

1. 文档File → 开新New → 文档Empty File (再另存新档x:\coding\hello.c)
2. File → New → File → C/C++Source : x:\coding\hello.c
3. 文件名 : x:\coding\hello.c 新文件名 (不要使用中文或者含空格的目录/文件名)

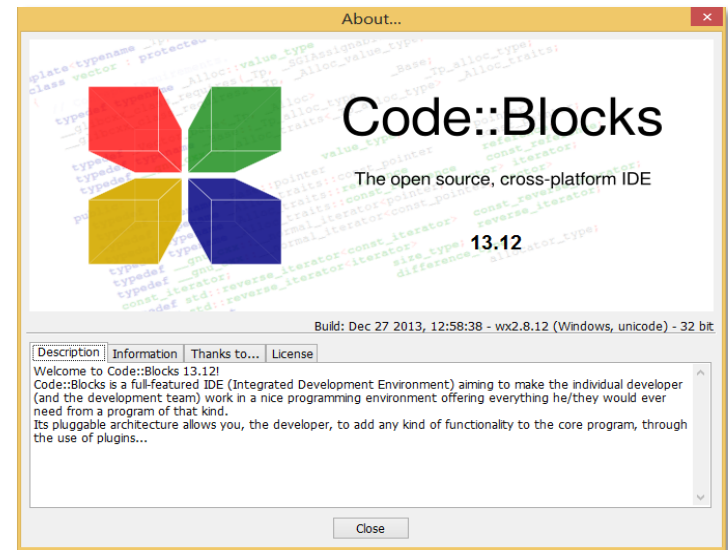
4. 按Ctrl-J (Auto Complete) 缩写替换
5. 编辑Edit → Complete Code所有程序关键字

6. 工具栏Tool Bar : 编译 & 执行
7. 构建Build > Build编译 & Run执行 (F9)
8. 插件Plugins > AStyle代码格式化 (indent)

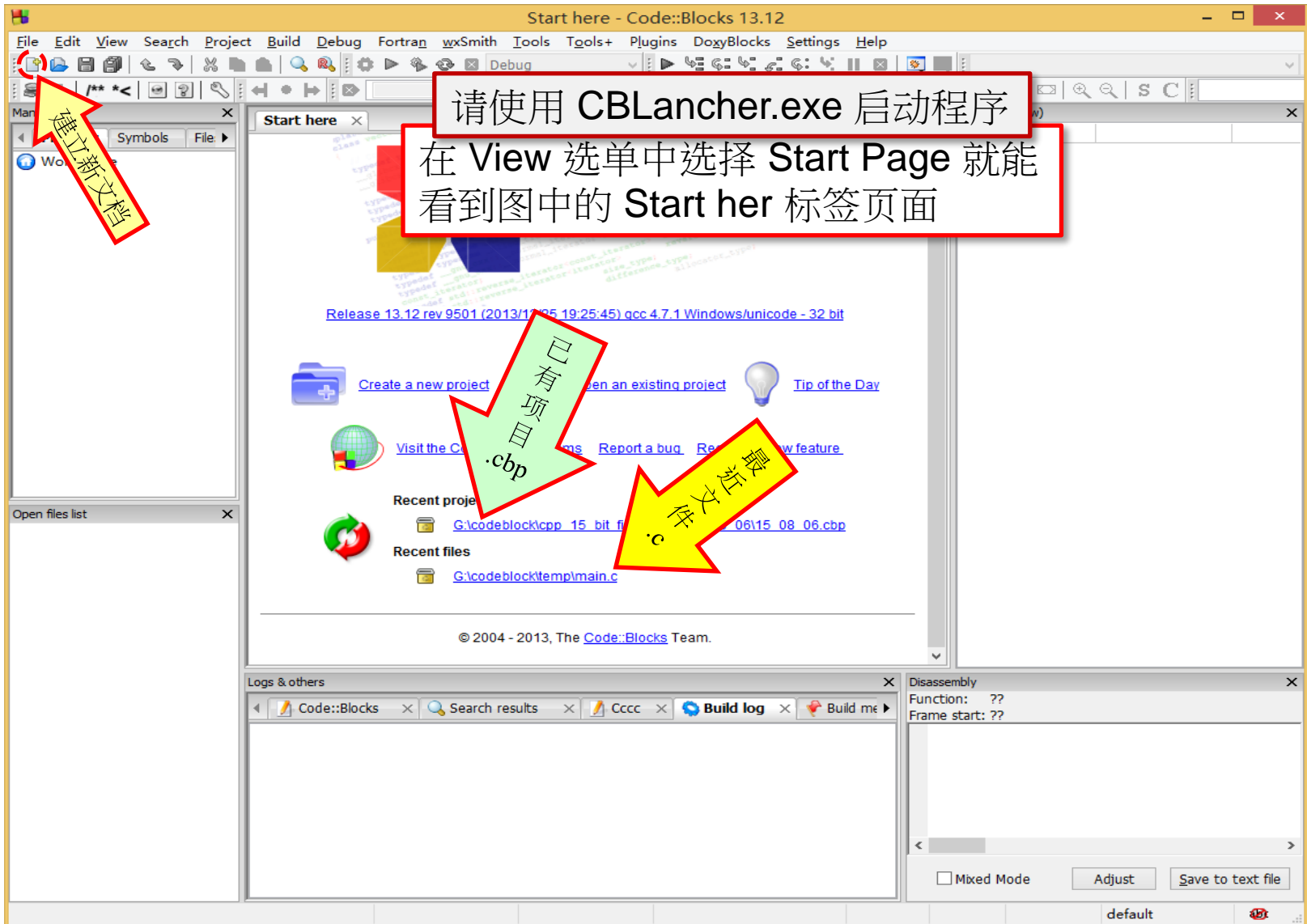
9. 设定Settings > Editor编辑器
10. 查找Find、替换Replace
11. 视图View→工具栏Toolbars
12. 配置文件:

CodeBlocks目录\AppData\codeblocks\default.conf (绿色便携版的配置文件位置)

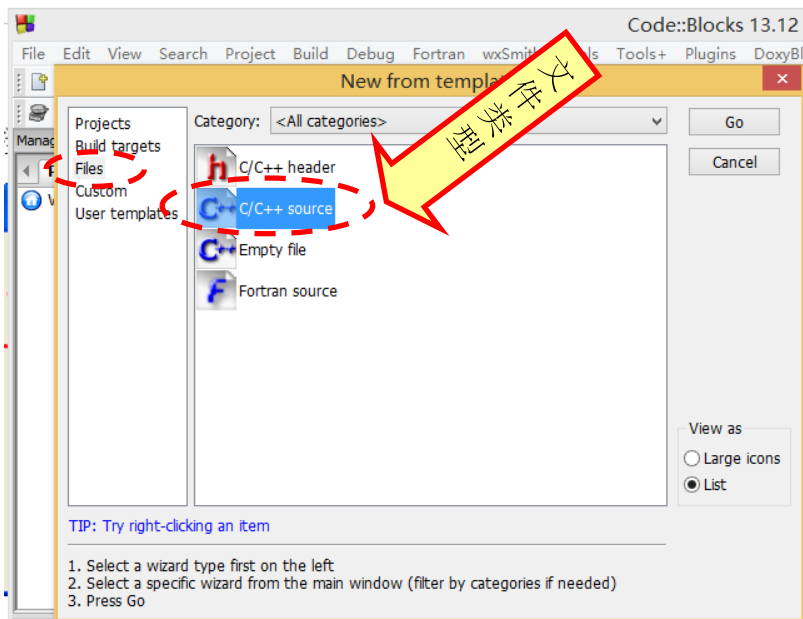
13. 设定Settings > Editor编辑器 > General settings常规设定
14. 设定Settings > Editor编辑器 > Syntax Highlighting语法高亮
关键字(常用字)集Set 1, 2, 3, ..., 10
15. 设定Settings > Editor编辑器 > Abbreviations缩写替换
16. 设定Settings > Editor编辑器 > Code completion自动完成编码
17. 设定Settings > Editor编辑器 > Source formatter代码格式化 (AStyle)
18. 启用汉化界面设定Settings > Environment环境 > View查看 (Internationalization)
19. 如何使用 Code::Blocks 的调试功能。



