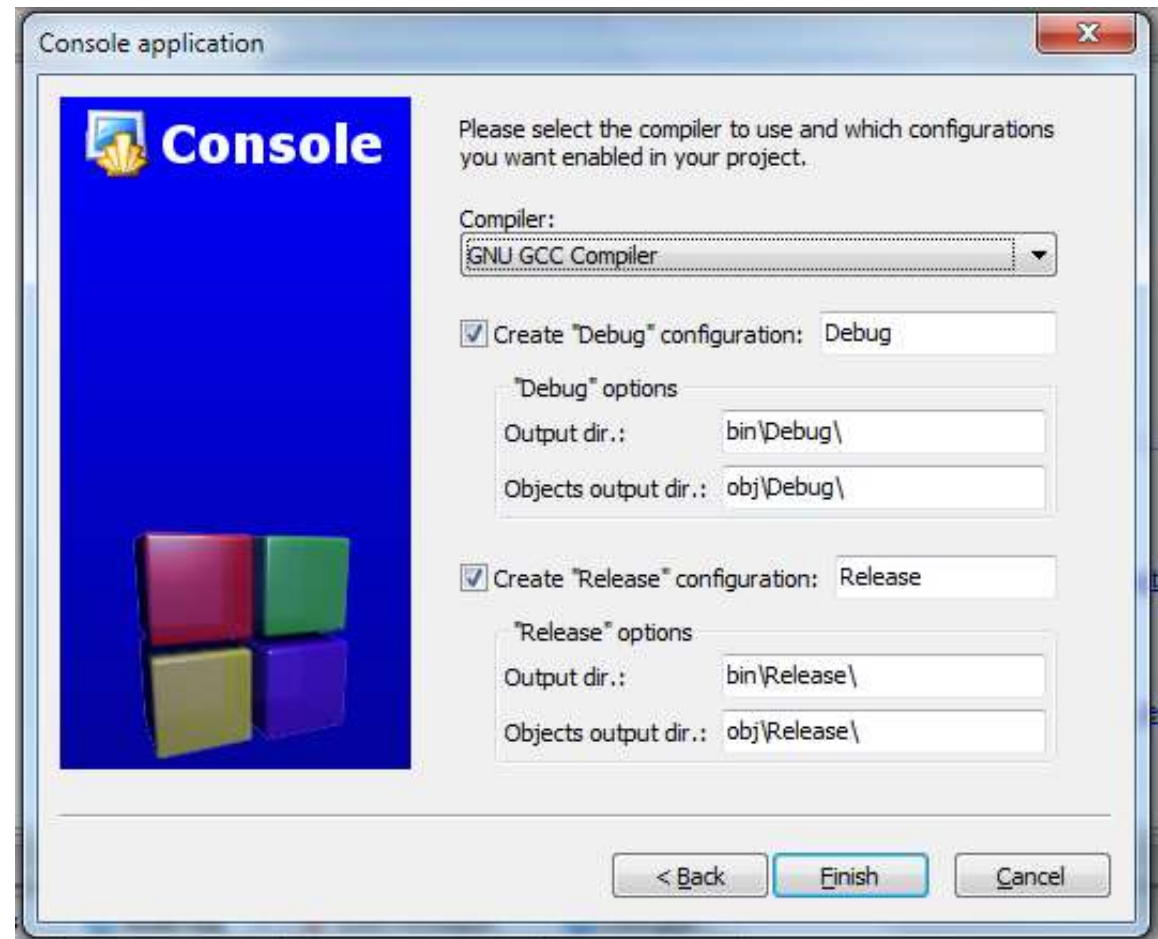
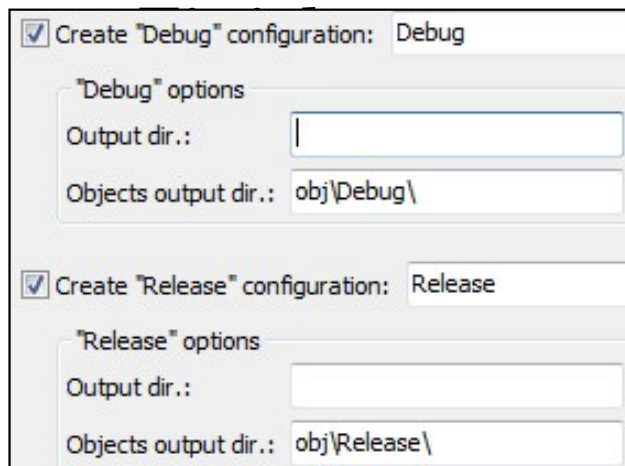
CRIANDO UM NOVO PROJETO NO CODE::BLOCKS

