FIT ICT Software Development

Lecture 3

Writing and Compiling a C Program

Lecturer : Charles Tyner

FIT ICT Software Development

- In C programs, the code to be executed is contained within "functions", which are defined using the following syntax

  data-type function-name(){ statements-to-be-executed}

- After a function has executed the code statements it contains, it can return a value to the caller

- Value must be of data-type specified before function name

FIT ICT Software Development

- Program can contain one or many functions but must always have a function called "main"

- The **main()** function is the starting point of all C programs

- If compiler doesn't find a **main()** function, program will not be compiled

FIT ICT Software Development

- Other function can be given any name within a small set of restrictions
  - must not begin with a digit
  - must not use reserved words
- Apart from these restrictions, a function name can be any combination of letters, digits and the underscore character
- As always, developers are encouraged to use suitably informative names for functions

FIT ICT Software Development

The 32 words listed in the table below are the keywords of the C programming language. They have special significance in C programs and may not be used for any other purpose.

auto	break	case	char
const	continue	default	do
double	else	enum	extern
float	for	goto	if
int	long	register	return
short	signed	sizeof	static
struct	switch	typedef	union
unsigned	void	volatile	while

FIT ICT Software Development

- The ( ) parentheses that follow the function name may, optionally, contain values to be used by that function.

- Take the form of a comma-separated list and are known as function arguments or parameters.

- The { } curly brackets (braces) contain the statements to be executed whenever the function is called

- Each statement terminated by a semi-colon

FIT ICT Software Development

- By tradition, the first program written in any new language is one that simply generates the message "Hello World"

- Open a plain text editor (e.g. Notepad)

- Type:

#include <stdio.h>

This is an instruction to the compiler to include information from the standard input/output library file. This makes all the functions contained within that library available to the program

FIT ICT Software Development

- The instruction **#include** is a **preprocessor instruction**

- Preprocessor instructions must always appear at the start of a program, before the actual program code is processed

- Preprocessor instructions must always be preceded by the # hash symbol

- Standard library names must be enclosed within < > angled brackets

FIT ICT Software Development

- Two lines below the preprocessor instructions, add an empty main function

```
int main()
{

}
```

- Function declaration specifies that an integer value of data type **int** should be returned by the function on completion

FIT ICT Software Development

- Between the curly brackets, insert a line of code that calls one of the functions defined in the standard input/output library

  printf ("Hello World!\n");

- The printf() function takes a single string argument

- In C, string s must always be enclosed within double quotes

- The string contains the text to be printed and the **\n** escape sequence that produces a new line

FIT ICT Software Development

- After the **printf()** call, insert a final line of code to return a zero integer value as required by the function declaration:

  return 0;

- By tradition, returning a value of zero after the execution of a program indicates to the operating system that the program executed correctly

FIT ICT Software Development

- Check that the program code looks exactly like the listing below

```
#include <stdio.h>

int main()
{
printf(("Hello World!\n");
return  0;
}
```

- Make sure to add a final newline character by hitting the Return key after the closing curly bracket.