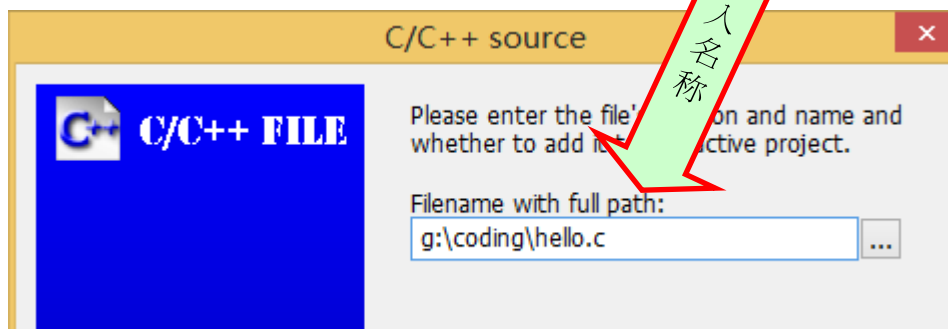
1. 文档File → 开新New → 文档Empty File (再另存新档 x:\coding\hello.c)



2. 开新文档 File → New → File → C/C++Source : x:\coding\hello.c



3. 文件名 : g:\coding\hello.c 新文件名



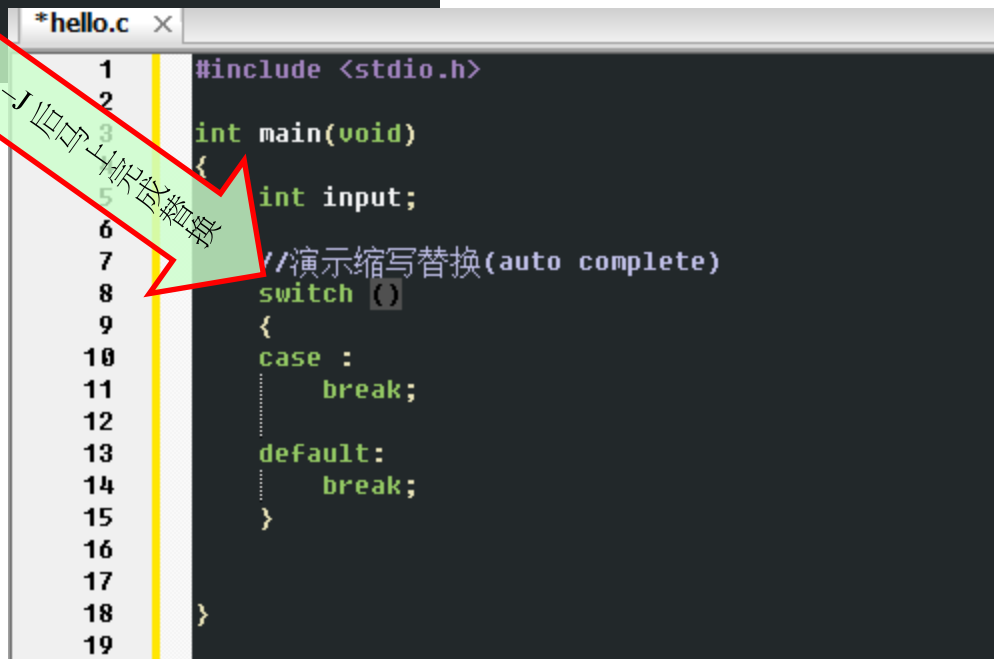
4. 按Ctrl-J (Auto Complete)

自动填上替换字符串



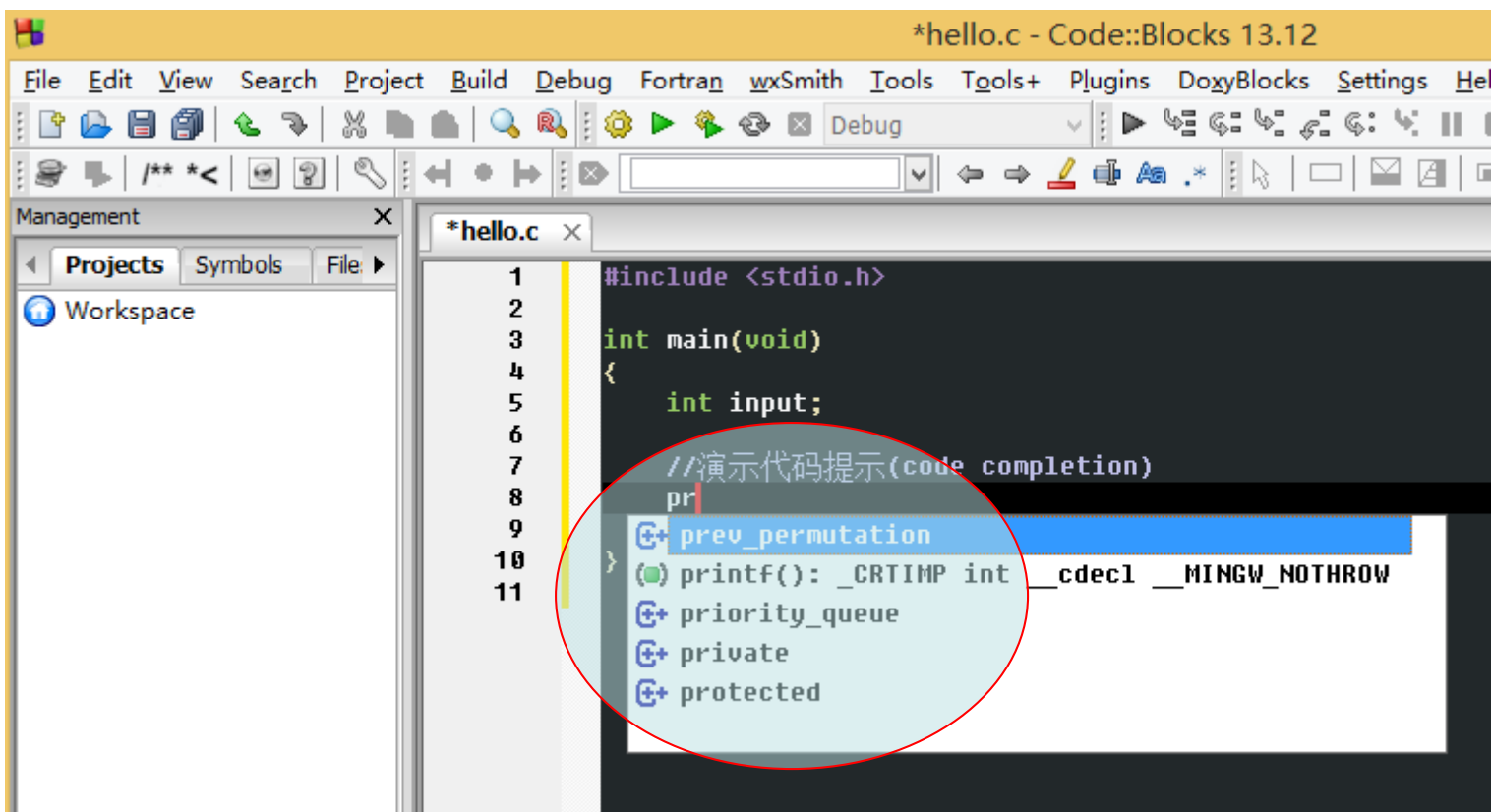
```
*hello.c x
1  #include <stdio.h>
2
3  int main(void)
4  {
5      int input;
6
7      //演示缩写替换(auto complete)
8      switch
9
10
11 }
```

Auto Complete 会自动替换 switch 所代表的代码
你可以在
settings->editor->Abbreviations 中
修改/添加/删除缩写替换内容

