C - APPENDICES

APPENDIX A : KEYWORDS

auto	extern	short	while
break	float	**signed**	*_Alignas*
case	for	sizeof	*_Alignof*
char	goto	static	*_Bool*
const	if	struct	*_Complex*
continue	*inline*	switch	*_Generic*
default	int	typedef	*_Imaginary*
do	long	union	*_Noreturn*
double	register	unsigned	*_Static_assert*
else	*restrict*	void	*#_Thread_local*
enum	return	**volatile**	

APPENDIX B : OPERATOR PRECEDENCE CHART

Operators	**Associativity**
( ) [ ] –> .	left to right *{( ) function call}*
! ~ ++ – – + – * & (*type*) sizeof	right to left *{All Unary}*
* / %	left to right
+ –	left to right
<< >>	left to right
< <= > >=	left to right
= = !=	left to right
&	left to right
^	left to right
|	left to right
&&	left to right
||	left to right
? :	right to left
= += –= *= /= %= &= ^= |= <<= >>=	right to left
,	left to right

APPENDIX C : OPERATORS

Arithmetic operators:

* / %	multiplication/division/modulus
+ −	addition/subtraction
+ −	positive/negative sign (unary)
++ − −	increment/decrement (unary)

Logical operators:

&&	AND
||	OR
!	NOT (unary)

Relational operators:

< <= > >=	less than, less than or equal to, greater than, greater than or equal to
= = !=	equal to and not equal to

Bit operators:

<< >>	left and right bit shift
&	bitwise AND
|	bitwise OR
^	bitwise exclusive or XOR
~	bitwise NOT (unary)

Assignment operators:

= += −= *= /= %= &= ^= |= <<= >>=

Address/Pointer operators:

&	address of (unary)
*	dereference (unary)

Structure operators:

.	structure member acccess
–>	member access thru a structure pointer

Other operators:

( )	function call
[ ]	array access
(*type*)	type cast (unary)
sizeof	data object size in bytes (unary)
?:	conditional operator
,	comma operator

APPENDIX D : RELATIONAL OPERATORS

- Relational operators test a relationship and produce a true/false result.

operator	function
= =	equality
<	less than
>	greater than
<=	less than or equal
>=	greater than or equal
!=	not equal

APPENDIX E : LOGICAL OPERATORS

- Logical operators work on logical values and produce a logical result.

operator	function
&&	AND
||	OR
!	NOT

APPENDIX F : CONVERSION SPECIFIERS

printf( )

%d	signed decimal int
%hd	signed short decimal integer
%ld	signed long decimal integer
%lld	signed long long decimal integer
%u	unsigned decimal int
%lu	unsigned long decimal int
%llu	unsigned long long decimal int
%o	unsigned octal int
%x	unsigned hexadecimal int with lowercase
%X	unsigned hexadecimal int with uppercase
%f	float or double [-]dddd.dddd.
%e	float or double of the form [-]d.dddd e[+/-]ddd
%g	either e or f form, whichever is shorter
%E	same as e; with E for exponent
%G	same as g; with E for exponent if e format used
%Lf,	
%Le,	
%Lg	long double
%c	single character
%s	string
%p	pointer

scanf( )

%d	signed decimal int
%hd	signed short decimal integer
%ld	signed long decimal integer
%u	unsigned decimal int
%lu	unsigned long decimal int
%o	unsigned octal int
%x	unsigned hexadecimal int
%f	float
%lf	double **NOTE:** double & float are distinct for scanf !
%LF	long double
%c	single character
%s	string

APPENDIX G : ESCAPE SEQUENCES

Escape	Value
\n	Newline
\t	Tab
\f	Formfeed
\a	Alarm
\b	Backspace
\r	carriage return
\v	vertical tab

APPENDIX H : FILE ACCESS MODES

	r	w	a	r+	w+	a+
File must exist before open	*			*		
Old file truncated to zero length		*			*	
Stream can be read	*			*	*	*
Stream can be written		*	*	*	*	*
Stream can be written only at end			*			*

APPENDIX I : USING THE GCC COMPILER

- Create a file using your favorite text editor or retrieve some file containing C source code from somewhere.{By convention your file should end in *.c* and the gcc compiler requires it.}
- To compile:

  gcc hello.c

  compiles and links the file *hello.c* and produces a file called *a.exe* that can be executed

  gcc -o hello hello.c

  compiles and links the file *hello.c* and produces an executable file called *hello.exe*
  Note: *-o* is for "output", the *.exe* is added if you don't specify it

  gcc -Wall hello.c

  compiles and links and displays warnings on *all* things that are somewhat questionable

  gcc -c hello.c

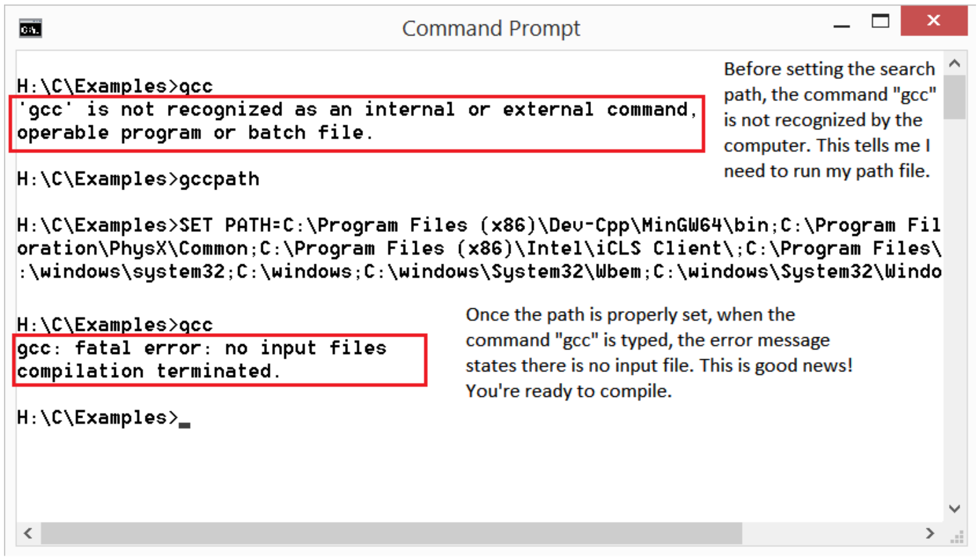
  compiles only; stops at object module and does not run the linker

- There is documentation available for the gcc compiler.
- The gcc compiler can cross-compile for many different processors.

…See some examples on the following pages…

GCC Basics:

- Check that the compiler is installed and the search path set correctly by simply typing “gcc” in the command window. If there are any errors, reference the “Search Paths” section of the Appendix notes on otto for more information.



```
H:\C\Examples>gcc
'gcc' is not recognized as an internal or external command,
operable program or batch file.

H:\C\Examples>gccpath

H:\C\Examples>SET PATH=C:\Program Files (x86)\Dev-Cpp\MinGW64\bin;C:\Program Fil
oration\PhysX\Common;C:\Program Files (x86)\Intel\iCLS Client\;C:\Program Files\
:\windows\system32;C:\windows;C:\windows\System32\Wbem;C:\windows\System32\Windo

H:\C\Examples>gcc
gcc: fatal error: no input files
compilation terminated.

H:\C\Examples>_
```

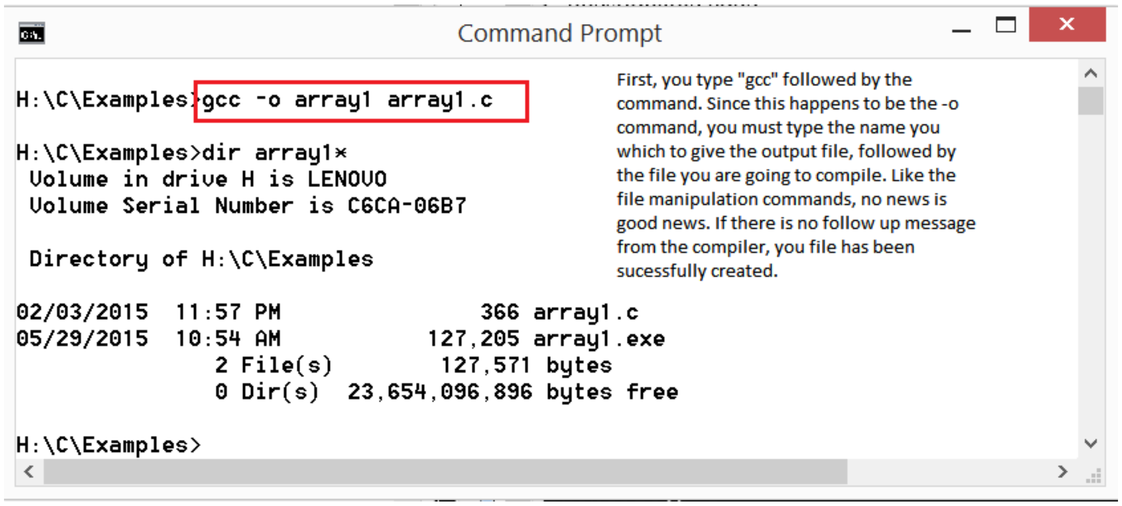
Before setting the search path, the command "gcc" is not recognized by the computer. This tells me I need to run my path file.

Once the path is properly set, when the command "gcc" is typed, the error message states there is no input file. This is good news! You're ready to compile.

Basic GCC Compiling Options

- Note! All gcc commands/options are case sensitive.

-Wall	Provide warnings about "all" questionable code
-c	Compile only. Produces an Object module, but does not create an executable
-o	Name the output file. If unspecified, defaults to "a.exe" or "a.out"



```
H:\C\Examples>gcc -o array1 array1.c

H:\C\Examples>dir array1*
 Volume in drive H is LENOVO
 Volume Serial Number is C6CA-06B7

 Directory of H:\C\Examples

02/03/2015  11:57 PM               366 array1.c
05/29/2015  10:54 AM           127,205 array1.exe
               2 File(s)        127,571 bytes
               0 Dir(s)  23,654,096,896 bytes free

H:\C\Examples>
```

First, you type "gcc" followed by the command. Since this happens to be the -o command, you must type the name you which to give the output file, followed by the file you are going to compile. Like the file manipulation commands, no news is good news. If there is no follow up message from the compiler, you file has been sucessfully created.

