

CSCE 240: Advanced Programming Techniques

Lecture 25: Review for Quiz2

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

11TH APRIL 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 25

- Introduction Section
 - Recap of Lecture 24
 - News / announcements / clarifications
- Main Section
 - Task: HW 6 – review
 - Project – Guidelines on submission and presentation
 - Review of concepts
- Concluding Section
 - About next lecture – Lecture 26
 - Ask me anything

Introduction Section

Recap of Lecture 24

- Project assignments
 - Programming practices by students in PA#4
 - Reviewed PA #5
- We discussed
 - AI / ML/ DL

Announcement

- McNair Junior Fellows program: **30 grantees** this summer, and we sure hope you can encourage your students to explore this opportunity. All details and applications are on: <http://www.cec.sc.edu/mjf> | **Deadline April 21st, 2023 !**
 - The program, in its 9th year since its foundation, and in its 5th year as an official CEC program, provides supports for undergraduate students up to 3k\$ in summer funds and runs activities that helps the students further explore research (as well as research posters, state of the art and other research initiation programs).
Contact: Ramy Harik
- Summer Internships
 - You can apply to fellowship and work with faculty ON YOUR IDEA
 - You can work with faculty ON THEIR IDEA and get paid
 - You can work on your idea with a faculty to mentor (with/ without fellowship)

Announcement

1. Workshop on Data-Driven Approaches to Transportation: Bridging Research and Practice
(hence, traffic management and AI)
Feb 28, 2023 - <https://sites.google.com/view/ccri-transportation-workshop/>

Main Section

Home Work 6 (Peer Review)

Due Tuesday, April 9, 2024

Home Work (#6) – C++ - Understand Code Optimization

- Consider BubbleSort algorithm for sorting (shown on right)
- Processing
 1. Generate n random numbers – called S
 2. (Store and Sort as Array)
 1. Store the numbers of S in an array: allocated with size n at the start
 2. **Sort using Bubble Sort**
 3. (Store and Sort as Vector)
 1. Store the numbers of S in a vector: allocated with size 1 and size increased one by one number until n
 2. Sort using Bubble Sort
 4. Measure time difference in both cases with n = 100, 1,000, 10,000, 100,000
- Output
 1. Make a graph showing any difference in timing
 2. Check the sorted results and confirm they are same (write a validator code to check)
 3. Document which one is your validation code

Algo 3 (BubbleSort):

- `current_array = a = Input`
- `For (i=0; i<N-1; i++) {`
 - `For (j=(i+1); j<N; j++) {`
 - `If(a[i] > a[j])`
 - `Swap(a[i], a[j])`
 - `}`
- `}`
- **Return current_array**

Example invocation

```
> SortN Bubble 1000  
Time for array: 1 sec  
Time for vector: 2 sec  
Result_checked: True
```

Home Work (#6) – C++ - Understand Code Optimization

- Consider BubbleSort algorithm for sorting (shown on right)
- Processing
 1. Generate n random numbers – called S
 2. (Store and Sort as Array)
 1. Store the numbers of S in an array: allocated with size n at the start
 2. **Sort using C++ Standard library's sort**
 3. (Store and Sort as Vector)
 1. Store the numbers of S in a vector: allocated with size 1 and size increased one by one number until n
 2. Sort using Bubble Sort
 4. Measure time difference in both cases with n = 100, 1,000, 10,000, 100,000
- Output
 1. Make a graph showing any difference in timing
 2. Check the sorted results and confirm they are same (write a validator code to check)
 3. Document which one is your validation code

See:

<https://en.cppreference.com/w/cpp/algorithm/sort>

Example invocation

```
> SortN Standard 1000  
Time for array: 1 sec  
Time for vector: 2 sec  
Result_checked: True
```

Summary

- Program name: **SortN**
 - Parameters: Bubble, Standard
- Test with: 100, 1,000, 10,000, 100,000
- Memory storage: Vector, Array
- Write a validator code to check result. Document which one is your validation code
- Draw a graph of results.