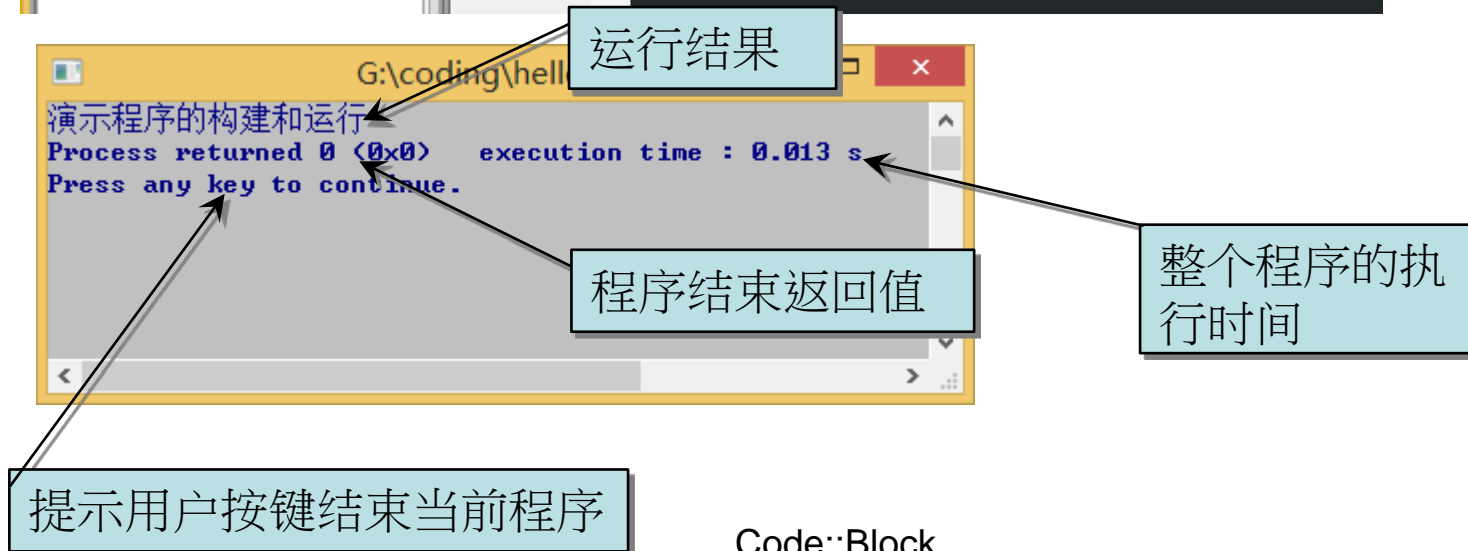
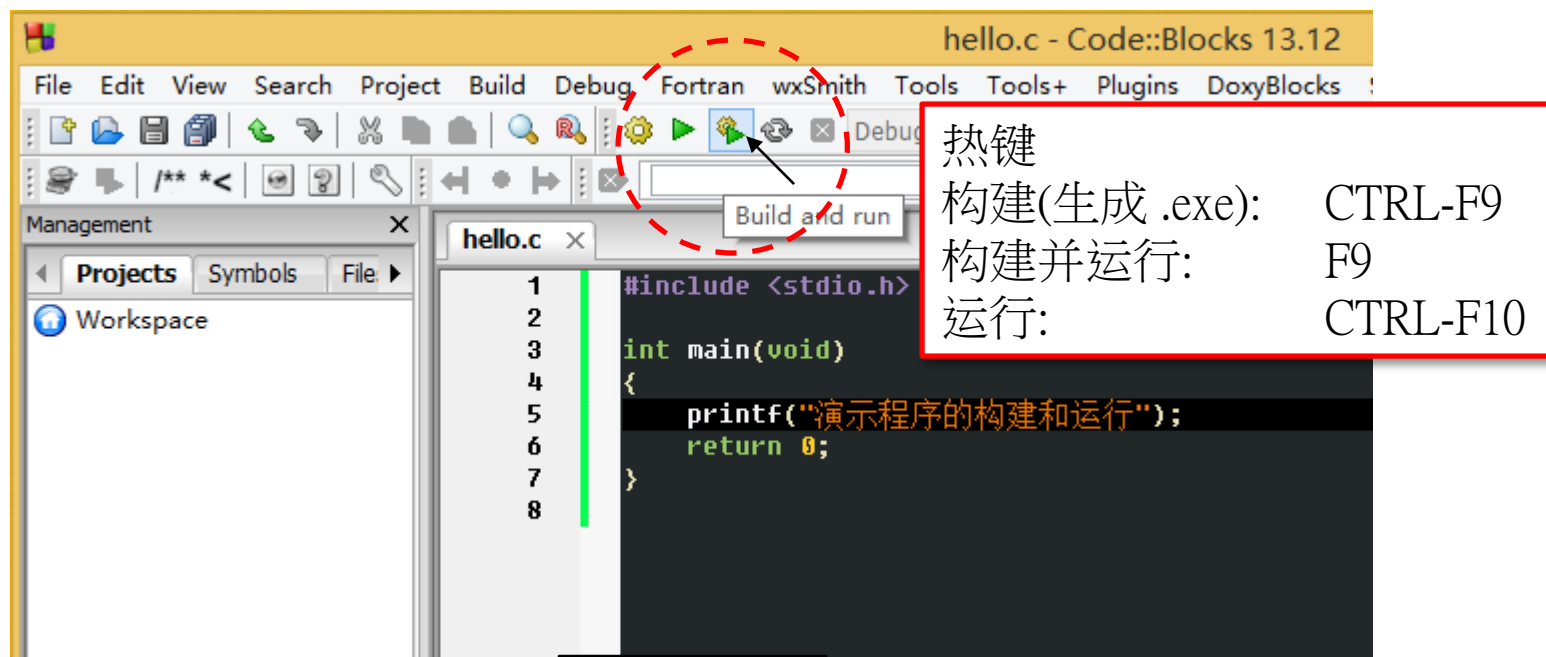


```
*hello.c x
1  #include <stdio.h>
2
3  int main(void)
4  {
5      int input;
6
7      //演示缩写替换(auto complete)
8      switch
9      {
10         case :
11             break;
12
13         default:
14             break;
15     }
16
17
18
19 }
```

