CSCE 240: Advanced Programming Techniques
Lecture 16: C++ Standard Library, PA 3 (Due)

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

3RD MARCH 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 16

- Introduction Section
  - Recap of Lecture 15
  - TA and SI Updates
- Main Section
  - Concept: Standard Library
  - Discussion: Project
- Concluding Section
  - About next lecture – Lecture 17
  - Ask me anything

Introduction Section

Recap of Lecture 15

- Reviewed HW#4
- We looked at the concept of operators
  - Many types
  - Precedence order when evaluating
- Programming Assignment #3 due today

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.

Updates from TA, SU

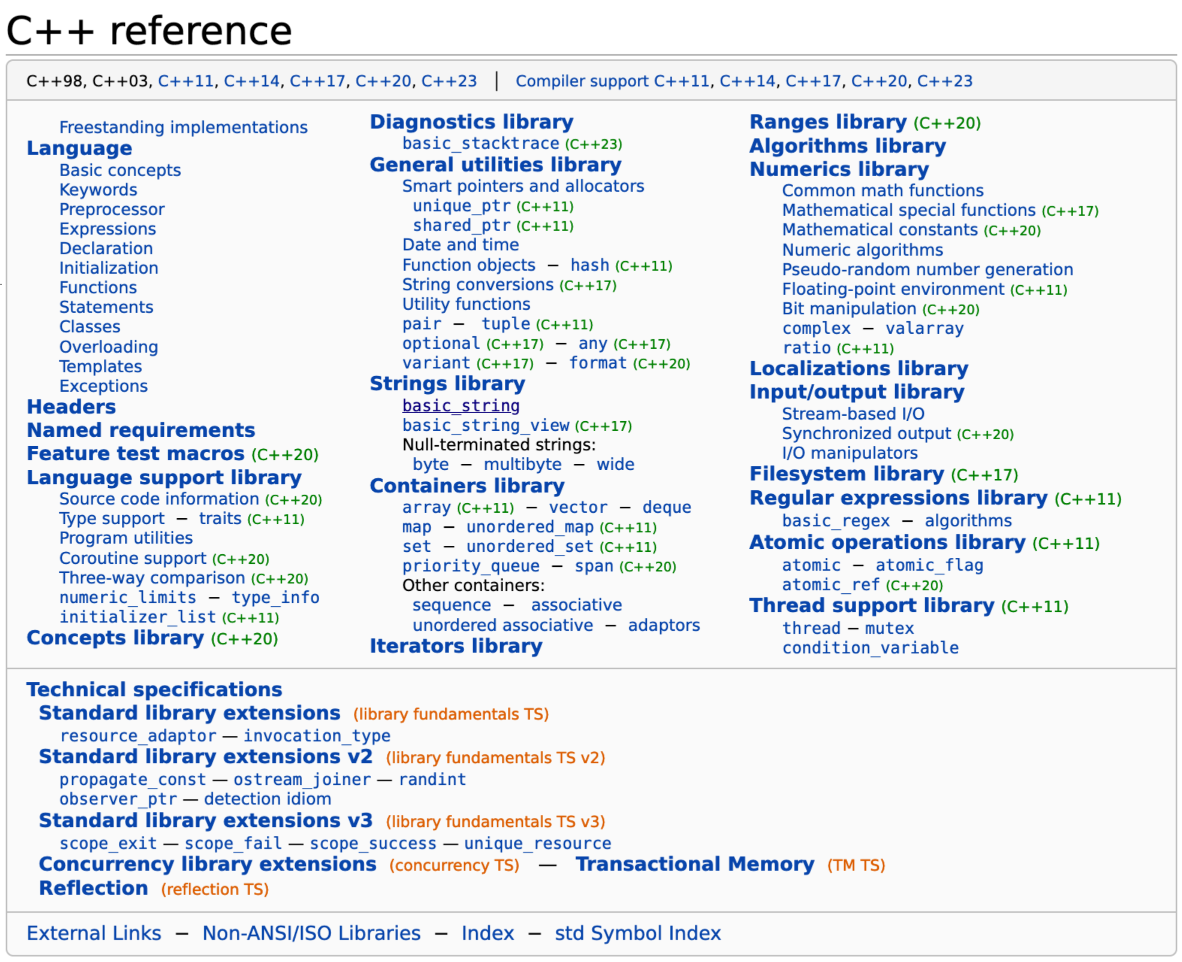
- TA update: Yuxiang Sun (Cherry)
  - HW4 marks now on Blackboard
  - Assignments and homeworks: confirm submission in spreadsheet with time completed.