- No campo **Project title**, coloque um nome para o seu projeto. No campo **Folder to create project in** é possível selecionar onde o projeto será salvo no computador.
 - Evite espaços e acentuação no nome e caminho do projeto
- Clique em **Next** para continuar



CRIANDO UM NOVO PROJETO NO CODE::BLOCKS

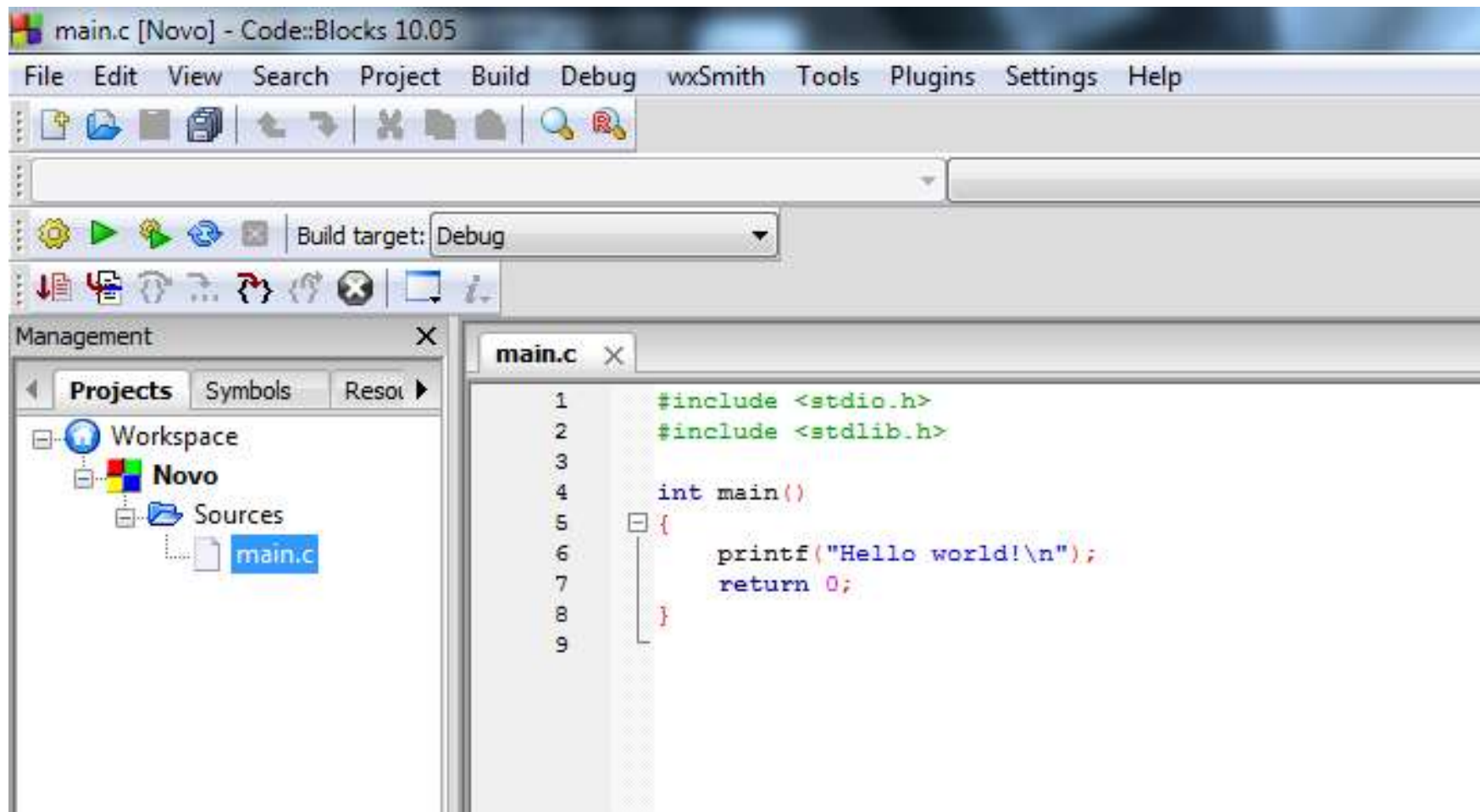
- Na tela a seguir, algumas configurações do compilador podem ser modificadas.
 - No entanto, isso não será necessário.
 - Basta clicar em



Dica: **Output dir** “em branco” gera executável na pasta do projeto!