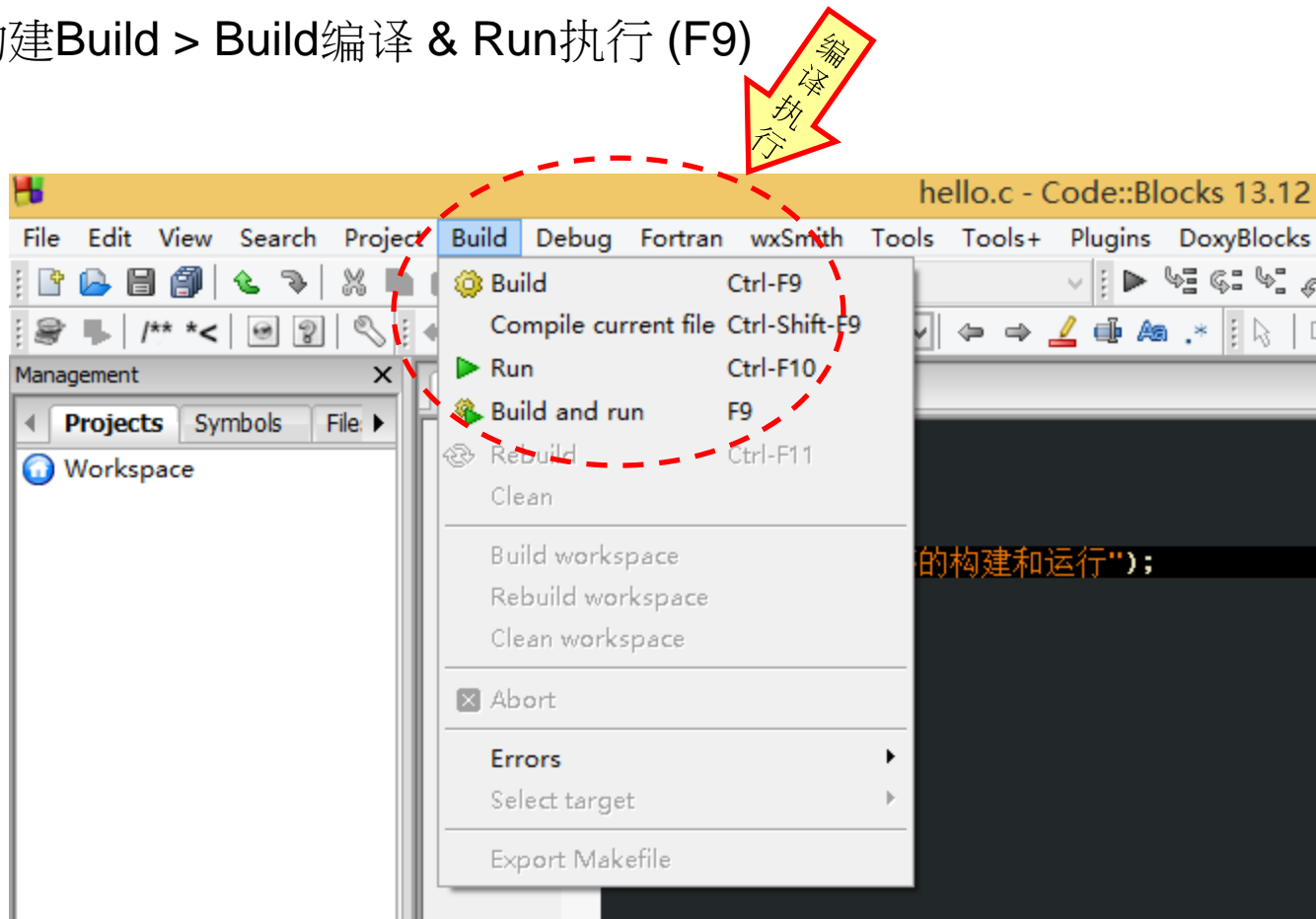
5. 代码提示。



6. 工具栏Tool Bar：编译 & 执行



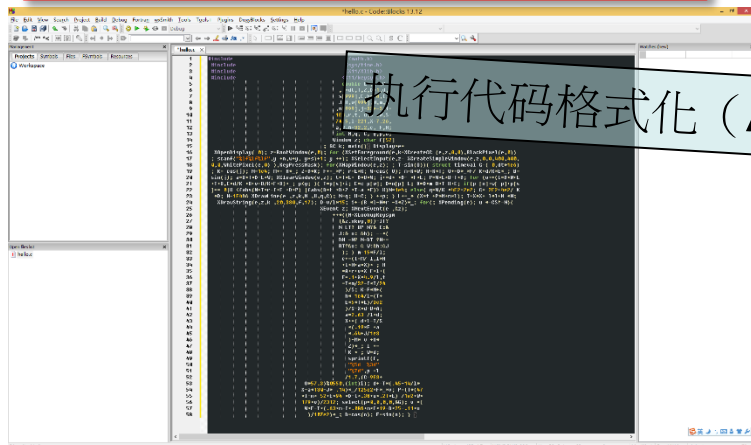
7. 构建Build > Build编译 & Run执行 (F9)



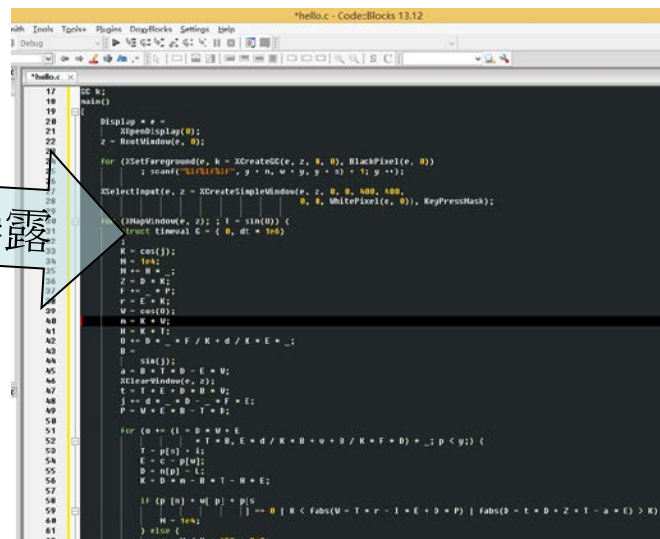
8. 插件Plugins > AStyle代码格式化 (indent32)