APPENDIX J : SETTING THE SEARCH PATH

- The operating system only looks in a few places for programs
  - The current working directory
  - The directories listed on the *search path*: PATH
- To change the directories the operating system will search you can modify the PATH environment variable:
  - On the Windows OS:
    - Use the *set* command

      set PATH=%PATH%;C:\ Dev-Cpp\MinGW64\bin

      - This will assign the previous value of PATH (indicated by %PATH%) to the PATH variable along with the new C:\ Dev-Cpp\MinGW64\bin directory

      - In this course the settings might be:

        set PATH=%PATH%; C:\program files\dev C++\bin

      Or//

        set PATH=%PATH%; C:\Program Files (x86)\Dev C++\bin

      Or//

        set PATH=%PATH%; D:\Dev-Cpp\MinGW64\bin

        - For using Dev C++ from a CD

  Note:

    - You can set the PATH environment variable permanently on a machine you control by going to:

      Start -> *Control Panel* -> *System* -> *Advanced System Settings* -> *Environment Variables*

      and editing the PATH variable there.

  - On the UNIX/Linux OS:

    Changing the PATH environment variable is generally unnecessary since gcc is commonly installed, or when installed is placed in the */usr/local/bin* directory which is generally searched by default.

cont…

But you can change the PATH variable on UNIX/Linux also:

- Change the PATH environment variable by assigning a new value
  PATH=$PATH:/usr/local/bin

- You can automate the setting of this value by altering:
  `.bash_profile` for a particular user

Or//

`/etc/profile` for all users

- If installed properly, and your PATH is set properly, you should be able to type *gcc*, the *gcc* compiler command, and get a response ”*gcc: no input files*“ back from the compiler.

  Ex:

  H:\examples> gcc
  gcc: no input files

If you don’t get this response, verify that the path you have set is correct, and that the *gcc.exe* file is in fact in that *\bin* subdirectory you suspect. If it is, try running it by typing the command with the complete path to the compiler in the command.

Ex:

D:\Dev-Cpp\MinGW64\bin\gcc

Or//

C:\program files\dev C++\bin\gcc

Or//

C:\Program Files (x86)\Dev C++\bin\gcc

Or//

whatever the actual path to the compiler is: *<some path>\gcc*

You should then get the ”*gcc: no input files*“ back from the compiler. If not, find the *\bin* directory that does contain the *gcc.exe* file.

cont…

- If the compiler is in fact on your machine; recheck your PATH variable. This can be done by typing *PATH* at the command prompt, to show your PATH variable setting, or by typing just *set*, to show all your environment variable settings. Your PATH variable must contain the complete path to your *compiler* in the *\bin* directory

Note:

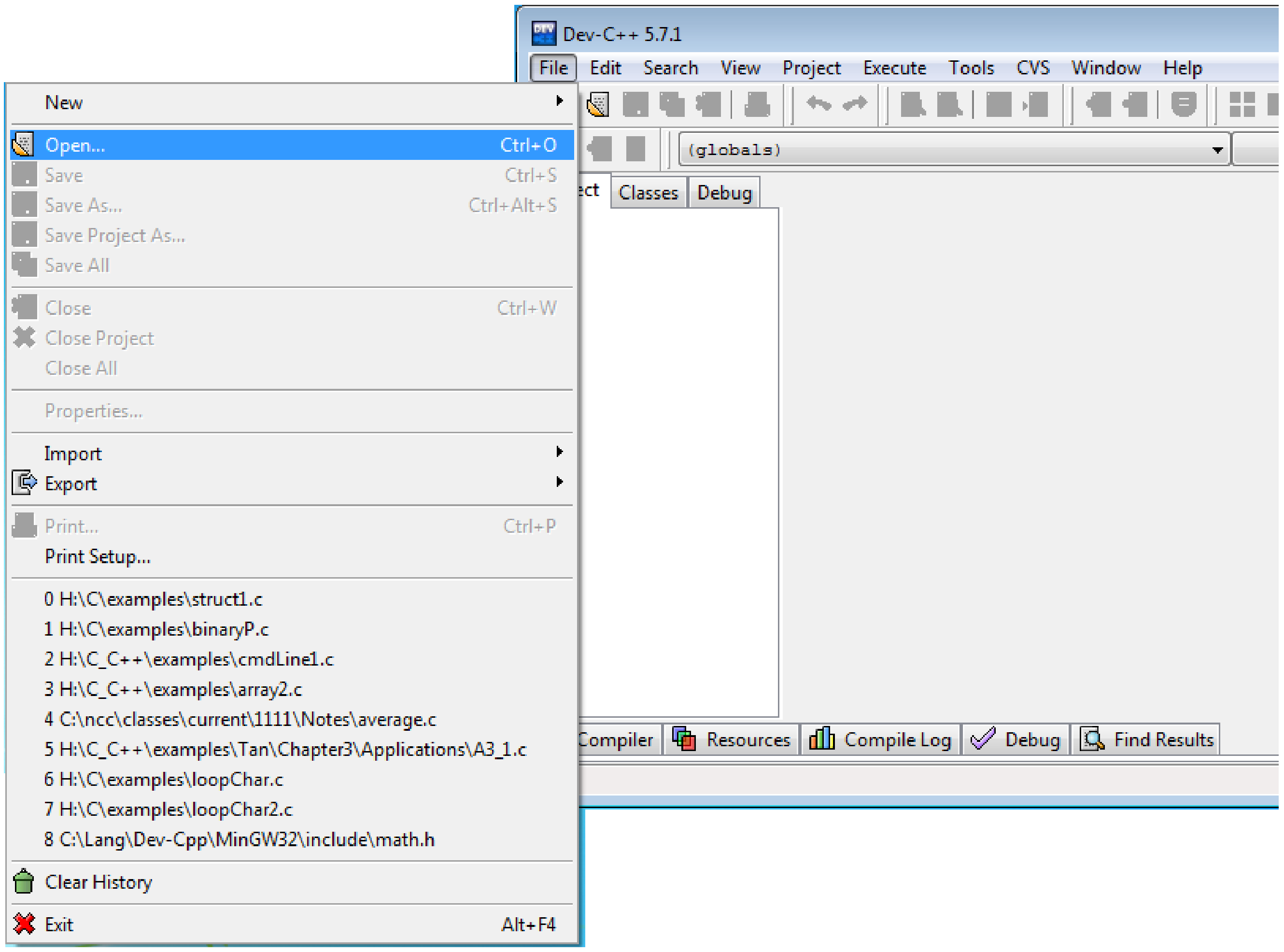
- The actual path to the */bin* subdirectory will likely be different on different machines, depending on the installation directory and name of the particular program that was installed.
- Generally, when setting the PATH value in a command window/shell, that setting is only good within that window/shell. If you exit that window/shell the setting will be lost
- Check your *\examples* directory for a provided batch file or script that will set the PATH variable for you, or at least serve as a model for you to set it.

APPENDIX K : USING THE DEV C++ IDE

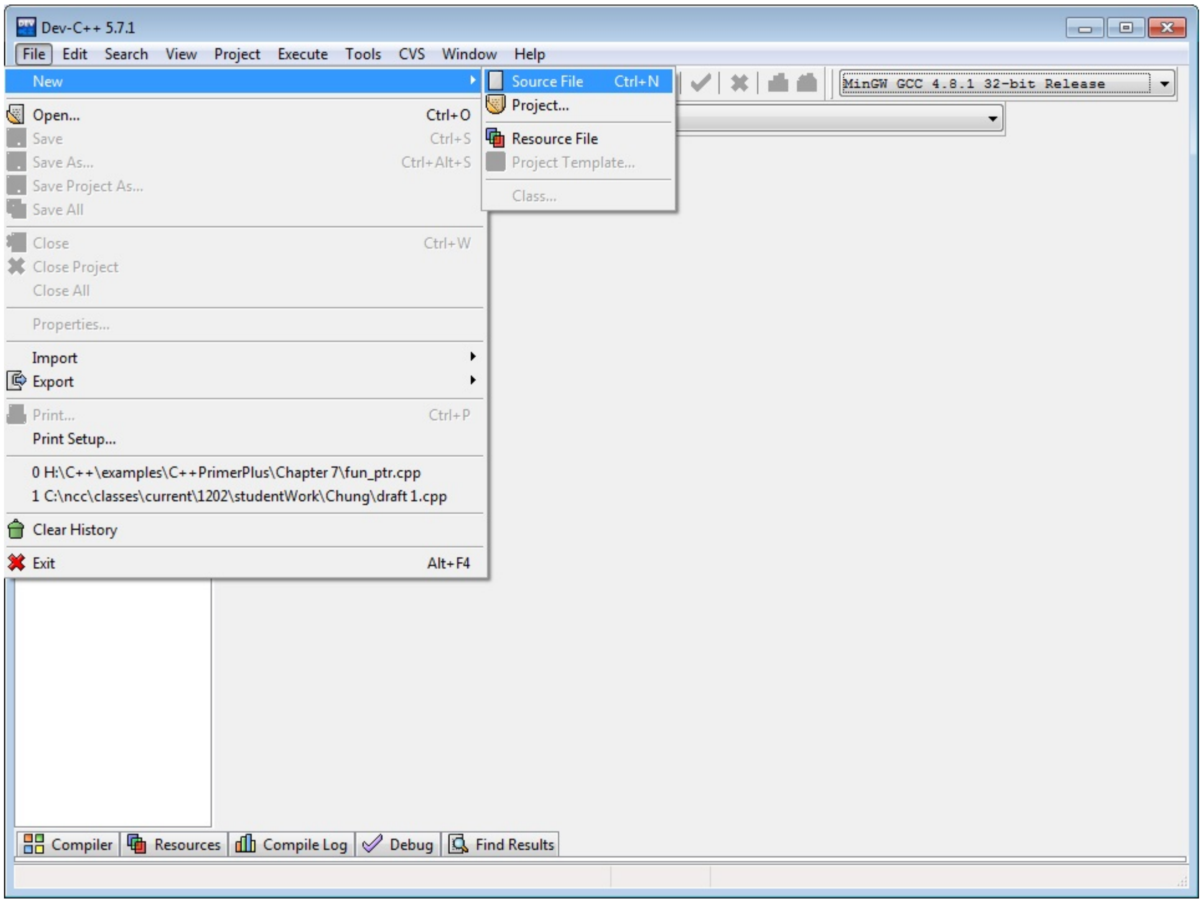
- The Dev-C++ Integrated Development Environment can be used to create, run and debug C programs on a PC.
- It has the following useful features:
  - It is available free of charge @ `http://sourceforge.net/projects/orwelldevcpp/`
  - It utilizes the gcc compiler