CRIANDO UM NOVO PROJETO NO CODE::BLOCKS

- Ao fim desses passos, o esqueleto de um novo programa em linguagem C terá sido criado



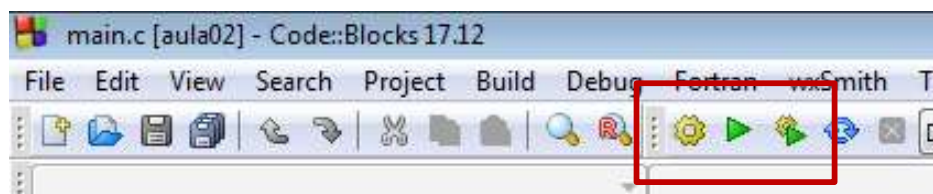
LINGUAGEM C

- Todo programa C inicia sua execução chamando a função `main()`, sendo obrigatória a sua presença no programa principal
- Para incluir bibliotecas de funções, é utilizado uma instrução especial:

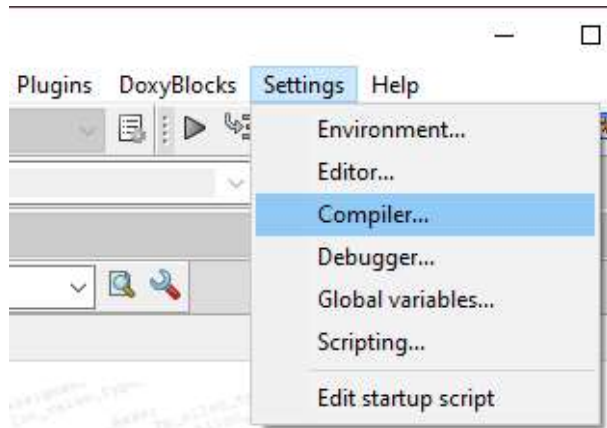
`#include <stdio.h>`

CRIANDO UM NOVO PROJETO NO CODE::BLOCKS

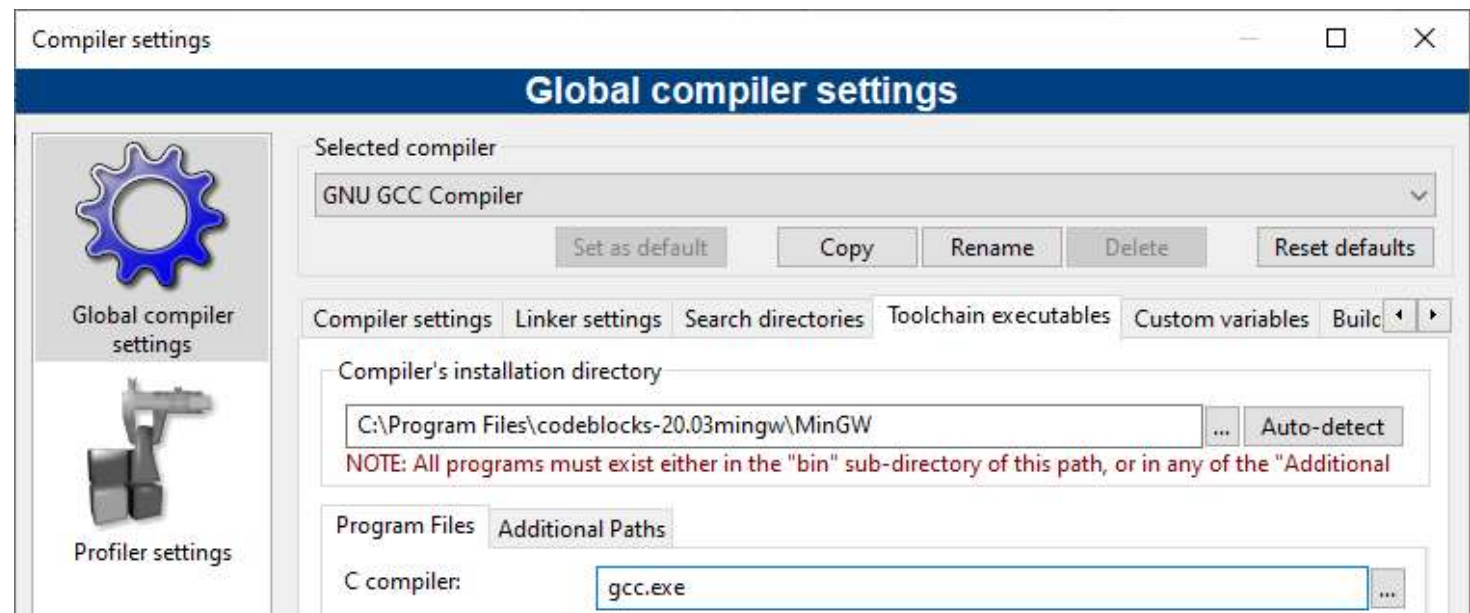
- Podemos utilizar as seguintes opções do menu **Build** para compilar e executar nosso programa
 - **Compile current file (Ctrl+Shift+F9)**
 - essa opção vai transformar seu arquivo de código-fonte em instruções de máquina e gerar um arquivo do tipo objeto.
 - **Build (Ctrl+F9)**
 - serão compilados todos os arquivos do seu projeto para fazer o processo de “linkagem” com tudo o que é necessário para gerar o executável do seu programa.
 - **Build and run (F9)**
 - além de gerar o executável, essa opção também executa o programa gerado.



