AI INSTITUTE #AIISC
UNIVERSITY OF SOUTH CAROLINA

UNIVERSITY OF SOUTH CAROLINA

CSCE 240: Advanced Programming Techniques
Lecture 15: Operators, HW 4 (Review)

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

28TH FEBRUARY 2022

Carolinian Creed: "I will practice personal and academic integrity."

Credits: Some material reused with permission of Dr. Jeremy Lewis. Others used as cited with thanks.

Organization of Lecture 15

- Introduction Section
  - Recap of Lecture 14
- Main Section
  - (Peer) Evaluation of Home work #4
  - Concept: Operators
  - Concept: Operator precedences
  - Discussion: Project – PA #3 Check
- Concluding Section
  - About next lecture – Lecture 16
  - Ask me anything

Introduction Section

Recap of Lecture 14

- We explored the concepts of
  - constructors
  - destructors
- Home Work #4 – due Tuesday, Feb 28, 2023
- Programming Assignment #3 starts, due Thursday, March 2, 2023

Main Section

Home Work 4 (Peer Review)

Due Tuesday, Feb 28, 2023

Home Work (#4) – C++ - Background

- Email programs parse Email headers and show content. The headers have **parts** (e.g., CC, To, From) that are part of a standard and also proprietary extensions.
- Examples for Microsoft Outlook and Gmail are shown.
- Let us assume that parts which are common to both are the standard and those unique are proprietary. So, "CC" is common and "X-MS-Has-Attach" is unique.
- Write a program, ***EmailInformationExtractor,*** which, when given a message header from either of the two programs, and a part name, will read the value of the message part.

Microsoft Outlook Header

- Received: from DS7PR19MB5853.namprd19.prod.outlook.com …
- Authentication-Results: dkim=none (message not signed
- Received: from …
- Content-Type: application/ms-tnef; name="winmail.dat"
- Content-Transfer-Encoding: binary
- From: "Sri Naga Sushmitha, Satti" <SATTI@cse.sc.edu>
- To: "Srivastava, Biplav" <BIPLAV.S@sc.edu>
- CC: "Baldwin, Randi" <baldwin@cse.sc.edu>
- Subject: Re: Possible need for … 240
- Thread-Topic: Possible need for printout for .. 240
- Thread-Index: … +AAAIRpoAAAp/ggAAAJH0=
- Date: Tue, 15 Feb 2022 13:52:33 +0000
- Message-ID: <…>
- References: …
- In-Reply-To: <…>
- Accept-Language: en-US
- Content-Language: en-US
- X-MS-Has-Attach:
- X-MS-Exchange-Organization-SCL: -1

Home Work (#4) – C++ - Requirement

- So, program name: ***EmailInformationExtractor***
- Inputs:
  - message header
  - Part name
- Output:
  - Value
- Hint
  - Use regex

Gmail Header

- Delivered-To: biplav.srivastava@gmail.com
- Received: by 2002:a05:7000:1f97:0:0:0:0 with SMTP …
- X-Google-Smtp-Source: ABdhPJz/…
- Received: from m08b.cvent-planner.com …
- From: Reply-To:To:Message-ID:Subject:MIME-Version:
- Content-Type: List-Unsubscribe; /Tvkdd8/15SWlBA=; …
- Date: Thu, 17 Feb 2022 23:56:12 +0000
- From: AAAI Staff <aaai22@aaai.org>
- Reply-To: <aaai22@aaai.org>
- To: Biplav Srivastava <biplav.srivastava@gmail.com>
- Message-ID: <..>
- Subject: AAAI-22 General Information
- MIME-Version: 1.0
- Content-Type: multipart/alternative; ..
- Content-Type: text/plain; charset=UTF-8
- Content-Transfer-Encoding: quoted-printable