Start Dev C++:

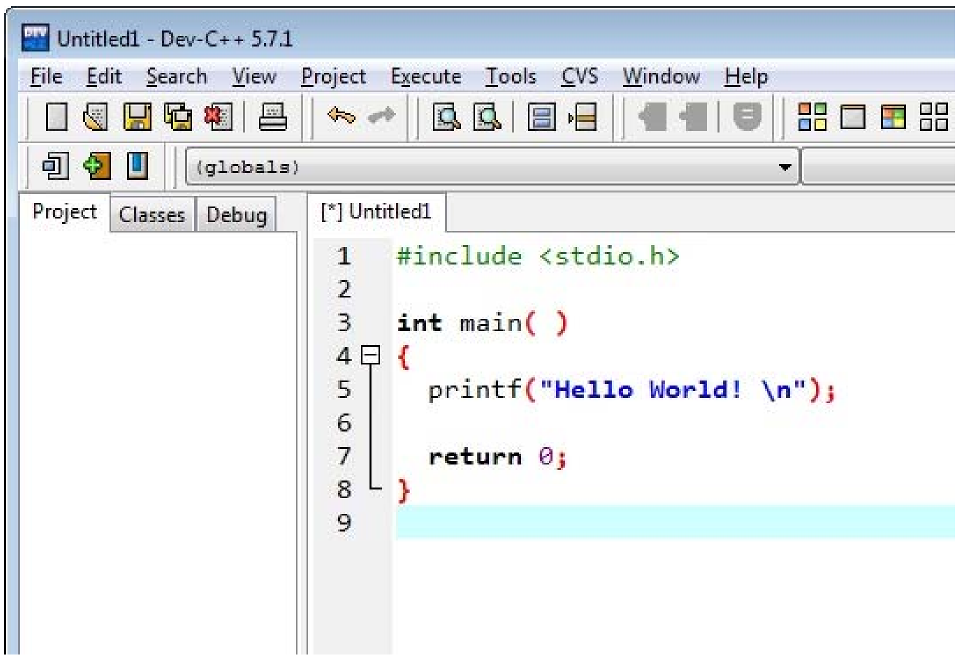
Open an exisiting C file: (Be sure you have "File extensions on – See Appendix.)



or// create a C file:

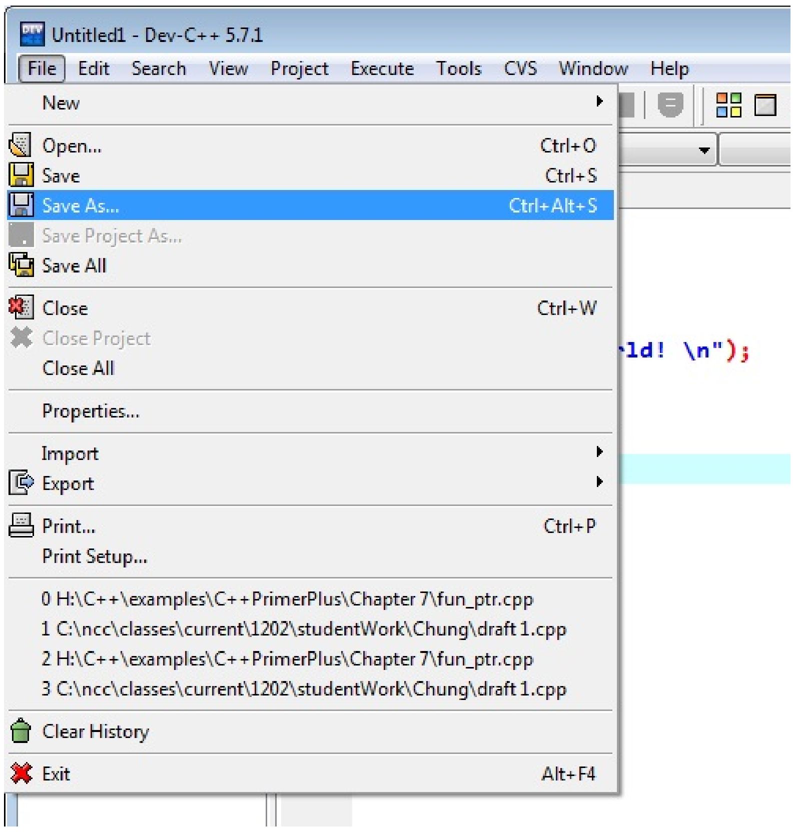


Type in your source code:

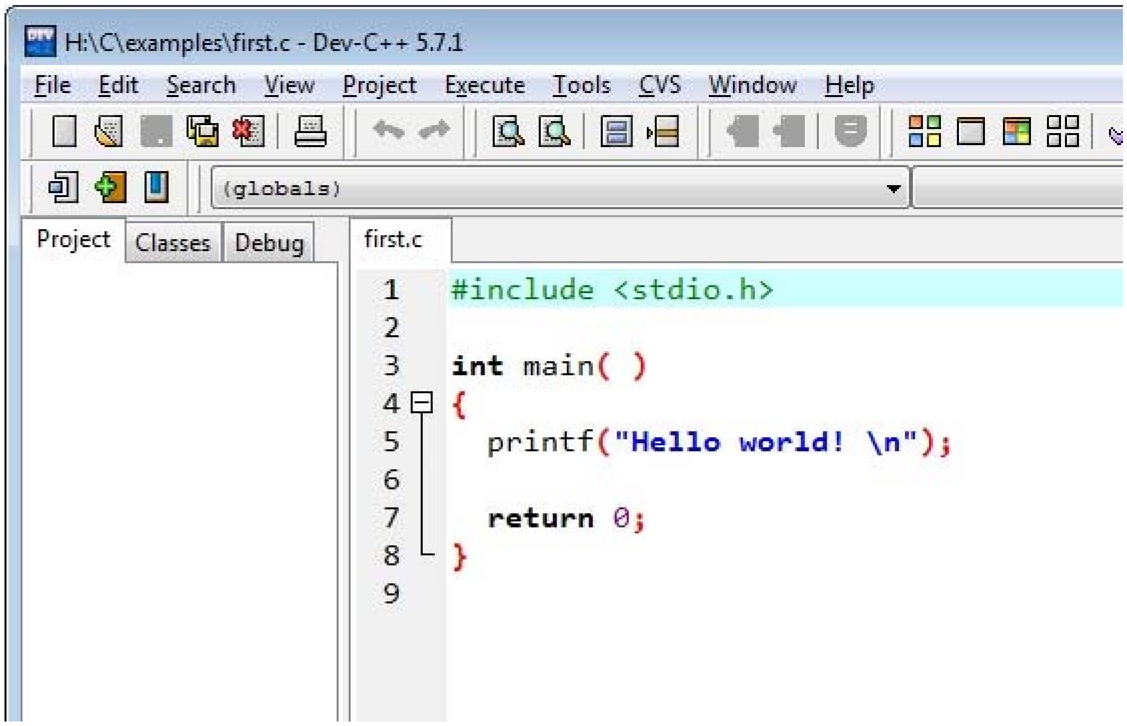

Untitled1 - Dev-C++ 5.7.1
File Edit Search View Project Execute Tools CVS Window Help
(globals)
Project Classes Debug
[*] Untitled1

```c
#include <stdio.h>

int main( )
{
   printf("Hello World! \n");

   return 0;
}