E SE O CODE::BLOCKS DIZ QUE O COMPILADOR NÃO FOI ENCONTRADO?



- Clique em “Auto-detect”



UTILIZANDO O DEBUGGER DO CODE::BLOCKS

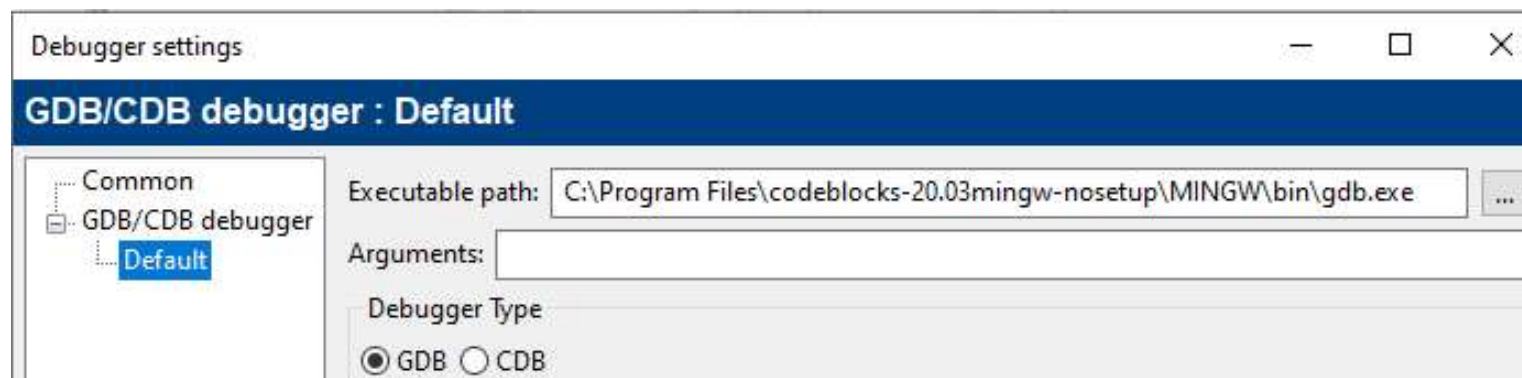
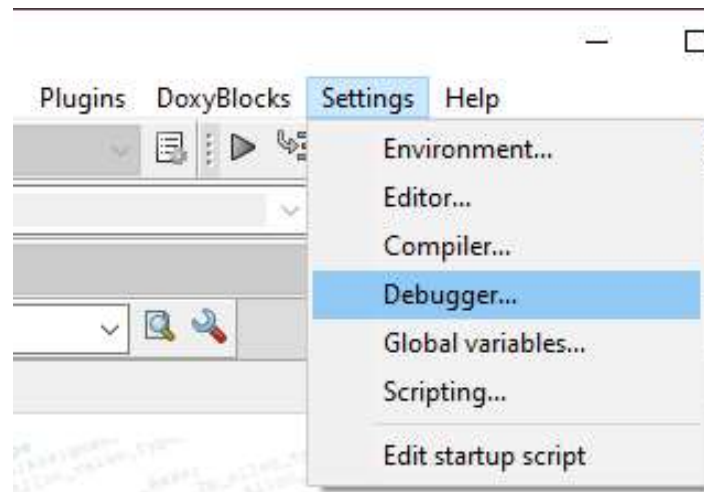
- Com o passar do tempo, nosso conhecimento sobre programação cresce, assim como a complexidade de nossos programas
- Surge então a necessidade de examinar o nosso programa à procura de erros ou defeitos no código-fonte.
- Para realizar essa tarefa, contamos com a ajuda de um **depurador** ou **debugger**.