- SI update: Blake Seekings

Main Section

Concept: C++ Standard Library

C++



C++ reference

C++98, C++03, C++11, C++14, C++17, C++20, C++23 | Compiler support C++11, C++14, C++17, C++20, C++23

Freestanding implementations

Language
- Basic concepts
- Keywords
- Preprocessor
- Expressions
- Declaration
- Initialization
- Functions
- Statements
- Classes
- Overloading
- Templates
- Exceptions

Headers

Named requirements

Feature test macros (C++20)

Language support library
- Source code information (C++20)
- Type support — traits (C++11)
- Program utilities
- Coroutine support (C++20)
- Three-way comparison (C++20)
- numeric_limits — type_info
- initializer_list (C++11)

Concepts library (C++20)

Diagnostics library
- basic_stacktrace (C++23)

General utilities library
- Smart pointers and allocators
  - unique_ptr (C++11)
  - shared_ptr (C++11)
- Date and time
- Function objects — hash (C++11)
- String conversions (C++17)
- Utility functions
- pair — tuple (C++11)
- optional (C++17) — any (C++17)
- variant (C++17) — format (C++20)

Strings library
- basic_string
- basic_string_view (C++17)
- Null-terminated strings:
  - byte — multibyte — wide

Containers library
- array (C++11) — vector — deque
- map — unordered_map (C++11)
- set — unordered_set (C++11)
- priority_queue — span (C++20)
- Other containers:
  - sequence — associative
  - unordered associative — adaptors

Iterators library

Ranges library (C++20)

Algorithms library

Numerics library
- Common math functions
- Mathematical special functions (C++17)
- Mathematical constants (C++20)
- Numeric algorithms
- Pseudo-random number generation
- Floating-point environment (C++11)
- Bit manipulation (C++20)
- complex — valarray
- ratio (C++11)

Localizations library

Input/output library
- Stream-based I/O
- Synchronized output (C++20)
- I/O manipulators

Filesystem library (C++17)

Regular expressions library (C++11)
- basic_regex — algorithms

Atomic operations library (C++11)
- atomic — atomic_flag
- atomic_ref (C++20)

Thread support library (C++11)
- thread — mutex
- condition_variable

Technical specifications

Standard library extensions (library fundamentals TS)
- resource_adaptor — invocation_type

Standard library extensions v2 (library fundamentals TS v2)
- propagate_const — ostream_joiner — randint
- observer_ptr — detection idiom

Standard library extensions v3 (library fundamentals TS v3)
- scope_exit — scope_fail — scope_success — unique_resource

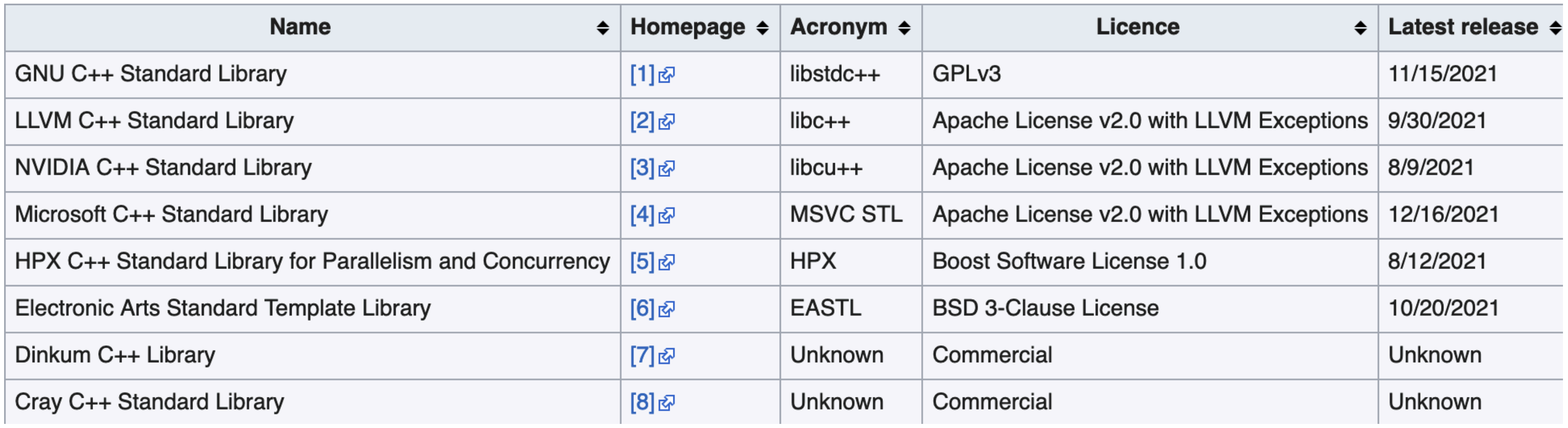
Concurrency library extensions (concurrency TS) — **Transactional Memory** (TM TS)

Reflection (reflection TS)