```

Save the file using “Save As”: {Be sure to add the .c extension!}

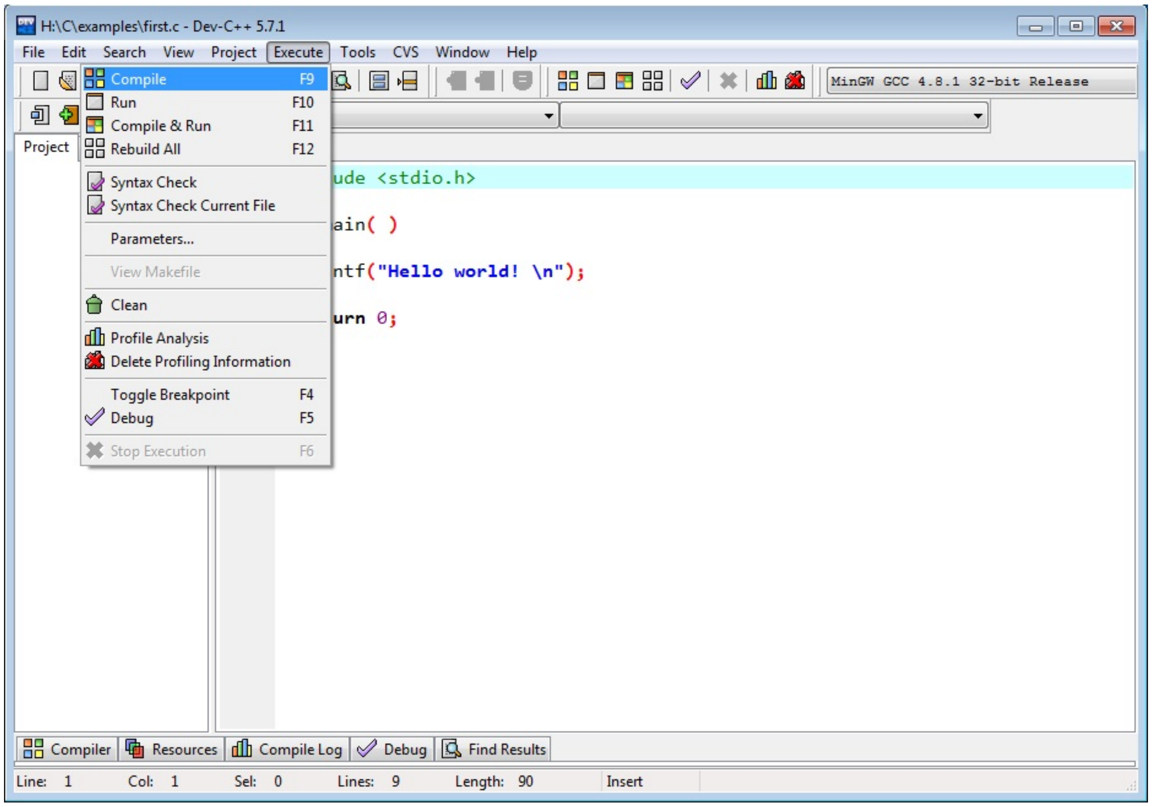

Untitled1 - Dev-C++ 5.7.1
File Edit Search View Project Execute Tools CVS Window Help
New
Open... Ctrl+O
Save Ctrl+S
Save As... Ctrl+Alt+S
Save Project As...
Save All
Close Ctrl+W
Close Project
Close All
Properties...
Import
Export
Print... Ctrl+P
Print Setup...
0 H:\C++\examples\C++PrimerPlus\Chapter 7\fun_ptr.cpp
1 C:\ncc\classes\current\1202\studentWork\Chung\draft 1.cpp
2 H:\C++\examples\C++PrimerPlus\Chapter 7\fun_ptr.cpp
3 C:\ncc\classes\current\1202\studentWork\Chung\draft 1.cpp
Clear History
Exit Alt+F4
ld! \n");

The <File-Tab> indicates the file name: ** **{Note the .c extension!}** **


H:\C\examples\first.c - Dev-C++ 5.7.1
File Edit Search View Project Execute Tools CVS Window Help
(globals)
Project Classes Debug
first.c

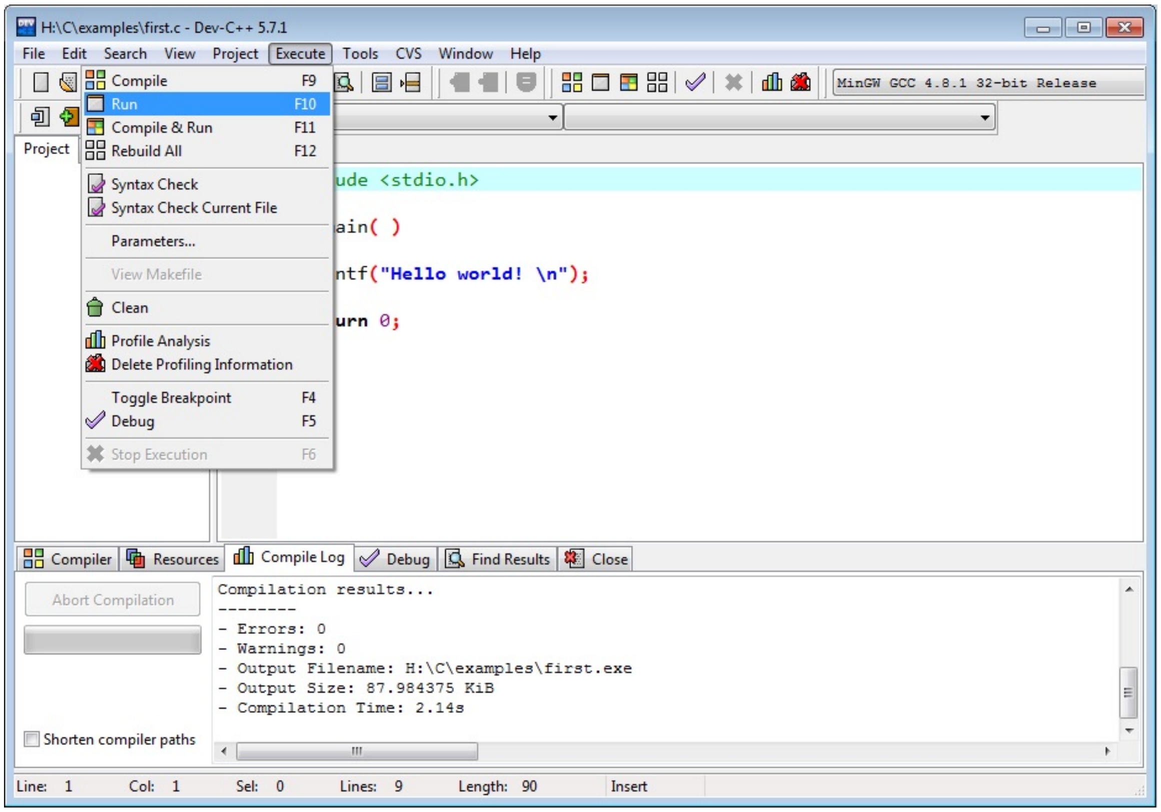
```
#include <stdio.h>

int main( )
{
  printf("Hello world! \n");

  return 0;
}

```

Go to the *Execute* menu and select *Compile*:


H:\C\examples\first.c - Dev-C++ 5.7.1
File Edit Search View Project Execute Tools CVS Window Help
Compile F9
Run F10
Compile & Run F11
Rebuild All F12
Syntax Check
Syntax Check Current File
Parameters...
View Makefile
Clean
Profile Analysis
Delete Profiling Information
Toggle Breakpoint F4
Debug F5
Stop Execution F6
MinGW GCC 4.8.1 32-bit Release
Project
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 1 Sel: 0 Lines: 9 Length: 90 Insert

```
ude <stdio.h>

ain( )

ntf("Hello world! \n");

urn 0;
```

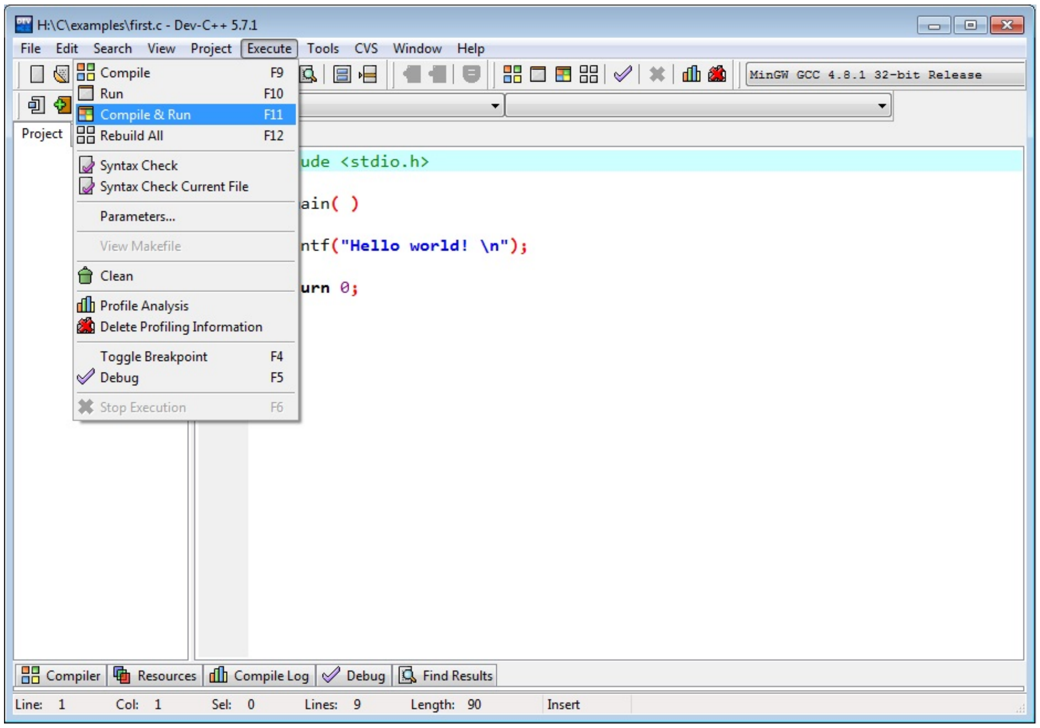
Go to the *Execute* menu and select *Run:*



```
ude <stdio.h>

ain( )

ntf("Hello world! \n");

urn 0;
```