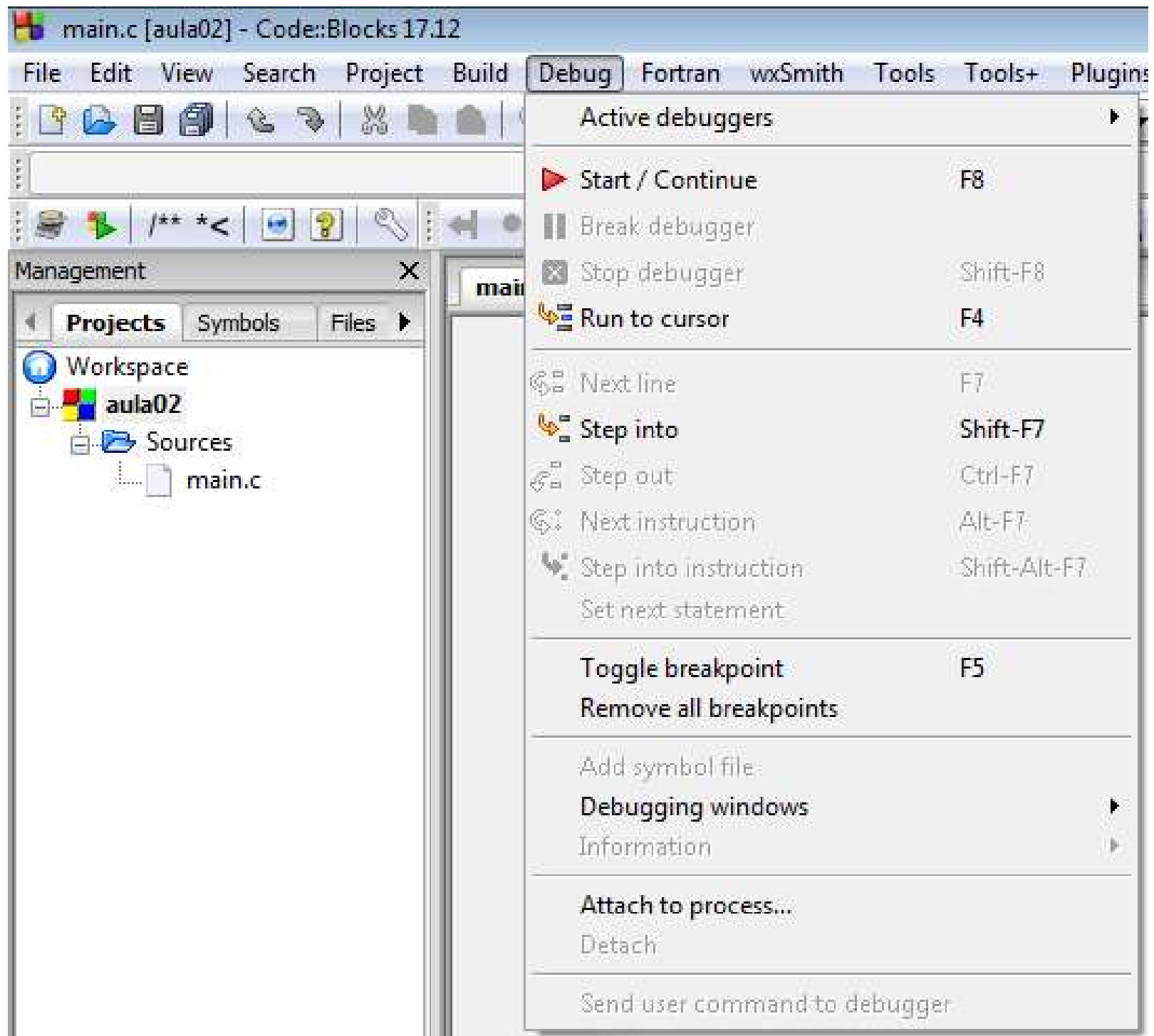
UTILIZANDO O DEBUGGER DO CODE::BLOCKS

- O debugger nada mais é do que um programa de computador usado para testar e depurar (limpar, purificar) outros programas
- Entre as principais funcionalidades de um debugger estão:
 - A possibilidade de executar um programa passo a passo
 - Pausar o programa em pontos predefinidos, chamados **pontos de parada** ou **breakpoints**, para examinar o estado atual de suas variáveis
- Todas as funcionalidades do debugger podem ser encontradas no menu **Debug**

E SE O DEBUGGER DO CODE::BLOCKS NÃO FUNCIONAR?

