AI INSTITUTE #AIISC
UNIVERSITY OF SOUTH CAROLINA

UNIVERSITY OF SOUTH CAROLINA

CSCE 240: Advanced Programming Techniques
Lecture 3: Input and Output

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

17TH JANUARY 2023

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 3

- Introduction Section
  - Recap of Lecture 2
  - Logistics: Slack, Github, Spreadsheet
- Main Section
  - Review additional tasks: Sorting of numbers
  - Concept: Data types
  - Concept: Strings
  - Concept: Streams, Input and Output
  - Concept: Error Handling
  - Numeric processing: Calculator
  - Team activity: code review, testing
- Concluding Section
  - About next lecture – Lecture 4
  - Ask me anything

Introduction Section

Recap of Lecture 2

- We discussed
  - Computer architecture
  - Concepts of pointers and iteration
- Looked at sorting of numbers in different languages

Logistics: Slack, Github, Spreadsheet

- Slack – please sign-in
- Github – only 5 students have filled
  - Create and share your private repo on “Students and Code Links” spreadsheet
  - Name the repo: CSCE240H-Spring2023-<your-first-name>. Example: CSCE240H-Spring2023-Bob
  - Additionally
    - Put a readme file (Readme.md) and give you full name
    - Create a sub-folder called “home-works”. Future home works will be here
    - Create a sub-folder called “project”. Your project will reside here.
- Spreadsheet for tracking student information (‘StudentEditable - Students and Code Links’)

Main Section

Review: Sorting of Numbers

- C++, Java, Python
- Using peer-reviewing

Peer Review: Homework Assignment #1

1. Go to spread sheet and on ”Homework Assignments - Peer Review” tab
2. Go to the row with your name
3. Peer review (10 mins)
    1. Enter roll 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 roll 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

Concept: Data Types

Common C++ types

Type	Typical Bit Width	Typical Range
char	1byte	-127 to 127 or 0 to 255
unsigned char	1byte	0 to 255
signed char	1byte	-127 to 127
int	4bytes	-2147483648 to 2147483647
unsigned int	4bytes	0 to 4294967295
signed int	4bytes	-2147483648 to 2147483647
short int	2bytes	-32768 to 32767
unsigned short int	2bytes	0 to 65,535
signed short int	2bytes	-32768 to 32767
long int	8bytes	-2,147,483,648 to 2,147,483,647
signed long int	8bytes	same as long int
unsigned long int	8bytes	0 to 4,294,967,295
long long int	8bytes	-(2^63) to (2^63)-1
unsigned long long int	8bytes	0 to 18,446,744,073,709,551,615
float	4bytes	
double	8bytes	
long double	12bytes	
wchar_t	2 or 4 bytes	1 wide character

Credit and Reference: https://www.tutorialspoint.com/cplusplus/cpp_data_types.htm

Concept: Strings in Languages

- C: unsigned array of characters
  - Methods to copy, find length,…
  - Array operators
- C++: standard class
  - Methods to find length, compare
- Java: built-in data type
- Python: built-in data type; sequence of Unicode

Reference:

- Python: https://docs.python.org/3/library/stdtypes.html#textseq, https://docs.python.org/3/library/string.html

Concept: Streams, Input, Output

- **Streams**: an abstraction of end-point (file, terminal, string, web resource - source, destination) with characters in sequential order of any length.
- Input
  - cin: command line for input
  - ifstream: file for input
- Output
  - cout: command line for output
  - ofstream: file for output
- Both
  - fstream, sstream

- Pattern for using streams
  1. Open a stream
  2. Do operation (read, write or both)
  3. Close the stream

Reference:

- C++ IO classes - https://www.cplusplus.com/reference/iolibrary/, https://www.cprogramming.com/tutorial/c++-iostreams.html

Concept: Streams, Input, Output

- Pattern for using streams
  1. Open a stream
  2. Do operation (read, write or both)
  3. Close the stream

open (filename, mode);

`ios::in`	Open for input operations.
`ios::out`	Open for output operations.
`ios::binary`	Open in binary mode.
`ios::ate`	Set the initial position at the end of the file. If this flag is not set, the initial position is the beginning of the file.
`ios::app`	All output operations are performed at the end of the file, appending the content to the current content of the file.
`ios::trunc`	If the file is opened for output operations and it already existed, its previous content is deleted and replaced by the new one.

Reference:

- C++ IO classes - https://www.cplusplus.com/reference/iolibrary/, https://www.cprogramming.com/tutorial/c++-iostreams.html
- Table courtesy - https://www.cplusplus.com/doc/tutorial/files/

Concept: Error Handling

- Error is unavoidable. Think ahead.
- Good programs have stable behavior; they handle known and unknown situations
- Error can be tested on streams anytime.
  - bad(): Returns true if a reading or writing operation fails.
  - fail(): Returns true in the same cases as bad(), but also in the cases of a format.
  - eof(): Returns true if a file open for reading has reached the end.
  - good(): It is the most generic state flag returning false if any of the previous functions would return true.

Reference:

- Table courtesy - https://www.cplusplus.com/doc/tutorial/files/

Illustrate Concepts

- C++
  - Class3and4_C++_IO.cpp
  - Notice: code and data are in separate folders

Illustration: Java

- Notice the three libraries for input and output, respectively
- Notice try/ catch to handle errors

Illustration: Python

- File: L3_DemoReadWrite.py
- Main code is 3 lines: open, read/ write, close
- No data typing

Review C++ With Peers – In Class

- Code walk through
- Unit testing

Programming Home Work (#1) – C++

- Write a program called FileBasedCalculator.
  - It reads three lines from an input file (called input.txt): the operation to be done (add, subtract, multiply or divide), and two integer numbers.
  - It writes two lines to an output file (called output.txt). The first line says – ”The result of <operation> on <num1> and <num2> is below”. The second line has the result.
- Modify the program so that it can handle missing input file name.

Discussion: Course Project

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.

Discussion: Nature and Simplifications

- Once you select a disease, the content is also fixed.
  - Enter selection in column F of spreadsheet
- Some simplifications
  - **Download local copy** v/s web query
  - **Read static content first**
  - **Handle a subset of content**
  - **Have default handling for questions** the chatbot does not understand
- Do project in a language you are most comfortable with
- Use all advanced programming concepts to simplify coding

Discussion: Chatbot Loop

- Input: from user (called utterance)
  - Problem specific query (i.e., about disease cauce)
  - Chitchat
  - Unrelated
- Output: from system (response)
  - Handle unrelated
  - Handle chitchat
  - Answer to query
- **Do it until user say over!**

Concluding Section

Lecture 3: Concluding Comments

- We discussed the concepts of data types, strings
- We discussed the concepts of streams and error handling
- We looked at programs in C++, Java and python on file handling
- Homework assignment - FileBasedCalculator
- Discussed projects

About Next Lecture – Lecture 4

Lecture 4: I/O

- Will review Home Assignment-1 (FileBasedCalculator) in class
- Handling mixed data types
- Error handling
- Printing values with formatting