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: H:\C\examples\first.exe
- Output Size: 87.984375 KiB
- Compilation Time: 2.14s
```

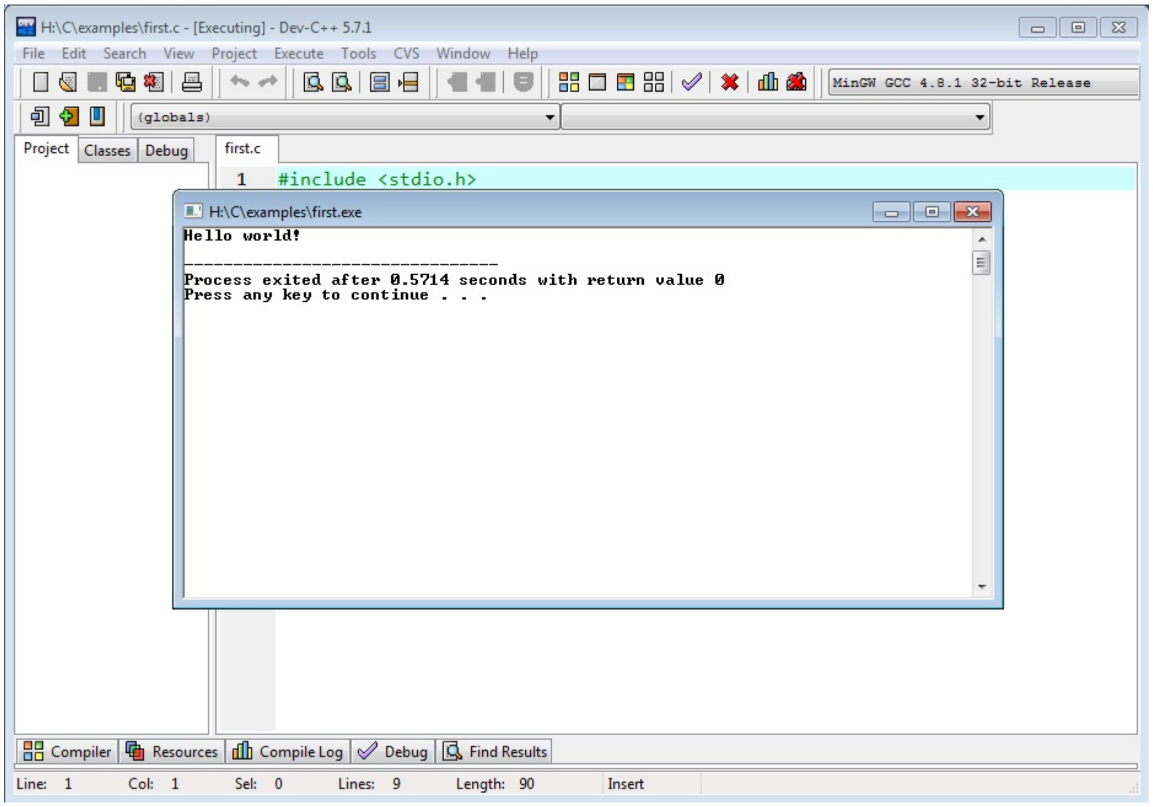
You can also select *Compile & Run*: {It will stop if there are errors.}


H:\C\examples\first.c - Dev-C++ 5.7.1
File Edit Search View Project Execute Tools CVS Window Help
Compile F9
Run F10
Compile & Run F11
Rebuild All F12
Syntax Check
Syntax Check Current File
Parameters...
View Makefile
Clean
Profile Analysis
Delete Profiling Information
Toggle Breakpoint F4
Debug F5
Stop Execution F6
MinGW GCC 4.8.1 32-bit Release
Project
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 1 Sel: 0 Lines: 9 Length: 90 Insert

```
ude <stdio.h>

ain( )

ntf("Hello world! \n");

urn 0;
```

To view the output you have two choices:

1) Run the code within Dev C++:

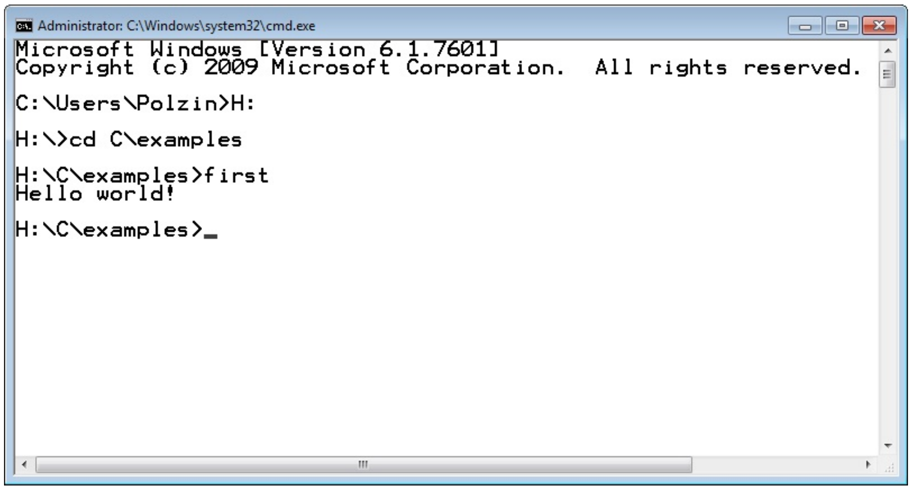

H:\C\examples\first.c - [Executing] - Dev-C++ 5.7.1
File Edit Search View Project Execute Tools CVS Window Help
MinGW GCC 4.8.1 32-bit Release
(globals)
Project Classes Debug
first.c
H:\C\examples\first.exe
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 1 Sel: 0 Lines: 9 Length: 90 Insert

```
#include <stdio.h>
```

```
Hello world!

--------------------------------
Process exited after 0.5714 seconds with return value 0
Press any key to continue . . .
```

2) Run the .exe file in a command window:

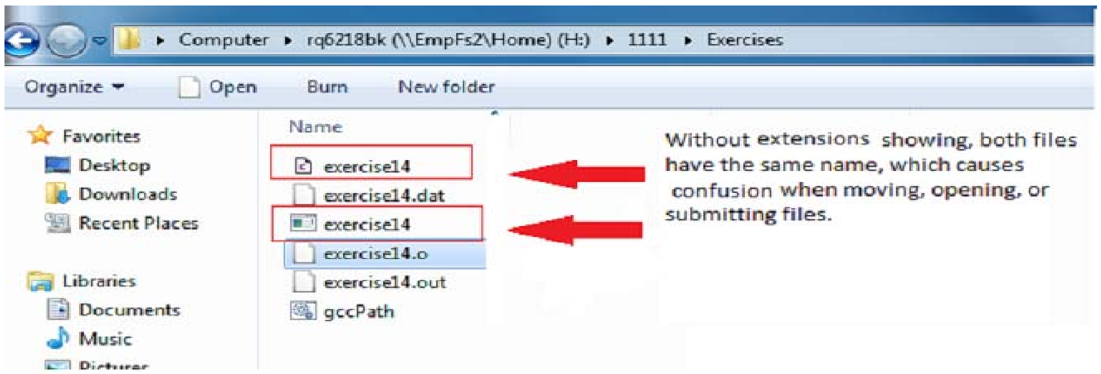
{Open a command window, set your *working* directory, execute the .exe file.}



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Polzin>H:

H:\>cd C\examples

H:\C\examples>first
Hello world!

H:\C\examples>_
```

APPENDIX L : SHOWING FILE EXTENSIONS.

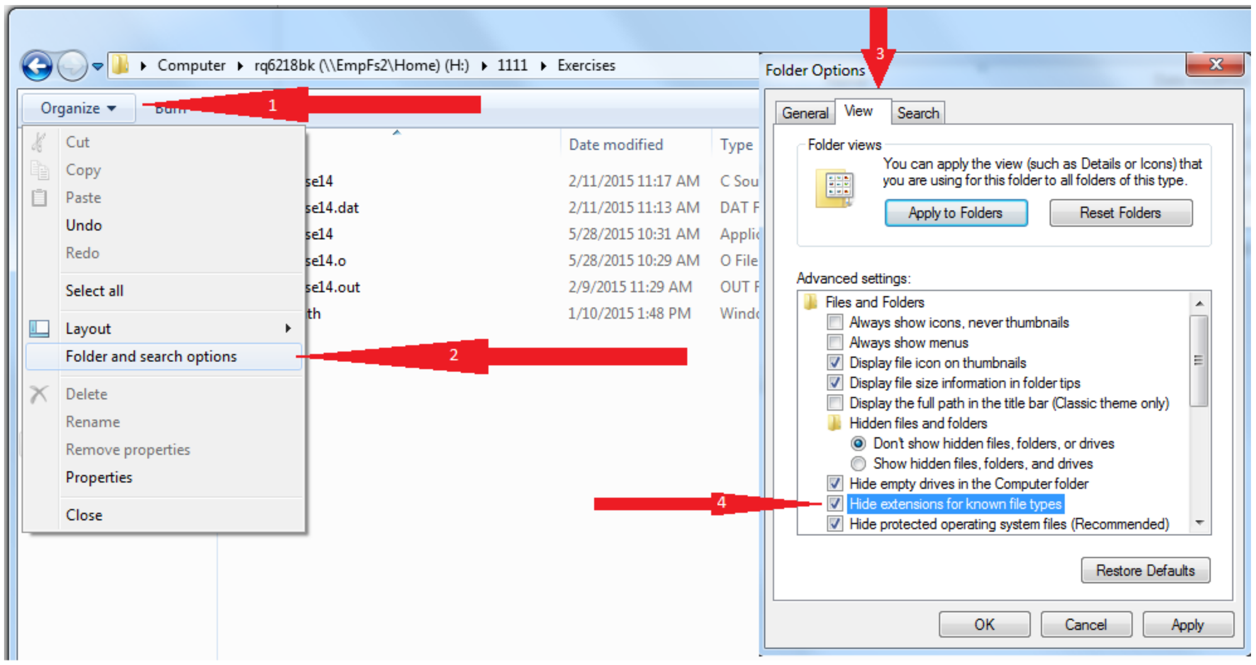
(See "True" file names.)



Without extensions showing, both files have the same name, which causes confusion when moving, opening, or submitting files.

- On Windows computers Windows Explorer is often set to hide file extensions. As programmers this can be a major inconvenience.