Peer Review: Homework Assignment #6

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

Presentation for Code Reuse

Context -- PA6: Assembling the Final Chatbot *With Selective Code Reuse*

- One can possibly make more marks when doing final project assembly
 - **Remember:** PA1, PA2, PA3, PA4, PA5 will be the 5 programs from assignments. [100 points for each assignment]
 - **Remember:** Assembling code from one's on assignments gets the standard [100 points].
 - Extra points will be given if you make your code (for PA1 – PA5) available to others (make repository public) AND someone uses your code (any of PA1-PA5). Both will have to be reported in their individual project reports.
 - 40 points will be given per assignment to student whose assignment is reused, and
 - 20 points will be given to person who reuses code
 - Extra points will not exceed 100 points for any student. That is, one cannot make more than 700 points.

<Student Name> - Project Summary

	Criteria	Information	What Needs to Change to Support Additional Company (Reuse Situation)
1	Companies and 10-k parts supported		
2	How data is extracted and stored		
3	How extracted data is processed, presented		
4	How intent is detected for user utterances		
5	How answer is given/ shown		
6	How statistics of session is calculated		
7	Performance of your chatbot	Quantitative, Qualitative	
8	Anything unique?		

Sample Output

Due One-Day Before Next Class

<Student Name> - Project Summary

Class Diagrams

Due Next Class

Discussion: Course Project

Course Project – Knowing About Companies

- **Project:** Develop collaborative assistants (chatbots) that offer useful information about companies
- Specifically, use the EDGAR dataset on companies at:
<https://www.sec.gov/edgar/searchedgar/companysearch>.
 - For Apple, it is: <https://www.sec.gov/edgar/browse/?CIK=320193&owner=exclude>
- **Each student will choose two companies (from thousand available).**
- Programming assignment programs will: (1) extract data about two companies from 10-k, (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 [\[prog4-userintent2querymapper\]](#)
- Prog 5: report statistics on interaction of a session, across sessions [\[prog5-sessionlogger\]](#)

Prog 6: Assembling the Chatbot

- Have a program - [\[my-chatbot\]](#)
- User interacts with the chatbot with any utterance and the system has to answer – see right
- User can ask about statistics and query log
 - Same as PA5
 - See next slide

[#1] “Quit” or “quit” or just “q” => Program exits

[#2a] “Tell me about the CEO of IBM”, “Tell me about the the risk factors”

[#2l] “Tell me everything” => Give all information
Extracted (Type-l12)

[#3] “What companies do you support for Q/A” => Give list of companies supported

[#4] “Give me your usage stats” => Give chat summary info

[#5] <User can enter any other text and the program has to handle it> => “I do not know this information” or
“Here is my guess - ” + <query> + <answer>. “Did I answer correctly ? “

All Queries to be Supported

[#1] "Quit" or "quit" or just "q" => Program exits

[#2a] "Tell me about the CEO of IBM", "Tell me about the the risk factors"

[#2] "Tell me everything" => Give all information
Extracted (Type-I12)

[#3] "What companies do you support for Q/A" => Give list of companies supported

[#4] "Give me your usage stats" => Give chat summary info

[#5] <User can enter any other text and the program has to handle it> => "I do not know this information" or
"Here is my guess - " + <query> + <answer>. "Did I answer correctly ? "

Chatbot usable in debug mode

my-chatbot -summary

=> There are 12 chats to date with user asking 23 times and system respond 24 times. Total duration is 456 seconds.

- **my-chatbot** -showchat-summary 2=> Chat 2 has user asking 2 times and system respond 2 times. Total duration is 4 seconds.

- **my-chatbot** -showchat 2

=> Chat 2 chat is:

...

- **my-chatbot** -showchat 200

=> ERROR: there are only 12 chat sessions. Please choose a valid number.

PA6: Assembling the Final Chatbot *With Selective Code Reuse*

- One can possibly make more marks when doing final project assembly
 - **Remember:** PA1, PA2, PA3, PA4, PA5 will be the 5 programs from assignments. [100 points for each assignment]
 - **Remember:** Assembling code from one's on assignments gets the standard [100 points].
 - Extra points will be given if you make your code (for PA1 – PA5) available to others (make repository public) AND someone uses your code (any of PA1-PA5). Both will have to be reported in their individual project reports.
 - 40 points will be given per assignment to student whose assignment is reused, and
 - 20 points will be given to person who reuses code
 - Extra points will not exceed 100 points for any student. That is, one cannot make more than 700 points.