Astyle 热键: CTRL- SHIFT -A

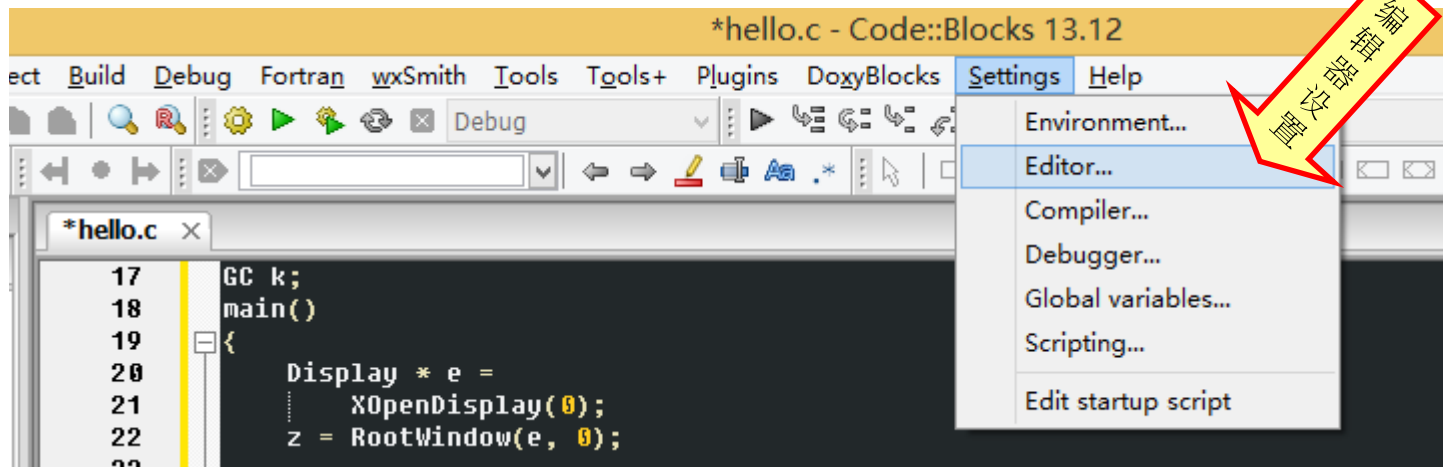
国际 C 语言混乱代码大赛作品



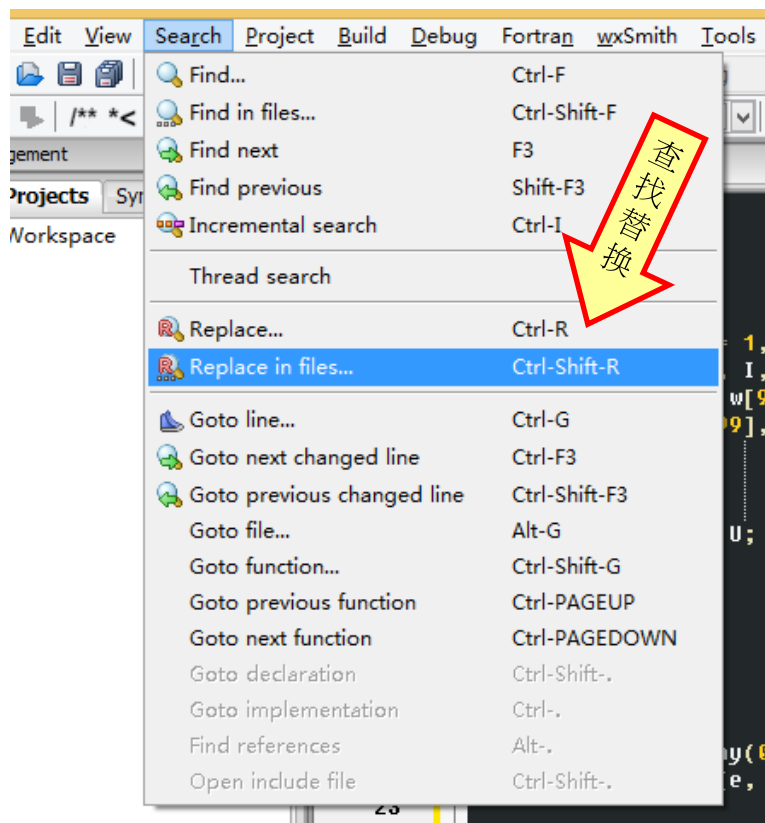
执行代码格式化 (Astyle) 后原形毕露



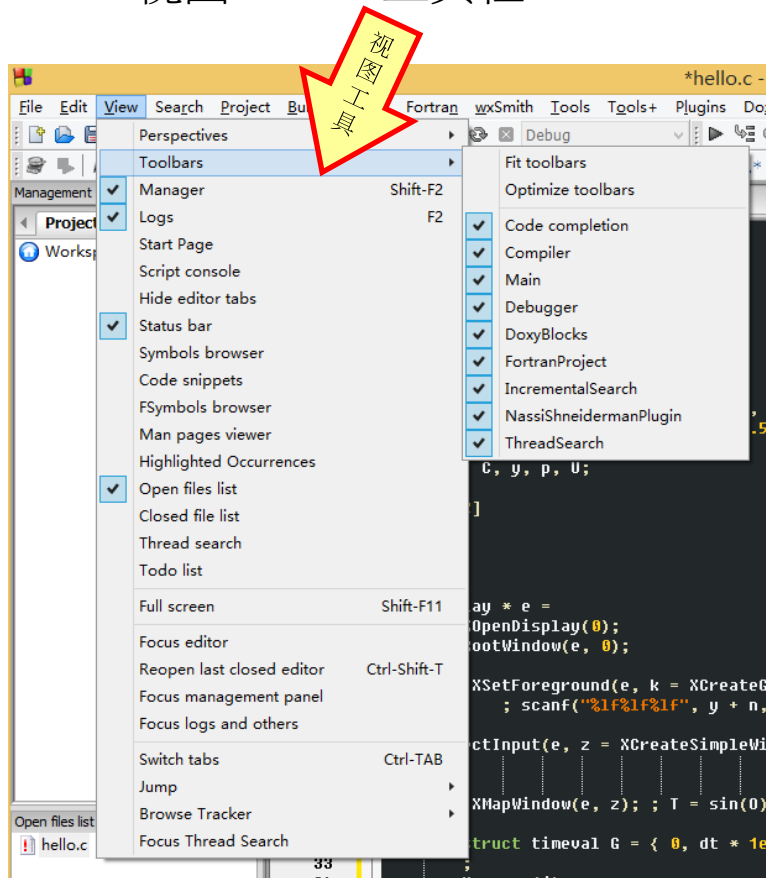
9. 设定Settings > Editor编辑器



10. 查找Find、替换Replace



11. 视图View→工具栏Toolbars



12. 配置文件:

绿色便携版: CodeBlocks目录\AppData\codeblocks\default.conf

官方安装版: C:\Documents and Settings\当前用户名\Application Data\CodeBlocks\default.conf

Code::Block

13. 设定Settings > Editor编辑器 > General settings常规设定

Configure editor
General settings

Editor settings C/C++ Editor settings Other settings

Font
This is sample text Choose

☐ Reset zoom of all editors to default, if leaving dialog

TAB options
☒ Detect indent style
☒ Use TAB character
☒ TAB indents
TAB size in spaces: 4

End-of-line options
☐ Show end-of-line chars
☒ Strip trailing blanks
☒ End files with blank line
☐ Ensure consistent EOLs
End-of-line mode: CR LF

Indent options
☒ Auto indent
☒ Smart indent
☒ Brace completion
☒ Backspace unindents
☒ Show indentation guides
☒ ~~Brace~~ Smart Indent
☐ Selection brace completion

Other options
☐ Word wrap
☒ Home/end to wrap point (line otherwise)
☐ Use POSIX style for RegEx searches
☐ Use Advanced RegEx searches
☒ Show line numbers
☒ Highlight line under caret
☐ Home key always moves caret to first column
☐ CTRL+cursor and CTRL+SHIFT+cursor moves/selects words considering Camel/Case
☐ Zooming resizes all editors
☐ Synchronise Editor with Project Manager

General settings
Folding
Margins and caret
Syntax highlighting
Default code
Abbreviations