  To turn file extensions on:
  1) Go to the "Organize" tab on Windows Explorer:
  2) Select "Folder and search options"
  3) Select the "View" tab
  4) Uncheck the "Hide extensions for known file type" option
  5) Click the "Apply" button
  6) Click "OK"

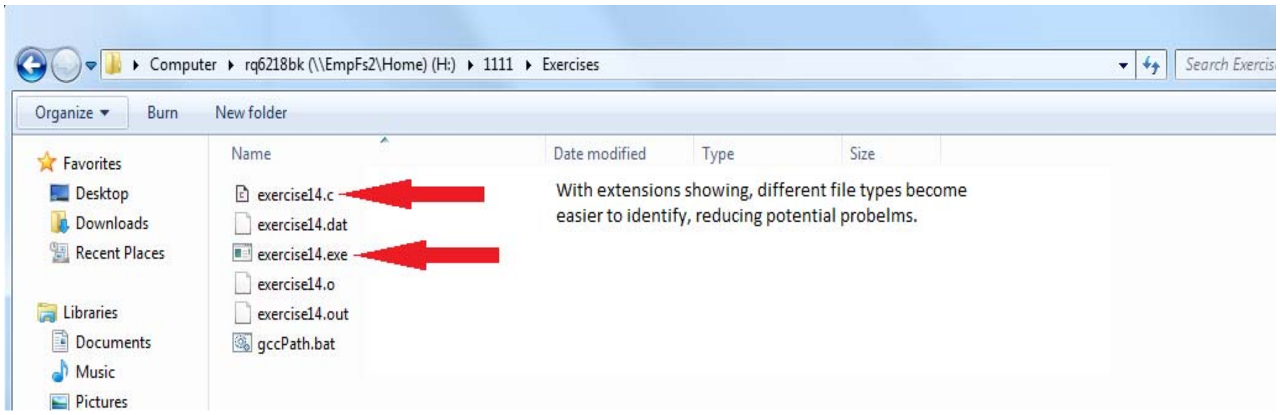


se14	2/11/2015 11:17 AM	C Sou
se14.dat	2/11/2015 11:13 AM	DAT F
se14	5/28/2015 10:31 AM	Applic
se14.o	5/28/2015 10:29 AM	O File
se14.out	2/9/2015 11:29 AM	OUT F
th	1/10/2015 1:48 PM	Windc

You can apply the view (such as Details or Icons) that you are using for this folder to all folders of this type.

Files and Folders
- [ ] Always show icons, never thumbnails
- [ ] Always show menus
- [✓] Display file icon on thumbnails
- [✓] Display file size information in folder tips
- [ ] Display the full path in the title bar (Classic theme only)
- Hidden files and folders
  - [●] Don't show hidden files, folders, or drives
  - [○] Show hidden files, folders, and drives
- [✓] Hide empty drives in the Computer folder
- [✓] Hide extensions for known file types
- [✓] Hide protected operating system files (Recommended)

Results:



exercise14.c
exercise14.dat
exercise14.exe
exercise14.o
exercise14.out
gccPath.bat

With extensions showing, different file types become easier to identify, reducing potential probelms.

APPENDIX M : COMMAND SUMMARY

Below is a brief summary of some basic commands that may be of use to you.

Also, remember that you can type:

Windows:

help *command*

Unix:

man *command*

To get some brief help on these and other commands.

Directory/folder manipulation Commands

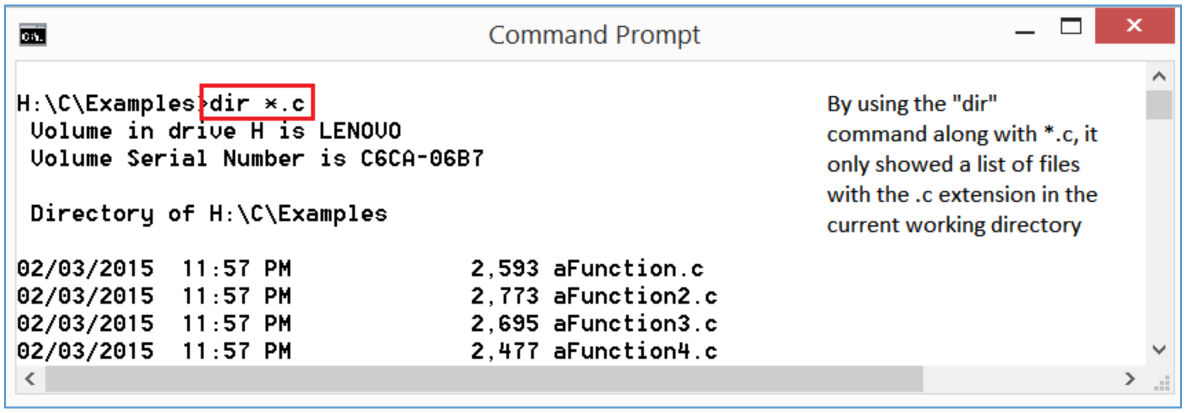
Unix	Function	Windows
	change current/default drive	*drive*:
pwd	display the current working directory/folder	cd/chdir
cd	change the current working directory/folder	cd/chdir
mkdir	create a directory/folder	md
rmdir	remove a directory/folder	rd

File manipulation Commands

Unix	Function	Windows
ls	list files	dir
cp	make a copy of a file	copy
rm	remove/delete a file	del
mv	move a file to a new location/name	rename/move
cat	display the contents of a file	type
more/less	page the display of a files contents	more
chmod	change file access permissions	attrib

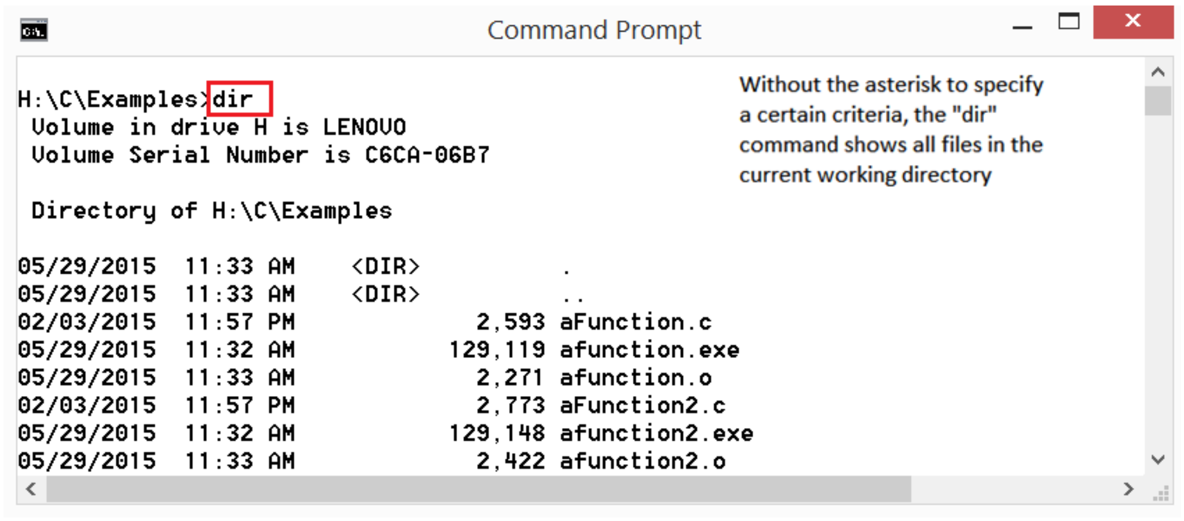
Command Line “Wild Card”:

- The command line “wild card” is denoted by using the asterisk (*****)
- It allows manipulation, selection, and viewing of files with common names and/or extensions



```
H:\C\Examples>dir *.c
 Volume in drive H is LENOVO
 Volume Serial Number is C6CA-06B7

 Directory of H:\C\Examples

02/03/2015  11:57 PM             2,593 aFunction.c
02/03/2015  11:57 PM             2,773 aFunction2.c
02/03/2015  11:57 PM             2,695 aFunction3.c
02/03/2015  11:57 PM             2,477 aFunction4.c
```

By using the "dir" command along with *.c, it only showed a list of files with the .c extension in the current working directory

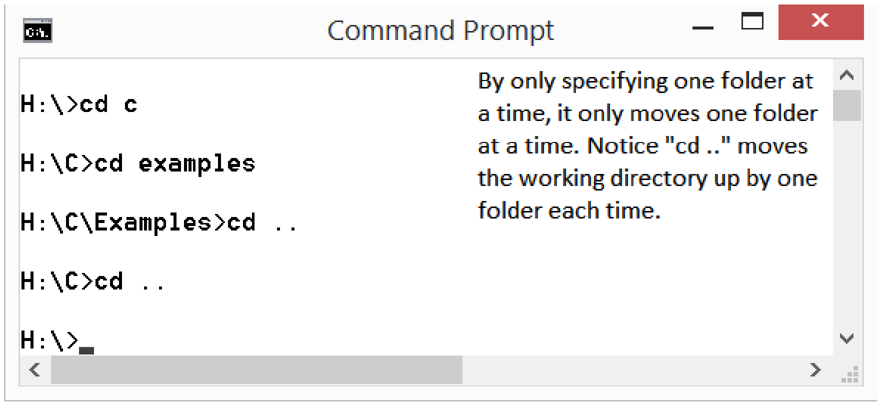


```
H:\C\Examples>dir
 Volume in drive H is LENOVO
 Volume Serial Number is C6CA-06B7

 Directory of H:\C\Examples

05/29/2015  11:33 AM    <DIR>          .
05/29/2015  11:33 AM    <DIR>          ..
02/03/2015  11:57 PM             2,593 aFunction.c
05/29/2015  11:32 AM           129,119 afunction.exe
05/29/2015  11:33 AM             2,271 afunction.o
02/03/2015  11:57 PM             2,773 aFunction2.c
05/29/2015  11:32 AM           129,148 afunction2.exe
05/29/2015  11:33 AM             2,422 afunction2.o
```

Without the asterisk to specify a certain criteria, the "dir" command shows all files in the current working directory

Using Directory/folder manipulation Commands:

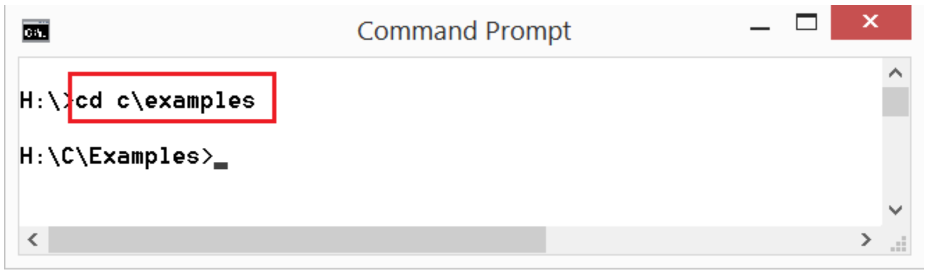
- There are a few different ways you can change your current working directory, here are two:
    - To navigate within, or close to, your current working directory:
        - Type "cd .." to move "up" one folder in your directory
        - Type "cd foldername" to continue to go deeper into your current directory



```
H:\>cd c

H:\C>cd examples

H:\C\Examples>cd ..

H:\C>cd ..

H:\>_
```

By only specifying one folder at a time, it only moves one folder at a time. Notice "cd .." moves the working directory up by one folder each time.