- Verifique se o gdb.exe listado como debugger leva ao arquivo gdb.exe





UTILIZANDO O DEBUGGER DO CODE::BLOCKS

- Para utilizar o debugger do Code::Blocks, imagine o código ao lado
- Primeiramente, vamos colocar dois pontos de parada ou **breakpoints** no programa, nas linhas 13 e 23
 - Isso pode ser feito clicando no lado direito do número da linha

```
1      #include <stdio.h>
2      #include <stdlib.h>
3      int fatorial(int n){
4          int i,f = 1;
5          for (i = 1; i <= n; i++)
6              f = f * i;
7          return f;
8      }
9      int main(){
10         int x,y;
11         printf("Digite um valor inteiro: ");
12         scanf("%d",&x);
13         if (x > 0){
14             printf("X eh positivo\n");
15             y = fatorial(x);
16             printf("Fatorial de X eh %d\n",y);
17         }else{
18             if (x < 0)
19                 printf("X eh negativo\n");
20             else
21                 printf("X eh Zero\n");
22         }
23         printf("Fim do programa!\n");
24         system("pause");
25         return 0;
26     }
27
```

UTILIZANDO O DEBUGGER DO CODE::BLOCKS

- Iniciamos o debugger com a opção **Start (F8)**.
 - Isso fará com que o programa seja executado normalmente até encontrar um breakpoint.

```
1      #include <stdio.h>
2      #include <stdlib.h>
3      int fatorial(int n){
4          int i,f = 1;
5          for (i = 1; i <= n; i++)
6              f = f * i;
7          return f;
8      }
9      int main(){
10         int x,y;
11         printf("Digite um valor inteiro: ");
12         scanf("%d",&x);
13         if (x > 0){
14             printf("X eh positivo\n");
15             y = fatorial(x);
16             printf("Fatorial de X eh %d\n",y);
17         }else{
18             if (x < 0)
19                 printf("X eh negativo\n");
20             else
21                 printf("X eh Zero\n");
22         }
23         printf("Fim do programa!\n");
24         system("pause");
25         return 0;
26     }
27
```

UTILIZANDO O DEBUGGER DO CODE::BLOCKS

- No nosso exemplo, o usuário deverá digitar, no console, o valor lido pelo comando **scanf()** e depois retornar para a tela do **Code::Blocks** onde o programa se encontra pausado

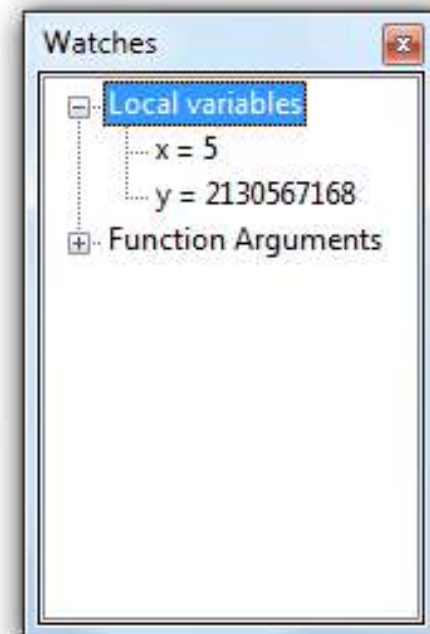
- Note que existe um **triângulo amarelo** dentro do primeiro **breakpoint**.
- Esse triângulo indica em que parte do programa a pausa está

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  int fatorial(int n){
4      int i, f = 1;
5      for (i = 1; i <= n; i++)
6          f = f * i;
7      return f;
8  }
9  int main(){
10     int x, y;
11     printf("Digite um valor inteiro: ");
12     scanf("%d", &x);
13     if (x > 0){
14         printf("X eh positivo\n");
15         y = fatorial(x);
16         printf("Fatorial de X eh %d\n", y);
17     }else{
18         if (x < 0)
19             printf("X eh negativo\n");
20         else
21             printf("X eh Zero\n");
22     }
23     printf("Fim do programa!\n");
24     system("pause");
25     return 0;
26 }
27
```

UTILIZANDO O DEBUGGER DO CODE::BLOCKS

- Dentro da opção **Debugging windows**, podemos habilitar a opção **Watches**
 - Essa opção vai abrir uma pequena janela que permite ver o valor atual das variáveis de um programa, assim como o valor passado para funções.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  int fatorial(int n){
4      int i, f = 1;
5      for (i = 1; i <= n; i++)
6          f = f * i;
7      return f;
8  }
9  int main(){
10     int x, y;
11     printf("Digite um valor inteiro: ");
12     scanf("%d", &x);
13     if (x > 0){
14         printf("X eh positivo\n");
15         y = fatorial(x);
16         printf("Fatorial de X eh %d\n", y);
17     }else{
18         if (x < 0)
19             printf("X eh negativo\n");
20         else
21             printf("X eh Zero\n");
22     }
23     printf("Fim do programa!\n");
24     system("pause");
25     return 0;
26 }
27
```



