CS218- Data Structures
Week 01

Muhammad Rafi
August, 22 2019

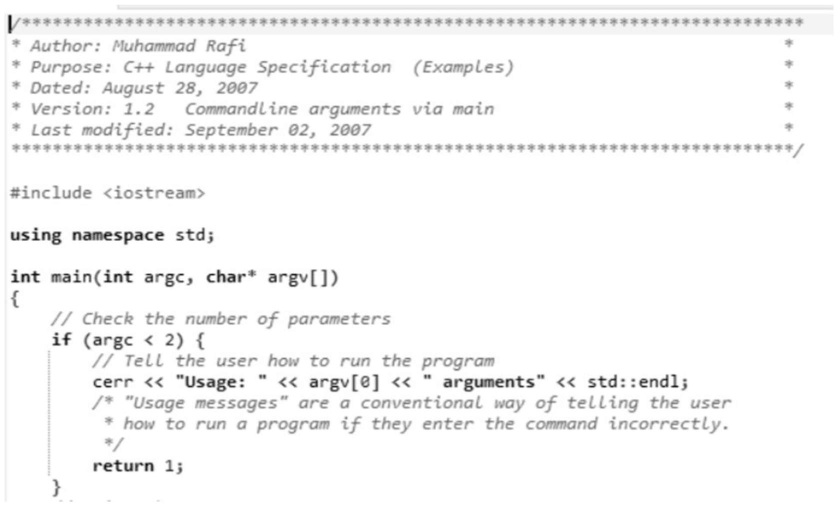
Agenda

- C++ Language Specification
  - Comments and Style
  - Data Types
  - Identifiers and Naming
  - Expression and Assignment
  - Operators
  - Selection
  - Repetition
  - Pointers
  - Functions
  - Function Pointers

Comments & Style

- Programming is an intellectual activity. The code has a life and there are people who interested in reviewing and learning from it.
- Comments
  - Statement
  - Line
  - Block
  - Author
  - Functional /Class

Comments & Style



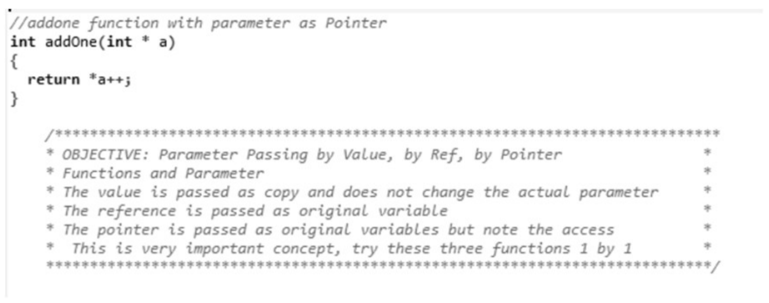
```
/*****************************************************************************
* Author: Muhammad Rafi                                                      *
* Purpose: C++ Language Specification  (Examples)                            *
* Dated: August 28, 2007                                                     *
* Version: 1.2   Commandline arguments via main                              *
* Last modified: September 02, 2007                                          *
*****************************************************************************/

#include <iostream>

using namespace std;

int main(int argc, char* argv[])
{
    // Check the number of parameters
    if (argc < 2) {
        // Tell the user how to run the program
        cerr << "Usage: " << argv[0] << " arguments" << std::endl;
        /* "Usage messages" are a conventional way of telling the user
         * how to run a program if they enter the command incorrectly.
         */
        return 1;
    }
```

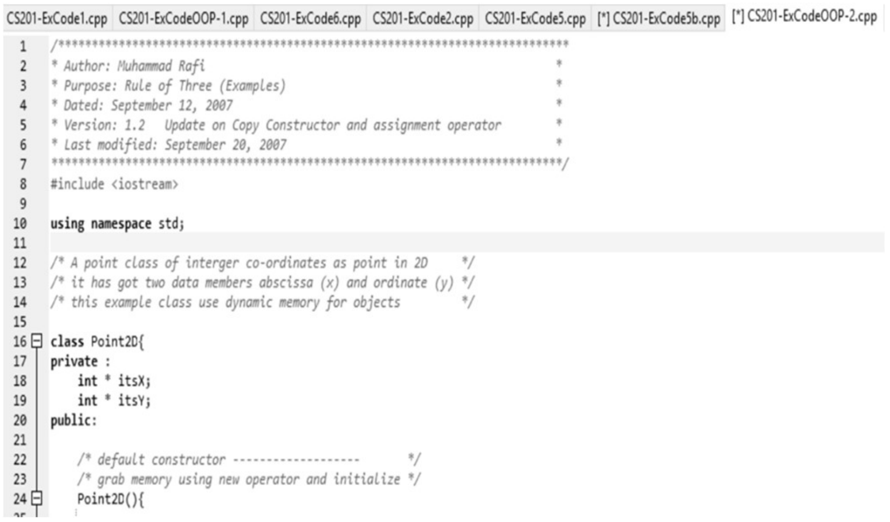
Comments & Style



```
//addone function with parameter as Pointer
int addOne(int * a)
{
  return *a++;
}

    /*******************************************************************************
    * OBJECTIVE: Parameter Passing by Value, by Ref, by Pointer                    *
    * Functions and Parameter                                                      *
    * The value is passed as copy and does not change the actual parameter         *
    * The reference is passed as original variable                                 *
    * The pointer is passed as original variables but note the access              *
    *  This is very important concept, try these three functions 1 by 1            *
    *******************************************************************************/
```