- If you already know exactly where you want to go, and do not want to navigate in a step-by-step manner, you can specify the entire desired path:
    - Type "cd foldername\foldername\foldername"



```
H:\>cd c\examples

H:\C\Examples>_
```

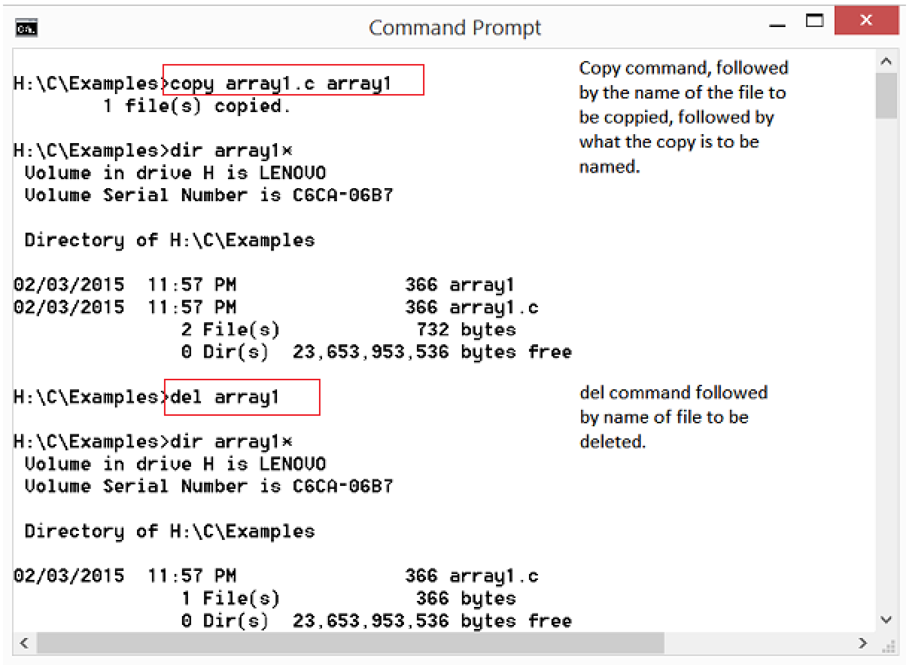
- To move to a location that is NOT relative to your current location, use an absolute path. Absolute paths begin with a "\".

    Example: cd \C\Examples

    Changes your working directory to \C\Examples regardless of your current directory

Using file manipulation Commands:

- Like folder manipulation commands, when using a file command, type the command followed by the file\folder name(s):

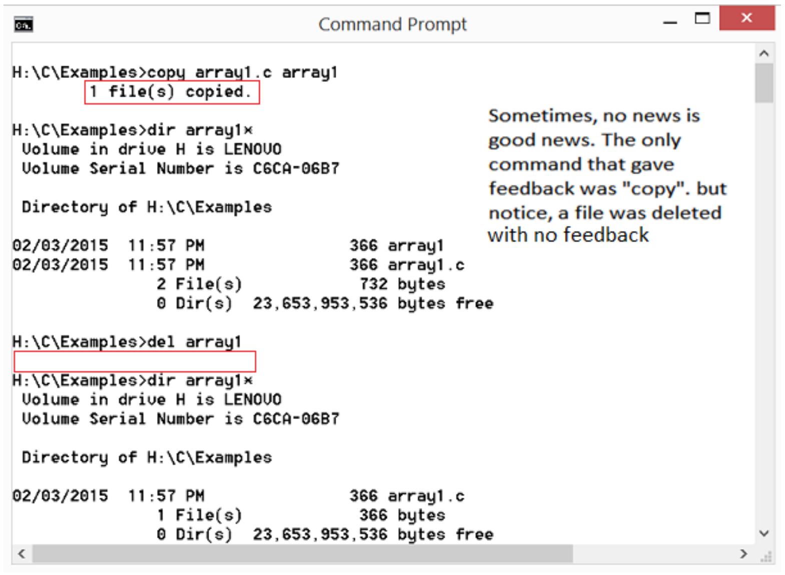


```
H:\C\Examples>copy array1.c array1
        1 file(s) copied.

H:\C\Examples>dir array1*
 Volume in drive H is LENOVO
 Volume Serial Number is C6CA-06B7

 Directory of H:\C\Examples

02/03/2015  11:57 PM               366 array1
02/03/2015  11:57 PM               366 array1.c
               2 File(s)            732 bytes
               0 Dir(s)  23,653,953,536 bytes free

H:\C\Examples>del array1

H:\C\Examples>dir array1*
 Volume in drive H is LENOVO
 Volume Serial Number is C6CA-06B7

 Directory of H:\C\Examples

02/03/2015  11:57 PM               366 array1.c
               1 File(s)            366 bytes
               0 Dir(s)  23,653,953,536 bytes free
```

Copy command, followed by the name of the file to be coppied, followed by what the copy is to be named.

del command followed by name of file to be deleted.

- **Remember:** Successful commands often work silently!



```
H:\C\Examples>copy array1.c array1
        1 file(s) copied.

H:\C\Examples>dir array1*
 Volume in drive H is LENOVO
 Volume Serial Number is C6CA-06B7

 Directory of H:\C\Examples

02/03/2015  11:57 PM               366 array1
02/03/2015  11:57 PM               366 array1.c
               2 File(s)            732 bytes
               0 Dir(s)  23,653,953,536 bytes free

H:\C\Examples>del array1

H:\C\Examples>dir array1*
 Volume in drive H is LENOVO
 Volume Serial Number is C6CA-06B7

 Directory of H:\C\Examples

02/03/2015  11:57 PM               366 array1.c
               1 File(s)            366 bytes
               0 Dir(s)  23,653,953,536 bytes free
```

Sometimes, no news is good news. The only command that gave feedback was "copy". but notice, a file was deleted with no feedback

APPENDIX N : USING OPENGL & GLUT

- OpenGL is an open source library of graphics routines available on many platforms.
- The Dev-C++ distribution comes with the OpenGL libraries. This means you can compile OpenGL code with Dev-C++.
- GLUT is the OpenGL Utility Toolkit. The library and include files for GLUT are not distributed with Dev-C++ but can be obtained on the Web.
    - Visit: *OpenGL.org* on the Web for a vast assortment of code, examples, and tutorials relating to OpenGL and GLUT.

- Several GLUT related files are included in the *examples/OpenGL/lib* and *examples/OpenGL/include/GL* directory for the course, the three we will use are:

glut.h – a header file for the GLUT functions
libfreeglut.a – a library of the GLUT functions
freeglut.dll – a dynamic link library for executing the programs built with these libraries. It should be placed in the same directory as your program.

- The compile command should look as follows:

```
gcc glutsource.c -I..\OpenGL\include -L..\OpenGL\lib -lfreeglut -lglu32 -lopengl32
```

libglu32.a and *libopengl32.a* are OpenGL files included with the Dev-C++ distribution. There are make files for the OpenGL examples in the *OpenGL/Examples* directory. These can be used to build the examples and also as a reference for the compile command, the libraries to use, and the order they are listed.
Note: The order of the files & libraries matters.

- The dynamic link library file *freeglut.dll* must be on the search path for the programs to run. If the current working directory is the examples directory, or the examples directory is in the search path, things will run fine.

- Recommended reading & information sources:

  OpenGL.org - on the web

  OpenGL Super Bible by Wright

  OpenGL Programming Guide by Woo

  OpenGL Reference Manual by Shreiner

Building with make

Command:

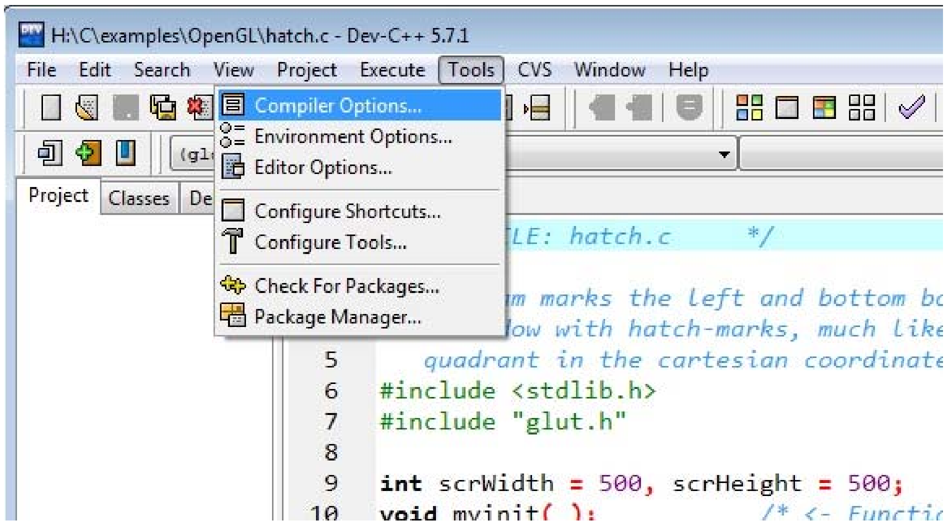
```
H:\c \examples\OpenGL>make -f glutsource.mak
gcc  -c  -I..\OpenGL\include  glutsource.c
gcc -L..\OpenGL\lib -o glutsource glutsource.o -lfreeglut -lglu32 -lopengl32 -mwindows
```