UTILIZANDO O DEBUGGER DO CODE::BLOCKS

- A partir de determinado ponto de pausa do programa, podemos nos mover para a próxima linha do programa com a opção **Next line (F7)**.
- Essa opção faz com que o programa seja executado passo a passo, sempre avançando para a linha seguinte do escopo onde estamos.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  int fatorial(int n){
4      int i, f = 1;
5      for (i = 1; i <= n; i++)
6          f = f * i;
7      return f;
8  }
9  int main(){
10     int x, y;
11     printf("Digite um valor inteiro: ");
12     scanf("%d", &x);
13     if (x > 0){
14         printf("X eh positivo\n");
15         y = fatorial(x);
16         printf("Fatorial de X eh %d\n", y);
17     }else{
18         if (x < 0)
19             printf("X eh negativo\n");
20         else
21             printf("X eh Zero\n");
22     }
23     printf("Fim do programa!\n");
24     system("pause");
25     return 0;
26 }
27
```


UTILIZANDO O DEBUGGER DO CODE::BLOCKS

- Se houver uma chamada de função (linha 15) a opção **Next line (F7)** chama a função, mas não permite que a estudemos passo a passo.
- Para entrar dentro do código de uma função, utilizamos a opção **Step into (Shift+F7)** na linha da chamada da função.
- Nesse caso, o triângulo amarelo que marca onde estamos no código vai para a primeira linha do código da função

```
1      #include <stdio.h>
2      #include <stdlib.h>
3      int fatorial(int n){
4          int i,f = 1;
5          for (i = 1; i <= n; i++)
6              f = f * i;
7          return f;
8      }
9      int main(){
10         int x,y;
11         printf("Digite um valor inteiro: ");
12         scanf("%d",&x);
13         if (x > 0){
14             printf("X eh positivo\n");
15             y = fatorial(x);
16             printf("Fatorial de X eh %d\n",y);
17         }else{
18             if (x < 0)
19                 printf("X eh negativo\n");
20             else
21                 printf("X eh Zero\n");
22         }
23         printf("Fim do programa!\n");
24         system("pause");
25         return 0;
26     }
27
```

UTILIZANDO O DEBUGGER DO CODE::BLOCKS

- Uma vez dentro de uma função, podemos percorrê-la passo a passo com a opção **Next line (F7)**.
 - Terminada a função, o **debugger** vai para a linha seguinte ao ponto do código que chamou a função (linha 16).
 - Caso queiramos ignorar o resto da função e voltar para onde estávamos no código que chamou a função, basta clicar na opção **Step out (Shift+Ctrl+F7)**.