常规设定

字体设定

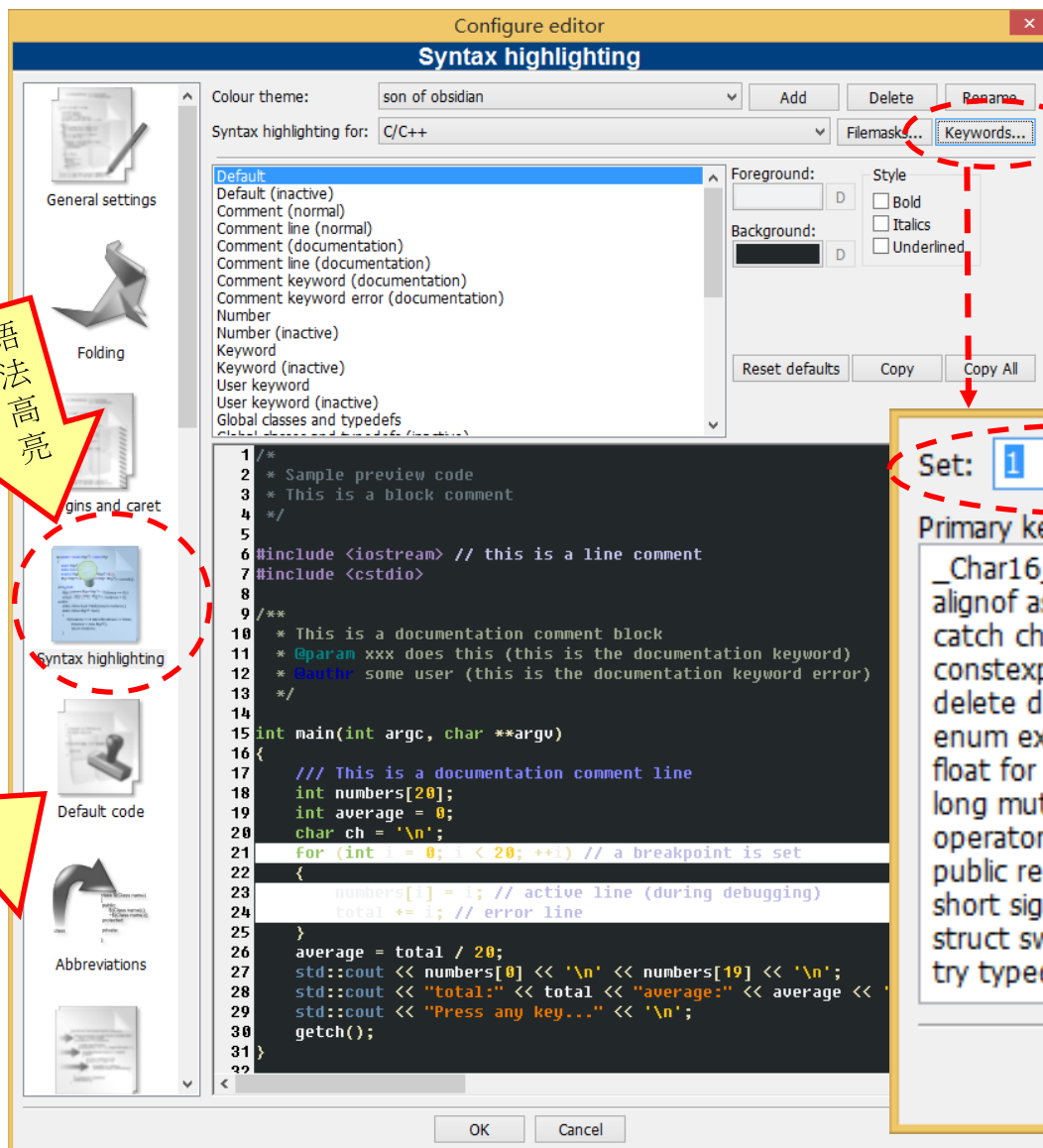
行结束设定

自动缩进设定

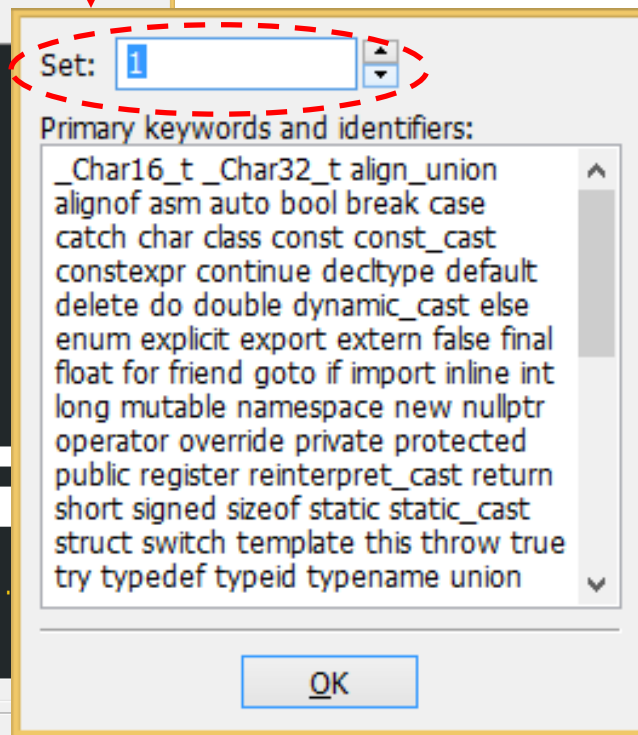
显示行号和高亮当前光标所在行

Code::Block

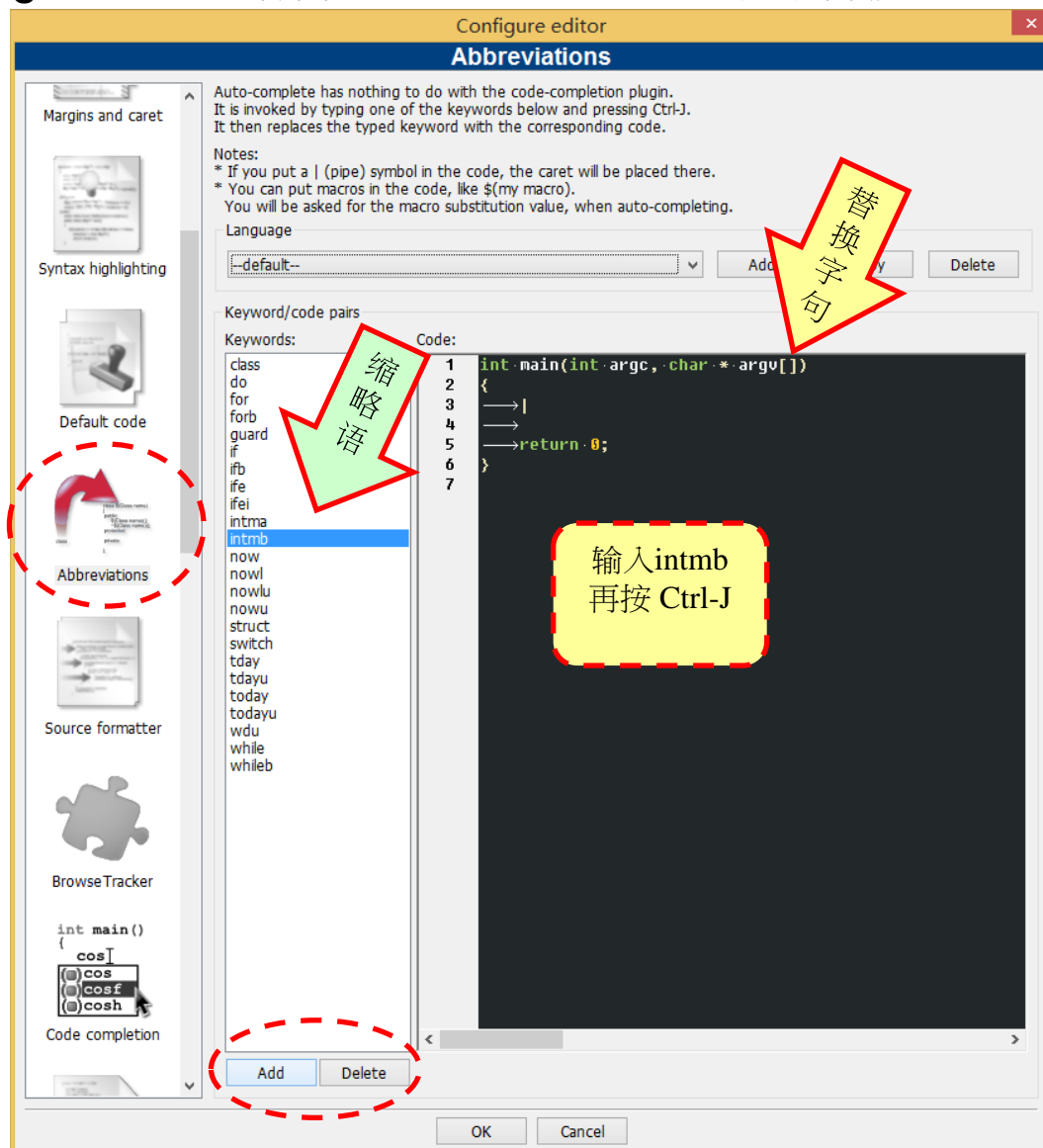
14. 设定Settings > Editor编辑器 > Syntax Highlighting语法高亮