Peer Review: Homework Assignment #4

1. Go to spread sheet and on ”Homework Assignments - Peer Review” tab. Go for today’s date
2. Go to the row with your name
3. Peer review (10 mins)
    1. Enter serial number of person on your **LEFT** under “ID of code reviewer”
    2. Share code for the reviewer to see
    3. Reviewer: enter review (1-5)
    4. **Note**: negotiate – review code of neighbor or get own’s code reviewed
4. Peer test (10 mins)
    1. Enter serial number of person on your **RIGHT** under “ID of code tester”
    2. Share command line for the tester to see
    3. Tester: enter review (1-5)
    4. **Note**: negotiate – test code of neighbor or get own’s code tested

Peer Reviewing Guideline (10 mins)

- Look out for
  - Can you understand what the code is doing ?
  - Can you explain the code to someone else (non-coder) ?
  - Can you spot possible issues without running it?
    - Are the variables initialized ?
    - Are files closed?
    - Is their unnecessary code bloat ?
- What not to judge
  - Usage of language features, unless they are inappropriate

Assign rating

1: code not available
2: code with major issues
3: code with minor issues
4: -
5: no issues

Peer Testing Guideline (10 mins)

- Look out for
  - Does the program run as the coder wanted it to be (specification) ?
  - Does the program run as the instructor wanted it to be (requirement - customer) ?
  - Does the program terminate abruptly ?
  - Any special feature?
- What not to judge
  - Person writing the code

Assign rating

1: code not available
2: code runs with major issues (abnormal termination, incomplete features)
3: code runs with minor issues
4: -
5: No issues

Discussion on HW

- Peer Code Reviewing
- Peer Testing

Concept: Operators

Operators - What are They?

- **Built-in special functions** to create **expressions**
- Expressions lead to computation of values
  - Can also cause side-effects
- Operators are employed over expressions called operands. Often operands are constants and variables.
- **Purpose**: create expressions in a compact and consistent, well-understood manner

Types
- Arithmetic
- Relational
- Logical
- Bitwise
- Increment/ decrement
- Assignment
- Conditional expressions
- Others

Operators - Arithmetic

Operator	Name	Example	
+	Addition	`12 + 4.9`	`// gives 16.9`
-	Subtraction	`3.98 - 4`	`// gives -0.02`
*	Multiplication	`2 * 3.4`	`// gives 6.8`
/	Division	`9 / 2.0`	`// gives 4.5`
%	Remainder	`13 % 3`	`// gives 1`

Notes

- Data type defines the nature of results – e.g., integer v/s float
- Remainder expects the two operands to be integers
- Overflow happens when outcome of an arithmetic operation to be too large for storing in a designated variable
- Division by zero error has to be handled; default is to terminate execution

Credits: C++ Essentials by Sharam Hekmat, PragSoft

Operators - Relational

Operator	Name	Example	
==	Equality	`5 == 5`	`// gives 1`
!=	Inequality	`5 != 5`	`// gives 0`
<	Less Than	`5 < 5.5`	`// gives 1`
<=	Less Than or Equal	`5 <= 5`	`// gives 1`
>	Greater Than	`5 > 5.5`	`// gives 0`
>=	Greater Than or Equal	`6.3 >= 5`	`// gives 1`

Notes

- Operands must evaluate to a number
- Comparing character works - 'A' < 'F' // gives 1 (is like 65 < 70)
- Comparing string will compare their addresses –
  ”CSCE" < ”240”, not desirable

Credits: C++ Essentials by Sharam Hekmat, PragSoft

Operators - Logical

Operator	Name	Example
!	Logical Negation	`!(5 == 5)` `// gives 0`
&&	Logical And	`5 < 6 && 6 < 6` `// gives 1`
||	Logical Or	`5 < 6 || 6 < 5` `// gives 1`

Notes

- C++ does not have boolean type
- 0 indicates a false, 1 a true

Credits: C++ Essentials by Sharam Hekmat, PragSoft

Operators - Bitwise

```
unsigned char x = '\011';
unsigned char y = '\027';
```

How the bits are calculated.