FIT ICT Software Development

Compiling a C Program

1. Create a directory to store your C programs e.g. C:\MyPrograms. Make sure there are no white spaces in the name.

2. Open a command prompt and navigate to this directory

3. Type **gcc hello.c** and hit Return to compile the program

FIT ICT Software Development

When the compilation succeeds, the compiler creates an executable file alongside the original source code file.
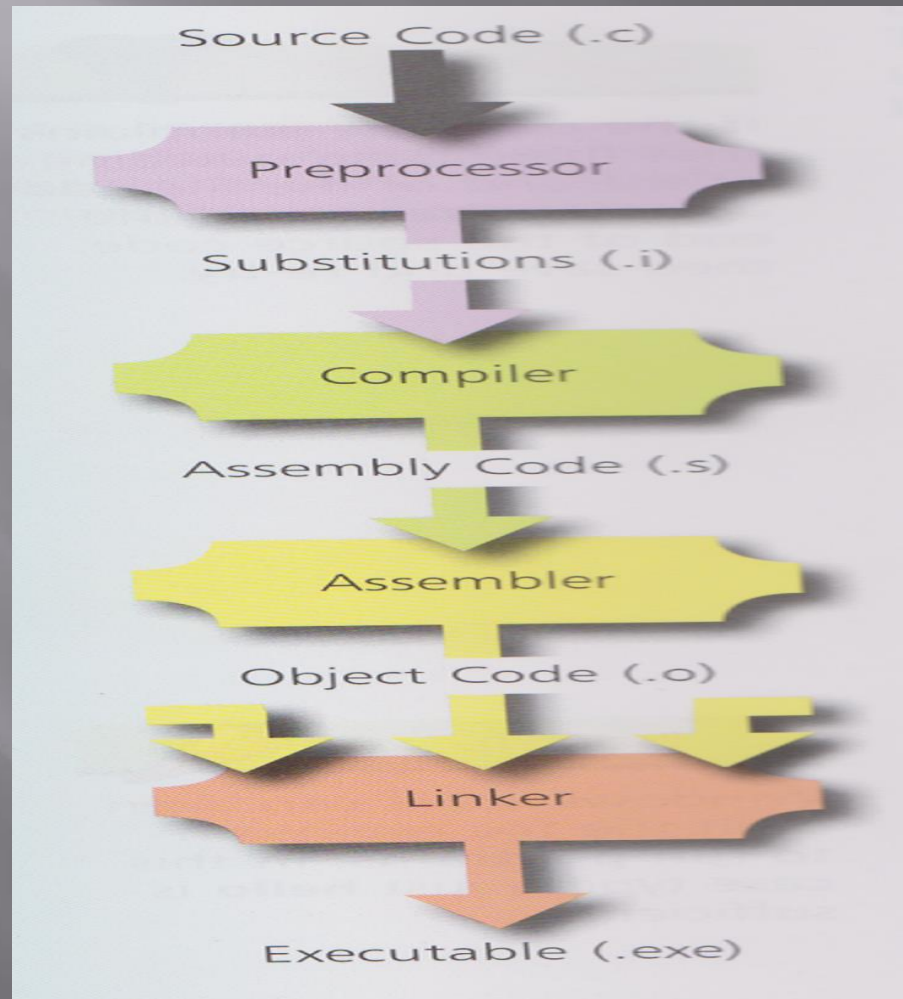
By default, this file will be named **a.exe**

Compiling a different C source file in the same directory would overwrite the existing executable **without warning**.

Obviously unsatisfactory

Therefore, must specify a custom name when compiling **hello.c**

    gcc hello.c –o hello.exe

FIT ICT Software Development

FIT ICT Software Development

In producing an executable, compilation undergoes 4 separate stages:

o Preprocessing – substitutes all pp directives with actual library code. Generated file is in text format and typically has a .i file extension

o Translating – high-level language in .i file translated into Assembly language instructions. Text format, typically .s extension

o Assembling – assembler converts text of .s file into machine code. Binary format, .o extension

o Linking – linker combines one or more .o files into a single executable. Binary, .exe

FIT ICT Software Development

- Strictly speaking, compilation is the first 3 stages.

- Operates on a single source text file and generates a single binary object

- Syntax errors will cause compilation to fail

- Linker can operate on multiple object files

- Produces a single executable file

- Allows creation of large programs from modular object files

- If linker finds function of same name defined in multiple object files, error reported and no executable created

FIT ICT Software Development

- Temporary files created during compilation automatically deleted by default.

- Can be saved for inspection by including a **-save-temps** switch in compile command

1. In directory containing .c file, type

   gcc hello.c –save-temps –o hello.exe

2. Open **hello.i** in Notepad and view. Your code at very end of file, preceded by substituted **stdio.h** library code

3. Now examine **hello.s** file in Notepad.