Data Structures in C

Prof. Georg Feil

C Program Structure

Summer 2018

Acknowledgement

- These lecture slides are partly based on slides and other material by Professor Magdin Stoica
- Additional sources are cited separately

Reading Assignment (required)

- C for Programmers (supplementary textbook)
  - Sections 1.1 – 1.5
  - Sections 2.1 – 2.5

Statements and Blocks

C Statement Blocks

- Blocks are used to organize statements
- Statement blocks have a start marker that marks the start of the block, and an end marker that marks the end of the block
- A block of statements is a compound statement
  - Allows programs to treat a group of statements as one
- Sometimes called a scope because it controls visibility of variables
- A block of statements can contain other blocks
  - A block of statements can also be empty

Indenting

- Statements in a block are indented to show they are part of the block
  - This makes your program easier to read and understand
  - You'll make fewer mistakes if you indent properly!
- The computer (compiler) does not care if you indent, but other programmers (and your boss) do
  - It's good programming practice and part of our Coding Standard
- In this course you should indent each block by 4 spaces
  - If there's another statement block inside, indent by 4 more spaces... etc.
- I will deduct marks for incorrect/inconsistent indenting

C Program Structure Basics

- The statement terminator in C is the semicolon **;**
- A C statement block is placed inside curly brackets { … }
  - "{" starts a statement block (start marker)
  - "}" ends a statement block (end marker)
- A statement block can be named, as with functions, or unnamed
  - Branching (if) statements and loops have unnamed blocks
  - You can start a new unnamed block or scope anywhere in a function
- The C programming language is case sensitive: name, Name, NAME and nAmE are four different things
  - File names are not case sensitive in Windows, but I encourage you to keep them consistent anyway

➢ All of this is the same as Java! (why?)

Recall our first C program

```
#include <stdio.h>

/* This is our first program in C */
int main(int argc, char** argv)
{
    printf("Hello, my name is Georg.\n");
    printf("I hope this C program works!\n");

    return 0;
}
```

Statement block

A C program with a nested statement block

```
#include <stdio.h>

/* Program to demonstrate a loop with output */
int main(int argc, char** argv)
{
    printf("This program prints numbers from 1 to 10.\n");

    for (int count = 1; count <= 10; count++) {
        printf("%d\n", count);
    }

    return 0;
}
```

Comments

- Block comments can span multiple lines
  - Start with "/*"
  - End with "*/"
  - Anything in between is considered a comment. Anything goes except the end of comment marker, */ (comments can't be "nested")
  - Example:

```
/* This is a multi-line
   comment. */
```

- Single line comments (allowed in C since the C99 standard)
  - Start with "//"
  - End at the end of the line. It doesn't matter what is in the comment, the rest of the text until the end of the line is considered a comment
  - Examples:

```
// This is a comment on one line

//Some developers prefer these types of comments
//even when they span multiple lines
```

Exercise 1

- Download the program scope.c from SLATE (week 1)
- Open it in Dev-C++ and try it out.
  - Explain the output
- Un-comment the printf that's commented out and try compiling
  - Explain why you get an error

Other ways to give your programs structure

- As programs get larger, you should divide your code into different functions
  - Functions are similar to methods in Java
- You should also divide large programs into different source files (modules)
  - A large program may consist of hundreds of .c files
- We'll talk more about functions and modules in C next week… modular programming!