Example	Octal Value	Bit Sequence							
x	011	0	0	0	0	1	0	0	1
y	027	0	0	0	1	0	1	1	1
~x	366	1	1	1	1	0	1	1	0
x & y	001	0	0	0	0	0	0	0	1
x | y	037	0	0	0	1	1	1	1	1
x ^ y	036	0	0	0	1	1	1	1	0
x << 2	044	0	0	1	0	0	1	0	0
x >> 2	002	0	0	0	0	0	0	1	0

Operator	Name	Example	
~	Bitwise Negation	`~'\011'`	`// gives '\366'`
&	Bitwise And	`'\011' & '\027'`	`// gives '\001'`
|	Bitwise Or	`'\011' | '\027'`	`// gives '\037'`
^	Bitwise Exclusive Or	`'\011' ^ '\027'`	`// gives '\036'`
<<	Bitwise Left Shift	`'\011' << 2`	`// gives '\044'`
>>	Bitwise Right Shift	`'\011' >> 2`	`// gives '\002'`

Credits: C++ Essentials by Sharam Hekmat, PragSoft

Code Sample

https://github.com/biplav-s/course-adv-proglang/blob/main/sample-code/CandC%2B%2B/Class15and16_OperatorSTL/src/Class15and16_OperatorSTL.cpp

Operators – Increment/Decrement

```
int k = 5;
```

Increment and decrement operators.

Operator	Name	Example	
++	Auto Increment (prefix)	++k + 10	// gives 16
++	Auto Increment (postfix)	k++ + 10	// gives 15
--	Auto Decrement (prefix)	--k + 10	// gives 14
--	Auto Decrement (postfix)	k-- + 10	// gives 15

Notes

- Applicable to integers and real values
- Note difference between prefix and posfix

Credits: C++ Essentials by Sharam Hekmat, PragSoft

Operators – Assignment

Operator	Example	Equivalent To
=	n = 25	
+=	n += 25	n = n + 25
-=	n -= 25	n = n - 25
*=	n *= 25	n = n * 25
/=	n /= 25	n = n / 25
%=	n %= 25	n = n % 25
&=	n &= 0xF2F2	n = n & 0xF2F2
|=	n |= 0xF2F2	n = n | 0xF2F2
^=	n ^= 0xF2F2	n = n ^ 0xF2F2
<<=	n <<= 4	n = n << 4
>>=	n >>= 4	n = n >> 4

Notes

- Improves programmer productivity
- Makes code less readable
- No impact to code performance

Credits: C++ Essentials by Sharam Hekmat, PragSoft

Operators – Conditional Expressions

operand1 ? operand2 : operand3

Example

```
int m = 1, n = 2;
int min = (m < n ? m : n);
```

Example – Same As

```
int m = 1, n = 2;
int min;
if  (m < n)
    min = m;
  else
    min = n
```

Credits: C++ Essentials by Sharam Hekmat, PragSoft

Other Operators

- Comma (,): evaluate expressions from left and then right side of comma
- sizeof(): calculating the size of any data item or type in bytes
- new(): allocate memory
- delete(): free memory

Credits: C++ Essentials by Sharam Hekmat, PragSoft

Operator Precedence

Operator precedence levels.

Level	Operator						Kind	Order
Highest	: :						Unary	Both
	( )	[ ]	->	.			Binary	Left to Right
	+ -	++ --	! ~	* &	new delete	sizeof ( )	Unary	Right to Left
	->*	.*					Binary	Left to Right
	*	/	%				Binary	Left to Right
	+	-					Binary	Left to Right
	<<	>>					Binary	Left to Right
	<	<=	>	>=			Binary	Left to Right
	==	!=					Binary	Left to Right
	&						Binary	Left to Right
	^						Binary	Left to Right
	|						Binary	Left to Right
	&&						Binary	Left to Right
	||						Binary	Left to Right
	? :						Ternary	Left to Right
	=	+= -=	*= /=	^= %=	&= |=	<<= >>=	Binary	Right to Left
Lowest	,						Binary	Left to Right

Expression

a == b + c * d

Same as

a == (b + (c * d))

Credits: C++ Essentials by Sharam Hekmat, PragSoft

Exercise: Creating Own Operator

```
// An object of this type represents a linear function of one variable a * x + b.


struct Linear
{
        double a, b;
        double operator()(double x) const {
                return a * x + b;
        }
 };
```

Credit: https://en.cppreference.com/w/cpp/language/operators

What More ?