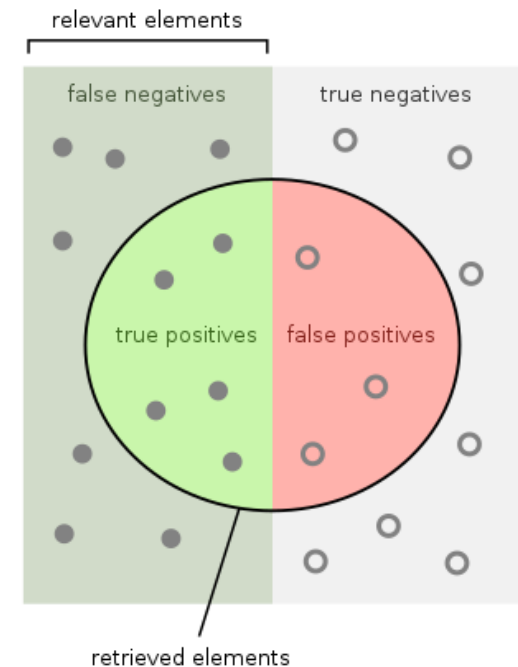
Project – PA#6

- Code organization
 - Create a folder in your GitHub called “**my-chatbot**”
 - Have sub-folders: src (or code), data, doc, test
 - Have data directory as shown in previous slide
 - `./data/chat_sessions/`
 - `./data/chat_statistics.csv`
 - Write a
 - Report in `./doc` sub-folder. Credit reuse
 - Create a presentation in `./doc` sub-folder
 - Put a log of system interacting in `./test`
 - Send a confirmation that code is done by updating Google sheet; optionally, send email to instructor and TA
- Use concepts learned in class
 - Exceptions
 - File operations
 - PA1 to PA5 from yourself or others; credit reuse in Readme, report and presentation

Evaluating a Chatbot's Performance Quantitatively (Information Retrieval)

- Precision:
(# Correct answers / # answers retrieved) * 100
// How well correctly answered
- Recall:
(# Correct answers / # answers stored) * 100
// How well correctly found
- Combined = F1 score =

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$



How many retrieved items are relevant?

Precision = $\frac{\text{true positives}}{\text{true positives} + \text{false positives}}$

How many relevant items are retrieved?

Recall = $\frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$

Credit: https://en.wikipedia.org/wiki/Precision_and_recall

Format for Project Report –

- Requirement – What did the instructor ask you to do?
- Specification – What did you do, what scope you selected and what decisions you made?
- Development highlights – How was your code implemented, e.g., module design, classes ? How did you test ? What problems did you face and how did you solve them?
- Reuse – What did you do to make your code reusable? Whose code did you use and why? Who is using your code and why ? What challenges did you face?
- Future work - What more can be done to make your chatbot useful? How will the code need to be changed over time?

Assignments: Late Submission Policy and Extra Marks

- There is no provision for late submission for programming assignments
 - Except when prior approval has been taken from instructor due to health reasons
- One can possibly make more marks when doing final project assembly
 - **Remember:** PA1, PA2, PA3, PA4, PA5 will be the 5 programs from assignments. [100 points for each assignment]
 - **Remember:** Assembling code from one's on assignments gets the standard [100 points].
 - Extra points will be given if you make your code (for PA1 – PA5) available to others (make repository public) AND someone uses your code (any of PA1-PA5). Both will have to be reported in project report.
 - 40 points will be given per assignment to student whose assignment is reused, and
 - 20 points will be given to person who reuses code
 - Extra points will not exceed 100 points for any student. That is, one cannot make more than 700 points.

Project Presenter Name:
Student Name:

Scope: Companies, Prog. Language

Data: What data is available and what
is retrieved from program ?

Code Organization: Anything significant
to highlight ?

PA1:

PA2:

...