关键字(常用字)集
Set 1, 2, 3, ..., 10

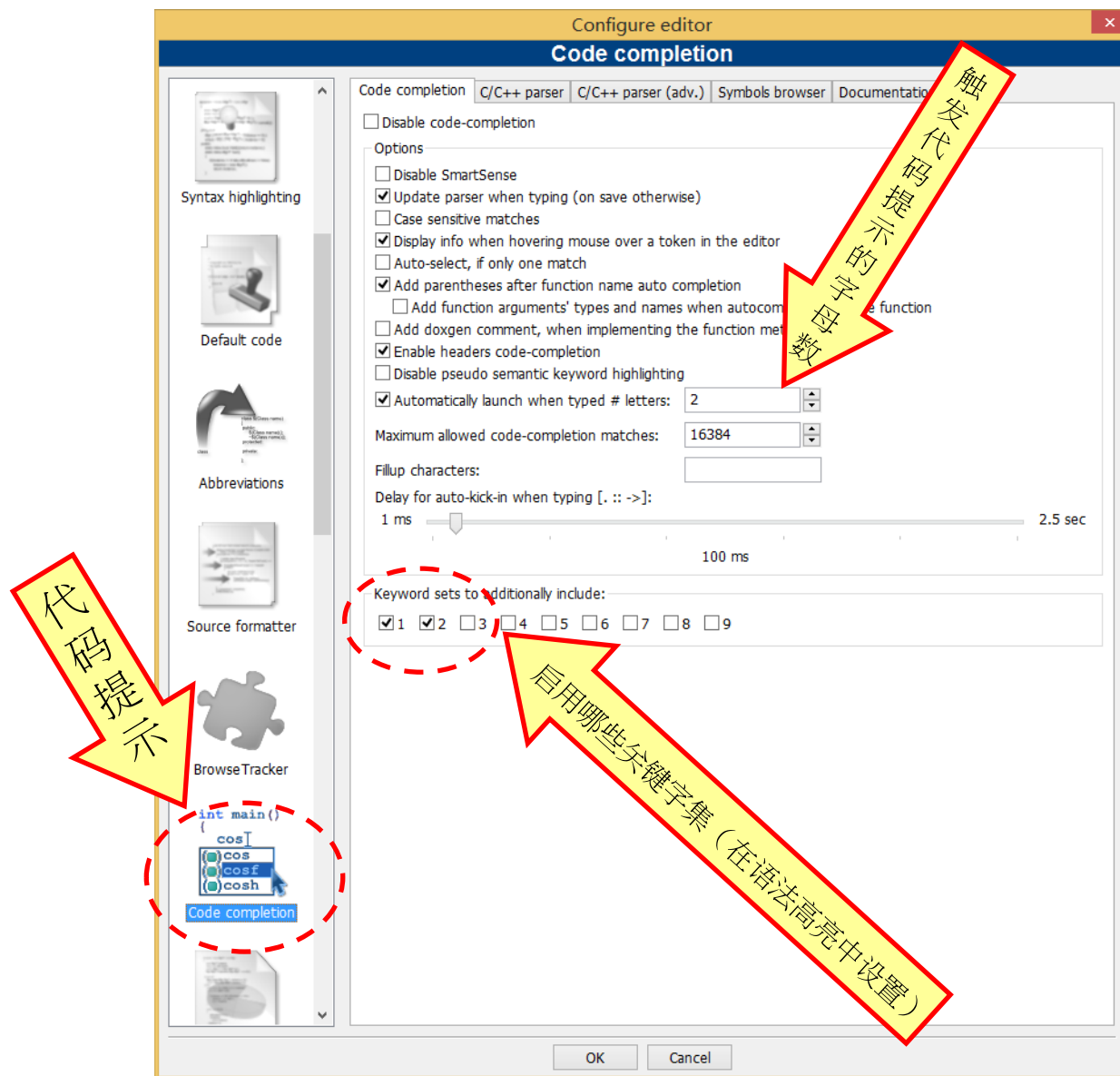


15. 设定Settings > Editor编辑器 > Abbreviations缩写替换

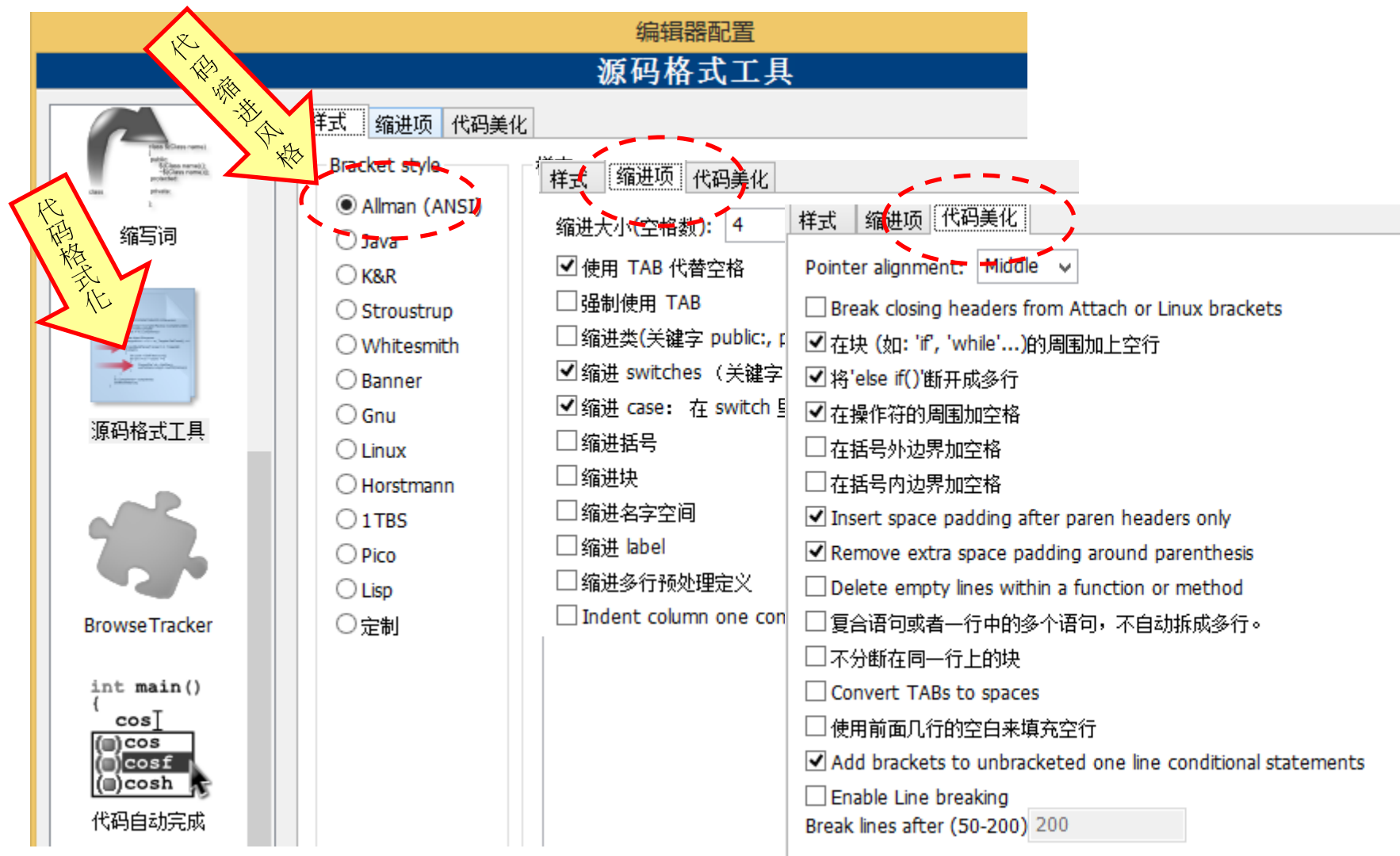


Code::Block

16. 设定Settings > Editor编辑器 > Code completion 代码提示



17. 设定Settings > Editor编辑器 > Source formatter代码格式化 (AStyle)



编辑器配置

源码格式工具

样式 缩进项 代码美化

Bracket style

- ☒ Allman (ANSI)
- ☐ Java
- ☐ K&R
- ☐ Stroustrup
- ☐ Whitesmith
- ☐ Banner
- ☐ Gnu
- ☐ Linux
- ☐ Horstmann
- ☐ 1TBS
- ☐ Pico
- ☐ Lisp
- ☐ 定制

缩进大小(空格数): 4

☒ 使用 TAB 代替空格

☐ 强制使用 TAB

☐ 缩进类(关键字 public, private, ...)

☒ 缩进 switches (关键字 case, ...)