```
1      #include <stdio.h>
2      #include <stdlib.h>
3      int fatorial(int n){
4          int i, f = 1;
5          for (i = 1; i <= n; i++)
6              f = f * i;
7          return f;
8      }
9      int main(){
10         int x, y;
11         printf("Digite um valor inteiro: ");
12         scanf("%d", &x);
13         if (x > 0){
14             printf("X eh positivo\n");
15             y = fatorial(x);
16             printf("Fatorial de X eh %d\n", y);
17         }else{
18             if (x < 0)
19                 printf("X eh negativo\n");
20             else
21                 printf("X eh Zero\n");
22         }
23         printf("Fim do programa!\n");
24         system("pause");
25         return 0;
26     }
27
```

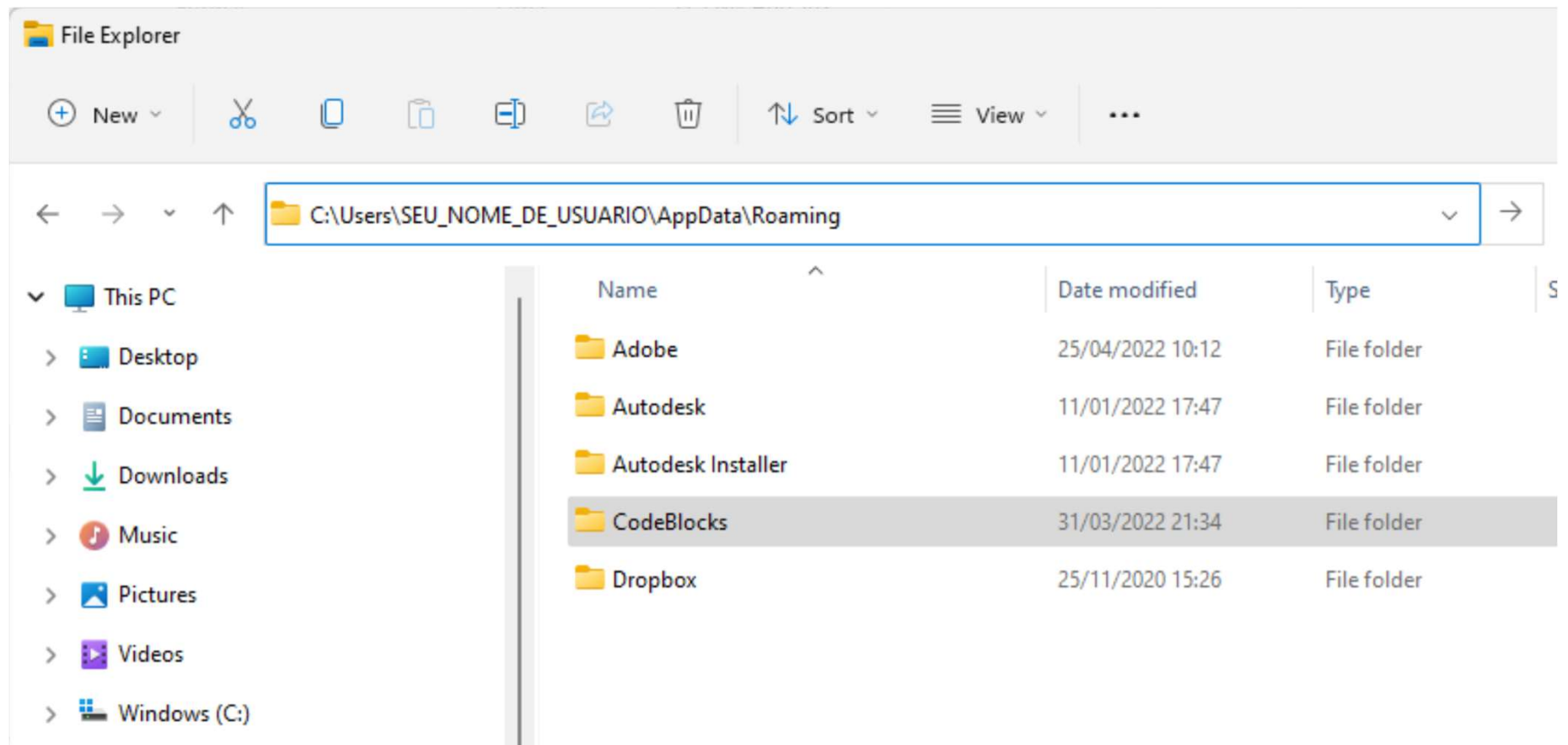
UTILIZANDO O DEBUGGER DO CODE::BLOCKS

- Para avançar todo o código e ir direto para o próximo **breakpoint** (linha 23), podemos usar a opção **Continue (Ctrl+F7)**.
- Por fim, para parar o **debugger**, basta clicar na opção **Stop debugger**

```
1      #include <stdio.h>
2      #include <stdlib.h>
3      int fatorial(int n){
4          int i, f = 1;
5          for (i = 1; i <= n; i++)
6              f = f * i;
7          return f;
8      }
9      int main(){
10         int x, y;
11         printf("Digite um valor inteiro: ");
12         scanf("%d", &x);
13         if (x > 0){
14             printf("X eh positivo\n");
15             y = fatorial(x);
16             printf("Fatorial de X eh %d\n", y);
17         }else{
18             if (x < 0)
19                 printf("X eh negativo\n");
20             else
21                 printf("X eh Zero\n");
22         }
23         printf("Fim do programa!\n");
24         system("pause");
25         return 0;
26     }
27
```

MEU CODEBLOCKS ESTÁ ESTRANHO: MAS SE EU REMOVO A INSTALAÇÃO E INSTALO NOVAMENTE, CONTINUA DO MESMO MODO; O QUE FAZER?

- Apague o diretório de configuração do CodeBlocks do seu perfil



UTILIZANDO O CODE BLOCKS

Contém slides originais gentilmente disponibilizados pelo Prof. André R. Backes (UFU)