

# MSVC++ Quick Reference for CSE 142

## Starting Visual Studio

Homework starter code is distributed as an MSVC++ project. Double-click on the **.dsw** file to launch MSVC++. **Do not double-click on the .c or any other file.**

## Workspace / Project

The last two pages give screen shots defining some terms.

To view a program file (e.g., a .c file), double-click on it in the **File View**.

The Windows standard Open and Save menu items and buttons exist. You shouldn't need open – use the File View window. VC++ automatically saves your files when a build occurs.

To add a file to the project, right-click a folder in the File View and select **Add Files To Folder...**

## Toolbars

The buttons shown come from VC++ toolbars. If any of them isn't on your VC++ screen, you need to make the corresponding toolbar visible: select the Tools menu, then Customize, and then the Toolbars tab.

Toolbars used here are **Menu**, **Standard**, **Build Mini-Bar**, and **Debug**.

Customizations apply to editing and debug mode **separately**.

## General



### Workspace

Toggles display of the workspace window.



### Output

Toggles display of the output window.

## Compiling / Running



### Build (F7)

Compiles all source files that have been changed since last build, but does not run the program. Any errors detected are syntax errors.



### Execute Program (Ctrl+F5)

Starts the program. Breakpoints are ignored. The DOS window stays up after the program completes.



### (Debug) Go (F5)

Starts or continues the program. Execution pauses when a line with a breakpoint is reached. Unless your program has code to prevent it, the DOS window disappears immediately after execution completes.

## Debugging Setup



### Insert/Remove Breakpoint (F9)

Place the cursor on the line in the program where you would like execution to pause, then hit this button. A red circle appears to the left of the line, indicating that the breakpoint is set. Hitting this button again removes the breakpoint.

## While Paused at a Breakpoint



### Step Over (F10)

Execute the current source line. If the line contains a subroutine or function call, that routine is executed fully before pausing again.



### Step Into (F11)

Execute the current source line. If the line contains a subroutine or function call, execution is paused at the first line of that routine.



### Step Out (Shift+F11)

Execute the current routine to completion. Execution pauses when control returns to the caller.



### Run To Cursor (Ctrl+F10)

Continue execution, but break when it reaches the line on which the input cursor is currently positioned.

Mouse  
Cursor

### Show Value

Holding the mouse cursor over a variable name in the text file (don't click, and don't move the cursor once positioned) will cause a pop-up display of the variable's current value.

## View/Debug Windows Menu

Toggles display of various informational windows while paused in the debugger.


**Variables** – Shows current values of variables involved in the statement just executed and the next one to be executed. Values in red were changed by the just executed statement. You can change the value in a variable by clicking on the value field and typing away.

**Watch** – Like the Variables window, except for a list of variables whose names you have typed in. Remember that upper and lower case matters. (If the variable name is not in the scope of the line at which execution is paused, a “Symbol not found” error is reported.)

**Call Stack** – Shows the set of calls that have occurred to take execution to where it is currently paused. Double-clicking on a routine name in this window will set the debugger context to that routine (so you can look at the current values of variables there).

# MSVC++ Quick Reference for CSE 142

## Editing

- Ctrl+Z Undo**  
Undoes the last editing change you made. Hit it again, it undoes the one before that. Etc.
- Ctrl+Y Redo**  
If you hit a Ctrl+Z, this puts back the change that Undo undid.
- Ctrl+F Find**  
Looks for next occurrence of a text string in the file. You can set options to insist on matching the case of the characters or that the match be for a full word.
- Ctrl+H Replace**  
You specify a string to match and a string to replace any matches with. Can find the next occurrence of the first string, or replace the current match, or replace all matches in the file.
-  **Find In Files**  
Does a 'Find' but in all files in the project. Matches are shown in the Output Window. Double-click on a match to go there.
- Tab Indent**  
If one or more entire lines is selected, indents those lines one tab. Shift+Tab "unindents" all selected lines one tab. (If no full line is selected, the selected text is replaced with a tab!)
- Ctrl+] Find Matching Brace**  
If the cursor is positioned next to a '{' or '}', moves the cursor to the brace that matches.
- F1 Help**  
Looks up in online help the word where the input cursor is positioned.

In addition, the Windows standard Cut (Ctrl+X), Copy (Ctrl+C), and Paste (Ctrl+V) keys work.

## File System Files

Generally speaking, you should use the File Explorer (or Folder) view of the files in the project only to locate the .dsw file (to start VC++). **Beyond that, there is no need for you to know about or understand the following:**

### Main Folder

- .dsw The workspace description file. Indicates which projects are in the workspace.
- .dsp Contains project information: which files are in the project, as well as project settings.
- .ncb "No Compile Browser" file used by Class Wizard (in C++).
- .opt Workspace option settings kept here.
- .plg The build log file. Actually an HTML file, meaning you can view it in a browser.

### Debug Folder

(For portability, we distribute projects set up so that this folder is created in c:\temp\

- .obj The actual machine instructions resulting from compiling a .c or .cpp file. These files are input to the linker.
- .exe The executable file resulting from linking all the .obj files. This file can, in theory, be given on its own to anyone with a Windows machine and they can run your program.
- .ilk Used for incremental linking.
- .pch Pre-compiled header file.
- .pdb Information for the debugger.
- .idb Used for incremental compiling.

The entire debug folder can be erased without harm to the project. (In fact, all files can be regenerated if you save the .c, .h, .dsw, and .dsp files alone, and erase the others.)

## Additional Information

Online help is available through the Help menu and the F1 key.

<http://www.cs.washington.edu/education/courses/142/tips/msvc6.0-tips.html> has more/other information.

# MSVC++ Quick Reference for CSE 142

## Screen Shot While In Editor

The screenshot displays the Microsoft Visual C++ IDE interface. The main editor window shows a C++ program with the following code:

```
int main(void) {  
    int firstNumber, secondNumber; /* input values typed by t  
  
    /* Prompt for first integer and get it from the keyboard  
    printf("Please enter an integer: ");  
    scanf ("%d", &firstNumber);|  
  
    /* Prompt for second integer and get it. */  
    printf("Please enter another integer: ");  
    scanf ("%d", &secondNumber)  
  
    /* Print both integers and their product */  
    printf("The value of the product %d * %d is %d.\n", first  
  
    return 0;  
}
```

The code contains a syntax error: a missing semicolon before the identifier 'printf' on the line: `printf("The value of the product %d * %d is %d.\n", first`. The error is highlighted in the Output window at the bottom, which shows the message: `error C2146: syntax error : missing ';' before identifier 'printf'`. The status bar at the bottom indicates the error location: `Ln 17, Col 1`.

Callouts in the image identify the following components:

- Editor window:** The main area where the C++ code is written.
- Workspace window:** The left pane showing the project structure, including 'Source Files' and 'Header Files'.
- FileView tab:** A tab at the bottom of the workspace window showing the current file being edited.
- Input cursor (blinks):** The vertical line in the editor window indicating the current position of the text cursor.
- Output window:** The bottom pane showing the compilation output and error messages.

Additional annotations include:

- Double-click here to show the file in the editor:** Points to the `hw0_orig.c` file in the workspace window.
- A breakpoint has been set on this line:** Points to a red dot on the line containing the second `scanf` call.
- The syntax error is actually here, just before the line where it was detected:** Points to the end of the line with the missing semicolon.
- Syntax errors are shown in the output window. Double-click to go to the line reported:** Points to the error message in the output window.

# MSVC++ Quick Reference for CSE 142

## Screen Shot While Paused in Debugger