Comments & Style



```
/****************************************************************************
* Author: Muhammad Rafi                                                     *
* Purpose: Rule of Three (Examples)                                         *
* Dated: September 12, 2007                                                 *
* Version: 1.2   Update on Copy Constructor and assignment operator         *
* Last modified: September 20, 2007                                         *
****************************************************************************/
#include <iostream>

using namespace std;

/* A point class of interger co-ordinates as point in 2D     */
/* it has got two data members abscissa (x) and ordinate (y) */
/* this example class use dynamic memory for objects         */

class Point2D{
private :
    int * itsX;
    int * itsY;
public:

    /* default constructor ------------------       */
    /* grab memory using new operator and initialize */
    Point2D(){
```

Data Types

- Intrinsic / Build-in types/ Atomic types
  - These types are available with compilers to process data. These are well-define and atomic in nature.
  - There are 5 types in C/C++: char, integer, float, double and void. Some more type extended in latest version of C/C++
- Users Define Types
  - struct / class / union
  - We will talk about classes in details soon.

Identifier and Naming

- Identifiers consists of letters, digits and underscore characters.
- Identifier must begin with a letter or underscore. Identifier with two-underscore are reserved for the system.
- Identifier are case sensitive names.
- Identifier can be of any length(but first 32 characters are significant.
- Identifier can not be keywords or reserved words from C/C++
- Compilers does not issue an error or warning for missing these rules.

Keywords

asm	do	if	return	typedef
auto	double	inline	short	typeid
bool	dynamic_cast	int	signed	typename
break	delete	long	sizeof	union
case	else	mutable	static	unsigned
catch	enum	namespace	static_cast	using
char	explicit	new	struct	virtual
class	extern	operator	switch	void
const	false	private	template	volatile
const_cast	float	protected	this	wchar_t
continue	for	public	throw	while
default	friend	register	true	union
delete	goto	reinterpret_cast	try	unsigned

Expression

- Expressions are sequences of operators, operands, and punctuators that specify a computation.
- Expression are computed with an standard approach for preference to computation.
- Using operator precedence and associativity every expression is unambiguously evaluated to a single values.
- Data types are promoted with an standard approach.

Assignment (=)

- Assignments are used to hold values from the expression.
- Lvalue vs. Rvalue

Operators

- Operators that compute: {+,-,*,/,%,unary (+,-)}
- Operators that make decisions: { >,<, >=, <=, ==, !=} { &&, ||, !}
- Conditional operator (?:)
- Logical operators { &&, ||, !}
- Bitwise Operator { &, |,^,~, <<, >}
- C/C++ is very rich in Operators

Operator precedence

Category	Operator	Associativity
Postfix	() [] -> . ++ - -	Left to right
Unary	+ - ! ~ ++ - - (type) * & `sizeof`	Right to left
Multiplicative	* / %	Left to right
Additive	+ -	Left to right
Shift	<< >>	Left to right
Relational	< <= > >=	Left to right
Equality	== !=	Left to right
Bitwise AND	&	Left to right
Bitwise XOR	^	Left to right
Bitwise OR	|	Left to right
Logical AND	&&	Left to right
Logical OR	||	Left to right
Conditional	?:	Right to left
Assignment	= += -= *= /= %= >>= <<= &= ^= |=	Right to left
Comma	,	Left to right

Promotion Hierarchy

Data types	
`long double`	
`double`	
`float`	
`unsigned long int`	(synonymous with `unsigned long`)
`long int`	(synonymous with `long`)
`unsigned int`	(synonymous with `unsigned`)
`int`	
`unsigned short int`	(synonymous with `unsigned short`)
`short int`	(synonymous with `short`)
`unsigned char`	
`char`	
`bool`	(`false` becomes 0, `true` becomes 1)

Fig. 3.5 Promotion hierarchy for built-in data types.

Operators

You can overload any of the following operators:

+	-	*	/	%	^	&	|	~
!	=	<	>	+=	-=	*=	/=	%=
^=	&=	|=	<<	>>	<<=	>>=	==	!=
<=	>=	&&	||	++	--	,	->*	->
( )	[ ]	**new**	**delete**	**new[]**	**delete[]**			

You cannot overload the following operators:

. .* :: ?:

Selection

- If (condition) else - statement
- Case-Switch statement
- Ternary operator

Repetition

- While{} – Statement
- Do{} while – Statement
- For () – Statement

Pointers

- Declaration of a pointer
- Assignment of values to a pointer
- De-referencing a pointer for value.
- Pointer Arithmetic

Functions

- Functions hold the executable code of a program with a single identifier (function name)
- A function has a function header and a function body. The function header comprises of three things { return type, name, and parameter list}
- Function declaration, Function definition, function calling.

Functions

- Polymorphic functions/Overloading of functions
- Default Parameters in functions

Function Pointers

- A pointer to the function, it is very handy for a lot of situations.
- Function Pointers can only hold compatible functions.
- return_type (* function_Ptr_name) (parameters)