PA6: code reuse by someone, and of
someone

Queries Snapshot

Video link:

Experience implementing the chatbot,
Testing

Experience with reuse

Review of Main Concepts

Assignments: Late Submission Policy and Extra Marks

- There is **no provision for late submission** for programming assignments
 - Except when prior approval has been taken from instructor due to health reasons
- One can possibly make more marks when doing final project assembly
 - **Remember:** PA1, PA2, PA3, PA4, PA5 will be the 5 programs from assignments. [100 points for each assignment]
 - **Remember:** Assembling code from one's on assignments gets the standard [100 points].
 - Extra points will be given if you make your code (for PA1 – PA5) available to others (make repository public) AND someone uses your code (any of PA1-PA5). Both will have to be reported in project report.
 - **40 points will be given per assignment to student whose assignment is reused**, and
 - **20 points will be given to person who reuses code**
 - Extra points will not exceed 100 points for any student. That is, one cannot make more than 700 points.

Review of Topics

Class #	Date	Description	Comments
1	Jan 10 (Tu)	Introduction	
2	Jan 12 (Th)	Introduction – Pointers, Iteration	
3	Jan 17 (Tu)	Input/ Output	
4	Jan 19 (Th)	I/O, Exceptions	HW 1 due
5	Jan 24 (Tu)	Memory management, User defined types	Prog 1 - start
6	Jan 26 (Th)	Object Oriented (OO) intro	HW 2 due
7	Jan 31 (Tu)	OO concepts, UML Notations	
8	Feb 2 (Th)	Code org (C++)	Prog 1 - end
9	Feb 7 (Tu)	OO – inheritance	Prog 2 - start
10	Feb 9 (Th)	Regex, OO - polymorphism	HW 3 due
11	Feb 14 (Tu)	In class test	Quiz 1 – In class
12	Feb 16 (Th)	Review: inheritance, Polymorphism	
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

17	Mar 14 (Tu)	Testing strategies	Prog 4 - start
18	Mar 16 (Th)	Advanced: Pointers	HW 5 due
19	Mar 21 (Tu)	Advanced: Pointers, I/O	
20	Mar 23 (Th)	Advanced: Operator overloading	Prog 4 – end (March 26, 2023)
21	Mar 28 (Tu)	Advanced: Memory Management	Prog 5 – start
22	Mar 30 (Th)	Advanced: Code efficiency	
23	Apr 4 (Tu)	Advanced: Templates	
24	Apr 6 (Th)	AI / ML and Programming	Prog 5 – end
25	Apr 11 (Tu)	Project code summary – student presentation for reuse Review material for Quiz 2	HW 6 due Prog 6 – assembling start
26	Apr 13 (Th)	In class test	Quiz 2 – In class
27	Apr 18 (Tu)	Project presentation	Prog 6 - due
28	Apr 20 (Th)	Project presentation	Last day of class
	Apr 25 (Tu)		Reading Day
29	May 2 (Tu)	9am – Exam or Final Overview	Examination

Concluding Section

Lecture 25: Concluding Comments

- Quiz 6 peer evaluation
- Project - the breakup of marks (100) will be as follows –
 - 20 points for the fully working demo. Submit code and video.
 - 40 points for report. Submit report in format.
 - 40 points for the presentation. Have slides ready.
 - There will be no further submissions. **All are due by Monday, April 17, 2023.**
- Review for Quiz 2

About Next Lecture – Lecture 26

Lecture 26: Quiz 2

- Quiz 2 – depends on reuse for PA 6
 - All concepts taught in class
 - No online giving option

19	Mar 19 (Tu)	Advanced: Pointers, I/O	
20	Mar 21 (Th)	Advanced: Operator overloading	Prog 4 – end
21	Mar 26 (Tu)	Advanced: Memory Management	Prog 5 – start HW 5 due
22	Mar 28 (Th)	Advanced: Code efficiency	
23	Apr 2 (Tu)	Advanced: Templates	
24	Apr 4 (Th)	AI / ML and Programming	Prog 5 – end
25	Apr 9 (Tu)	Project code summary – student presentation for reuse Review material for Quiz 2	HW 6 due Prog 6 – assembling start
26	Apr 11 (Th)	In class test	Quiz 2 – In class
27	Apr 16 (Tu)	Project presentation	Prog 6 - due
28	Apr 18 (Th)	Project presentation	Last day of class (April 22 per bulletin)
	Apr 23 (Tu)		Reading Day
29	Apr 25 (Tu)	9am – Final Overview	Examination