- Recall <<
  - Example: cout << “Hello World! << endl;
- Operator overloading
  - To be covered in future lecture

Discussion: Course Project

PA #3 Check: Due Thursday, March 2, 2023

Course Project – Building and Assembling of Prog. Assignments in Health

- **Project**: Develop collaborative assistants (chatbots) that offer useful information about diseases
- Specifically, use the CDC dataset on diseases at: https://wwwnc.cdc.gov/travel/diseases
  - For polio, it is: https://wwwnc.cdc.gov/travel/diseases/poliomyelitis
  - Each student will choose two diseases (from 47 available).
  - Each student will also use data about the disease from WebMD. Example for polio - https://www.webmd.com/children/what-is-polio
  - Programming assignment programs will: (1) extract data about a disease from two sites, (2) process it, (3) make content available in a command-line interface, (4) handle any user query and (5) report on interaction statistics.

Core Programs Needed for Project

- Prog 1: extract data from the district **[prog1-extractor]**
- Prog 2: process it (extracted data) based on questions **[prog2processor]**
- **Prog 3: make content available in a command-line interface [prog3-ui]**
- Prog 4: handle any user query and
- Prog 5: report statistics on interaction of a session, across session

Programming Assignment # 3

- **Goal**: **make content available in a command-line interface** [Name: **prog3-ui**]

- Program should do the following:
  - Run in an infinite loop until the user wants to quit
  - Handle any user response
    - User can quit by typing "Quit" or "quit" or just "q"
    - User can enter any other text and the program has to handle it. The program should write back what the user entered and say – "I do not know this information".
  - Handle known user query types
    - "Tell me about the disease", "What is ***malaria***?" => (Type-I1)
    - "What can I do after travel?" => (Type-I4)
    - "what is the treatment? " => (Type-I10)
    - "Tell me about ***malaria*** vaccine" => (Type-12)
    - …
    - "**Tell me everything**" => ***Give all information extracted (I1-I12)***

S1: https://www.cdc.gov/travel/diseases/malaria
- What is malaria? [I1]
- Who is at risk? [I2]
- What can travelers do to prevent malaria? [I3]
- After Travel [I4]
- More Information [I5]

S2: https://www.webmd.com/a-to-z-guides/malaria-symptoms
- What Is Malaria? [I1]
- Malaria Causes and Risk Factors [I2]
- Types of Malaria [I6]
- Symptoms [I7]
- When to Call a Doctor About Malaria [I8]
- Malaria Diagnosis [I9]
- Malaria Treatment [I10]
- Malaria Complications [I11]
- Malaria Vaccine [I12]

Notes on PA#3

- Handle all 12 information types
  - Multiple ways to ask for same information type
  - Variant assumes disease name from context or is specified
- Handle special query: ***Tell me everything***
- Handle others
  - Chit-chat
  - Give controlled response under all condition

Programming Assignment # 3

- Code organization
  - Create a folder in your GitHub called “prog3-ui”
  - Have sub-folders: src (or code), data, doc, test
  - Write a 1-page report in ./doc sub-folder
  - Send a confirmation that code is done by updating Google sheet; optionally, send email to instructor and TA

- Use concepts learned in class
  - Classes
  - Exceptions
  - UML Diagrams

Concluding Section

Lecture 15: Concluding Comments

- Reviewed HW#4
- We looked at the concept of operators
  - Many types
  - Precedence order when evaluating
- Reminder: Programming Assignment #3 due Thursday, March 2, 2023

About Next Lecture – Lecture 16

Lecture 16: C++ Standard Libraries

- C++ standard library

13	Feb 21 (Tu)	Exceptions	Prog 2 - end
14	Feb 23 (Th)	OO – Constructor, Destructor	Prog 3 - start
15	Feb 28 (Tu)	OO – operators, access control	HW 4 due
16	Mar 2 (Th)	C++ standard library	Prog 3 - end Semester - Midpoint
	Mar 7 (Tu)		Spring break – No class