UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING
Department of Electrical &
Computer Engineering

ECE 150 *Fundamentals of Programming*

Anatomy of a program

Prof. Hiren Patel, Ph.D. P.Eng.
Prof. Werner Dietl, Ph.D.
Douglas Wilhelm Harder, M.Math. LEL

Outline

- In this presentation, we will:
  - Define the components of a program
    - Pre-processor directives
    - Statements
    - Blocks of statements
    - Function declarations and definitions

Pre-processor directives

```
#include <iostream>

int main();

int main() {
      std::cout << "Hello world!";
      std::cout << std::endl;

      return 0;
}
```

- Indicates that a particular standard library or file should be included in the compilation
  - C++ Standard libraries contain functionality available to all programmers except perhaps in embedded systems
  - Possible to include other source code you or others have written
  - All pre-processor directives start with a "#"
  - Pre-processor directives are not part of the C++ programming language

Statements

```
#include <iostream>

int main();

int main() {
        std::cout << "Hello world!";
        std::cout << std::endl;

        return 0;
}
```

```
Valid = FALSE;
BytesRemaining = 0;
MM_AppData.ErrCounter++;
CFE_EVS_SendEvent(MM_PSP_READ_ERR_EID, CFE_EVS_ERROR,
                  "PSP read memory error: RC=0x%08X, "
                  "Src=0x%08X, Tgt=0x%08X, Type=MEM16",
                   (unsigned int)PSP_Status,
                   (unsigned int)DataPointer16,
                   (unsigned int)&ioBuffer16[i] );
break;
```

Statements from the NASA core Flight System Memory Manager Application

- A statement may always be described a
  - the introduction (or "*declaration*") of a name (or "*identifier*")
  - an action that is being performed on data
- A statement is always terminated by a semi-colon

Function declarations and definitions

```
#include <iostream>

int main();

int main() {
        std::cout << "Hello world!";
        std::cout << std::endl;

        return 0;
}
```

Function declaration

Function definition

A function is the unit of execution in C++

The statements executed when the function is called is also referred to as the *function body*.

Function declarations

- From your secondary school mathematics courses, you have seen:

$$\sin\left(\frac{\pi}{6}\right) \qquad \gcd(91,119)$$

- The names of these functions are sin and gcd:
  - sin has one real parameter and evaluates to a real
  - gcd has two integer parameters and evaluates to an integer
- A *function declaration* indicates to the compiler:
  - That a function with a specific name exists
    - The name is called the *identifier* of the function
  - The information that must be passed (the parameters)
  - What results, or what is returned

```
int main();
double sin( double x );
int gcd( int m, int n );
```

Function definition

- A *function definition* is the function declaration immediately followed by a block of statements
  - This block of statements is also called the *body of the function*
  - These statements are executed when the function is called
    - Or "run" or "executed"
  - The three statements executed when `main()` is called include:
    - Printing "Hello world!" to the console output,
    - Printing an end-of-line character to the console output, and
    - Returning the integer `0`

```
int main() {
        std::cout << "Hello world!";
        std::cout << std::endl;

        return 0;
}
```

Function definition

- The *main* function is especially important in C++
  - There can by many functions, but if source code is compiled into an executable, when that executable is run, it is the `main()` function that is first called

Matching delimiters

- As in your mathematics courses, `( )`, `[ ]`, and `{ }` are used to group mathematical expressions
  - We will call these parentheses, brackets and braces
    - For clarity, we may use the terms *round parentheses*, *square brackets*, and *curly braces*
  - (, [, and { are referred to as opening delimiters
    while),], and } are referred to as closing delimiters

- Given an opening delimiter, its *matching* closing delimiter is next closing delimiter that does not have a closer matching opening delimiter
  - Each opening delimiter must have a matching closing delimiter

```
{ ( [ ( ( ) [ ( ) ] ) { } ] { } [ ] ) } [ ( ) { [ ] } ]
```

Summary

- In this presentation, you now
  - Described the include pre-processor directive
  - Defined a statement
    - Operators and function calls terminated by a semi-colon
  - Defined a block of statements
    - Zero or more statements surrounded by braces
  - Defined function declarations and definitions

References

[1] Wikipedia
https://en.wikipedia.org/wiki/Statement_(computer_science)

Acknowledgments

Proof read by Dr. Thomas McConkey and Charlie Liu.

Colophon

These slides were prepared using the Georgia typeface. Mathematical equations use Times New Roman, and source code is presented using `Consolas`.

The photographs of lilacs in bloom appearing on the title slide and accenting the top of each other slide were taken at the Royal Botanical Gardens on May 27, 2018 by Douglas Wilhelm Harder. Please see

https://www.rbg.ca/

for more information.

Disclaimer