☒ 缩进 case: 在 switch 语句中

☐ 缩进括号

☐ 缩进块

☐ 缩进名字空间

☐ 缩进 label

☐ 缩进多行预处理定义

☐ Indent column one comment

样式 缩进项 代码美化

Pointer alignment: Middle

☐ Break closing headers from Attach or Linux brackets

☒ 在块(如: 'if', 'while'...)的周围加上空行

☒ 将'else if()'断开成多行

☒ 在操作符的周围加空格

☐ 在括号外边界加空格

☐ 在括号内边界加空格

☒ Insert space padding after paren headers only

☒ Remove extra space padding around parenthesis

☐ Delete empty lines within a function or method

☐ 复合语句或者一行中的多个语句, 不自动拆成多行。

☐ 不分断在同一行上的块

☐ Convert TABs to spaces

☐ 使用前面几行的空白来填充空行

☒ Add brackets to unbracketed one line conditional statements

☐ Enable Line breaking

Break lines after (50-200) 200

缩写词

源码格式工具

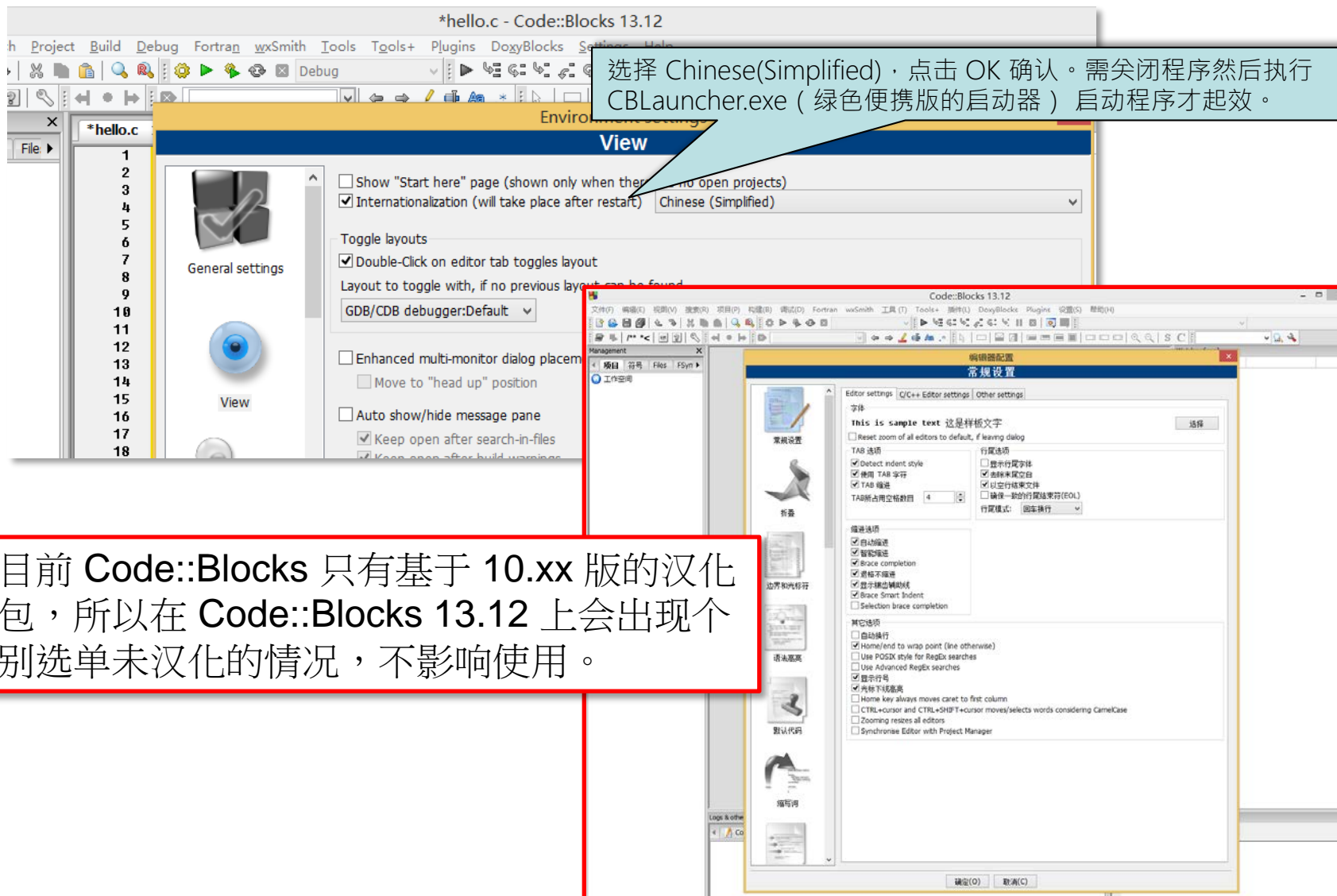
BrowseTracker

```
int main()
{
    cos
    cosf
    cosh
}
```

代码自动完成

18. 设定Settings > Environment环境 > View查看 (Internationalization)

选择 Chinese(Simplified) , 点击 OK 确认。需关闭程序然后执行 CBLauncher.exe (绿色便携版的启动器) 启动程序才起效。



目前 Code::Blocks 只有基于 10.xx 版的汉化包，所以在 Code::Blocks 13.12 上会出现个别选单未汉化的情况，不影响使用。

19. 程序调试 (GDB)

在 CodeBlocks 中启用 GDB 的前提条件有两个：

- 1、新建文档的时候采用项目 (Project) 方式并将当前构建模式设置为 Debug；
- 2、项目/文件目录名和文件名不包含空格、中文等，即必须是纯英文、下划线、数字。

The screenshot shows the Code::Block IDE with the following components and annotations:

- Top Menu Bar:** File, Edit, Build, Debug, Fortran, wxSmith, Tools, Plugins, Settings, Help.
- Toolbar:** A red box highlights the first three buttons: Run, Step Over, and Step Into.
- Main Editor:** Displays the source code of `main.c`. A red arrow points to the first breakpoint set at line 12.
- Watches (new) window:** Shows the value of the `time` variable as `1.0822178536442624e-312` and `1111111111111111111111111111111100` in binary.
- Debugger window:** Shows the GDB console output, including the command `gdb (GDB) 7.6.1` and the process ID `6344`.
- Disassembly window:** Shows the assembly code for the `main` function, starting at address `0028F30`. A yellow arrow points to the instruction `call $0x401377, (%esp)`.

Annotations (Callouts):

- 当前模式需设置为 Debug 模式** (Current mode must be set to Debug mode): Points to the `Debug` menu item.
- 按 F4 的话可以从光标所在行开始执行程序并在光标所在行停下来** (Pressing F4 allows you to start execution from the line where the cursor is and stop at the line where the cursor is): Points to the `Run` button in the toolbar.
- 按 F5 可将断点设定为当前行。然后按 F8 可运行程序直到遇到断点才停下。右键点击红色断点标记可以编辑启动断点条件 (Edit break point)** (Pressing F5 can set the breakpoint to the current line. Then pressing F8 can run the program until it meets the breakpoint and stops. Right-clicking the red breakpoint mark can edit the start breakpoint condition (Edit break point)): Points to the breakpoint icon in the main editor.
- Debug 工具条。其中红色三角按钮即第一个按钮是跑到下一个断点或者执行到程序结束。从它往右起的第三个按钮是逐行 C 代码步进方式执行 (热键 F7)。** (Debug toolbar. The red triangle button is the first button, which runs to the next breakpoint or executes to the end of the program. The third button from it is the step-by-line C code execution mode (hotkey F7)). Points to the first three buttons in the toolbar.
- 程序的汇编码 黄色小箭头代表下一条要执行的汇编指令，按 alt-f7 则可以逐条汇编指令的形式进行程序调试。** (Assembly code of the program. The yellow small arrow represents the next assembly instruction to be executed. Pressing alt-f7 can debug the program in the form of assembly instructions one by one). Points to the yellow arrow in the disassembly window.
- Watches 窗口中可以查看各个变量以及各种表达式、常量的值。也可以手动敲表达式求解，例如图中的 `t-20` 含义是显示 `-20` 的二进制表达方式，在该行的中间一栏就会马上显示 `-20` 的二进制 (补码) 形式。你也可以在中栏直接点击右键选择数值的显示方式。** (In the Watches window, you can view the values of various variables and expressions, constants. You can also manually type expressions to solve, for example, the `t-20` in the figure means to display the binary expression of `-20`. In the middle column of this line, it will immediately display the binary (two's complement) form of `-20`. You can also click the right button in the middle column to select the display mode of the value). Points to the Watches window.
- 新建文档时必须使用图中的 project (项目) 方式构建程序才能启用 Debug。** (When creating a new document, you must use the project (project) method shown in the figure to build the program to enable Debug). Points to the `Project` button in the `New from template` dialog.