Make file:

```
# ********************************************************
# *** A make file to build the opengl, GLUT files        ****
# ********************************************************
glutsource.exe:           glutsource.o
        gcc -L..\OpenGL\lib -o glutsource glutsource.o -lfreeglut -lglu32 -lopengl32 -mwindows
glutsource.o:   glutsource.c
        gcc -c  -I..\OpenGL\include  glutsource.c
```

Building within the Dev-C++ IDE:

Under *Tools* select *Compiler Options*:



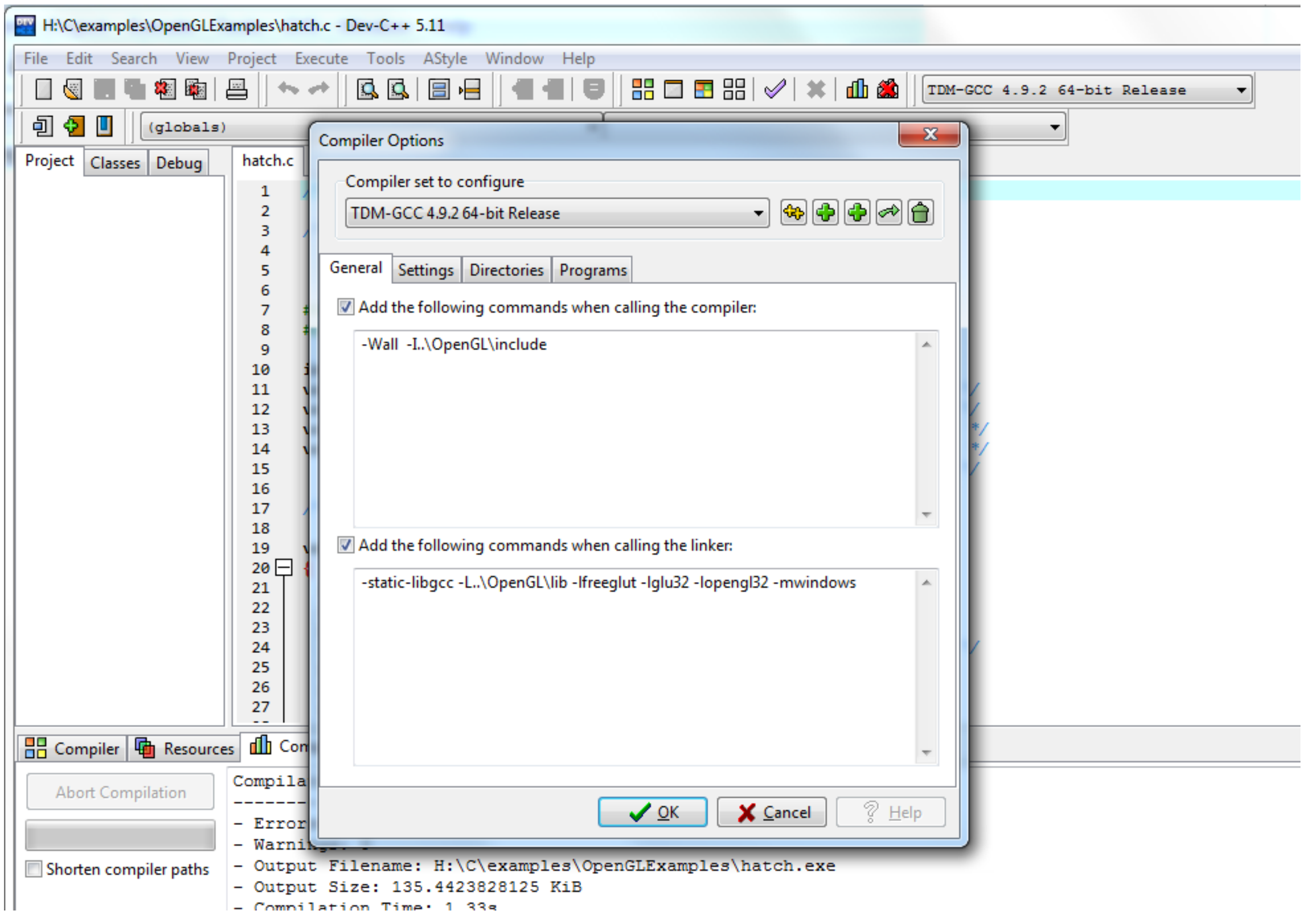
```
                                LE: hatch.c     */

                                m marks the left and bottom bo
                                ow with hatch-marks, much like
   quadrant in the cartesian coordinate
#include <stdlib.h>
#include "glut.h"

int scrWidth = 500, scrHeight = 500;
void myinit( );          /* <- Functic
```

In the *Add ... when calling the compiler* field; check √ the box and type:

-I..\OpenGL\include

In the *Add ... when calling the linker* field; check √ the box and type:

-L..\OpenGL\lib -lfreeglut -lglu32 -lopengl32 -mwindows


H:\C\examples\OpenGLExamples\hatch.c - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
hatch.c
Compiler Options
Compiler set to configure
TDM-GCC 4.9.2 64-bit Release
General Settings Directories Programs
Compiler Resources
Abort Compilation
Shorten compiler paths
OK Cancel Help

[✓] Add the following commands when calling the compiler:

```
-Wall  -I..\OpenGL\include
```

[✓] Add the following commands when calling the linker:

```
-static-libgcc -L..\OpenGL\lib -lfreeglut -lglu32 -lopengl32 -mwindows
```

```
- Output Filename: H:\C\examples\OpenGLExamples\hatch.exe
- Output Size: 135.4423828125 KiB
```

APPENDIX O : ASCII CHARACTER SET

0 (nul)	16 (dle)	32 (sp)	48 0	64 @	80 P	96 `	112 p
1 (soh)	17 (dc1)	33 !	49 1	65 A	81 Q	97 a	113 q
2 (stx)	18 (dc2)	34 "	50 2	66 B	82 R	98 b	114 r
3 (etx)	19 (dc3)	35 #	51 3	67 C	83 S	99 c	115 s
4 (eot)	20 (dc4)	36 $	52 4	68 D	84 T	100 d	116 t
5 (enq)	21 (nak)	37 %	53 5	69 E	85 U	101 e	117 u
6 (ack)	22 (syn)	38 &	54 6	70 F	86 V	102 f	118 v
7 (bel)	23 (etb)	39 ‘	55 7	71 G	87 W	103 g	119 w
8 (bs)	24 (can)	40 (	56 8	72 H	88 X	104 h	120 x
9 (tab)	25 (em)	41)	57 9	73 I	89 Y	105 i	121 y
10 (lf)	26 (eof)	42 *	58 :	74 J	90 Z	106 j	122 z
11 (vt)	27 (esc)	43 +	59 ;	75 K	91 [	107 k	123 {
12 (np)	28 (fs)	44 ,	60 <	76 L	92 \	108 l	124 |
13 (cr)	29 (gs)	45 -	61 =	77 M	93]	109 m	125 }
14 (so)	30 (rs)	46 .	62 >	78 N	94 ^	110 n	126 ~
15 (si)	31 (us)	47 /	63 ?	79 O	95 _	111 o	127

APPENDIX P : MAKE UTILITY

- The *make* utility allows easy construction/compilation of applications involving many source files.
- *make* reads a set of rules from a "make" file that describes what pieces need to be combined to produce the final product, and any dependencies between those pieces.
- *make* is a very powerful utility and can do much more than will be described here.
- Lines in a basic make file consist of commands that describe how to build the parts that make something, and dependencies that describe which parts to make first.
- *make* by default uses a file named *makefile* in the current working directory to determine how to build something. You can tell *make* to use a file of another name.

  - *Note:* You MUST use actual <*tab*> characters to indent/separate rules and dependencies

Ex:

```
# **********************************************************
# *** A make file to build the opengl, GLUT files       ****
# **********************************************************
glutsource.exe:    glutsource.o
                   gcc -L..\OpenGL\lib -o glutsource glutsource.o -lfreeglut -lglu32 -lopengl32

glutsource.o:      glutsource.c
                   gcc -c -I..\OpenGL\include glutsource.c
```

➢ Result when only the source code exists:

```
H:\c\examples\OpenGLExamples>make -f glutsource.mak
gcc -c -I..\OpenGL\include glutsource.c
gcc -L..\OpenGL\lib -o glutsource glutsource.o -lfreeglut -lglu32 -lopengl32
```

➢ Result when previously compiled and source code is unchanged:

```
H:\c\examples\OpenGLExamples>make -f glutsource.mak
make: `glutsource.exe' is up to date.
```

Ex:

```
# *************************************************************
# *** A   make file to build a set of interdependent files.****
# *************************************************************
extern3.exe:     extern3.o extern2.o
                 gcc -o extern3 extern3.o extern2.o
extern3.o:extern3.c
                 gcc -c extern3.c
extern2.o:extern2.c
                 gcc -c extern2.c
```

- Result when only the source code exists:

```
H:\c\examples>make -f extern3.mak
gcc -c extern3.c
gcc -c extern2.c
gcc -o extern3 extern3.o extern2.o
```

- Result when previously compiled and source code is unchanged:

```
H:\c\examples >make -f extern3.mak
make: `extern3.exe' is up to date.
```

- Result when *extern2.o* is not present:

```
H:\c\examples>del extern2.o

H:\c\examples>make -f extern3.mak
gcc -c extern2.c
gcc -o extern3 extern3.o extern2.o
```