External Links — Non-ANSI/ISO Libraries — Index — std Symbol Index

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

Many Implementations



Name	Homepage	Acronym	Licence	Latest release
GNU C++ Standard Library	[1]	libstdc++	GPLv3	11/15/2021
LLVM C++ Standard Library	[2]	libc++	Apache License v2.0 with LLVM Exceptions	9/30/2021
NVIDIA C++ Standard Library	[3]	libcu++	Apache License v2.0 with LLVM Exceptions	8/9/2021
Microsoft C++ Standard Library	[4]	MSVC STL	Apache License v2.0 with LLVM Exceptions	12/16/2021
HPX C++ Standard Library for Parallelism and Concurrency	[5]	HPX	Boost Software License 1.0	8/12/2021
Electronic Arts Standard Template Library	[6]	EASTL	BSD 3-Clause License	10/20/2021
Dinkum C++ Library	[7]	Unknown	Commercial	Unknown
Cray C++ Standard Library	[8]	Unknown	Commercial	Unknown

Credit: https://en.wikipedia.org/wiki/C%2B%2B_Standard_Library

Why Use Standard Library and Why Not ?

- Note: One can always implement a functionality themselves
- Reasons to reuse
  - Lesser development effort. Someone has created it.
  - Task needs specialized knowledge that the developer does not have
  - Usually, well tested.
  - Usually, efficient.
  - Well-documented. So, code using them easier to maintain
- Reasons not to reuse
  - Want to be in control of behavior and performance
  - Want to control code size/ memory footprint
  - Task needs specialized knowledge that the developer has

Credit: Adapted from 'Fundamentals of C++ Programming', Richard Halterman

Commonly Used: String

- Purpose: Make working with strings easy
- Examples
  - **Position**: front, back
  - **Size related**: size, capacity
  - **Character manipulation:** replace
  - **Search**: find
  - **Type conversion**: stoi, stof

C++ Standard Library

- Input/output
- Strings
- algorithm
- functional

Containers

- Sequence containers
- Associative containers
- Unordered associative containers

C standard library

- Data types
- Character classification
- Strings
- Mathematics
- File input/output
- Date/time
- Localization
- Memory allocation
- Process control
- Signals
- Alternative tokens
- Miscellaneous headers:
  - <assert.h>
  - <errno.h>
  - <setjmp.h>
  - <stdarg.h>

Reference:
https://en.cppreference.com/w/cpp/string/basic_string

Credit: https://en.wikipedia.org/wiki/C%2B%2B_Standard_Library

Commonly Used: String

- Code illustration
  - Front
  - Back
  - Size
  - Capacity
  - substr

Commonly Used: Mathematical Functions

- Purpose: Make numerical computation easy

- Examples
  - **Basic**: abs, mod, nan (not a number), round, nearestint, infinity
  - **Exponential**: exp, log
  - **Power:** pow, sqrt, hypot (computes square root of the sum of the squares of two or three)
  - **Trigonometric**: sin, cos, tan, atan
  - **Floating point**: round, floor, ceil

Reference:
https://en.cppreference.com/w/cpp/numeric/math

Commonly Used: Mathematical Functions

- Code illustration
  - Sqrt
  - Cbrt
  - Round
  - Nearbyint
  - Infinity, nan

- Support for complex numbers - example
  - https://en.cppreference.com/w/cpp/numeric/complex

Sometimes Used: Algorithmic Functions

- Purpose: Make ready implementation of popular algos

- Examples
  - Sequence operations: count, find, search
  - Sorting: sort
  - Partitioning
  - Permutation
  - Set operations
  - Numeric

Sometimes Used: Algorithmic Functions

- Code illustration
  - Sort
  - permutation

Sometimes Used: Algorithmic Functions

- Purpose: Make implementation of useful containers easily available

- Examples
  - Array
  - Vector
  - Map (also called HashMap or dict in other languages)
  - Priority_queue

Discussion: Course Project

Course Project – Assembling of Prog. Assignments

- **Project**: Develop collaborative assistants (chatbots) that offer innovative and ethical solutions to real-world problems ! *(Based on competition - https://sites.google.com/view/casy-2-0-track1/contest)*
- Specifically, **the project will be building a chatbot that can answer questions about a South Carolina member of state legislature from**:

https://www.scstatehouse.gov/member.php?chamber=H

  - Each student will choose a district (from 122 available).
  - Programming assignment programs will: (1) extract data from the district, (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
    - “Tell me about the representative”, “Tell me about the rep” => Personal Information (Type-I2)
    - “Where does the rep live” => Contact Information (Type-I1): Home Address
    - “How do I contact my rep ” => Contact Information (Type-I1)
    - “What committees is my repo on” => Committee Assignments (Type-I3)
    - “Tell me everything” => *Give all information extracted*

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

Example: Representative Information

Input and Output Example

prog3ui

System: “Hi – Welcome”

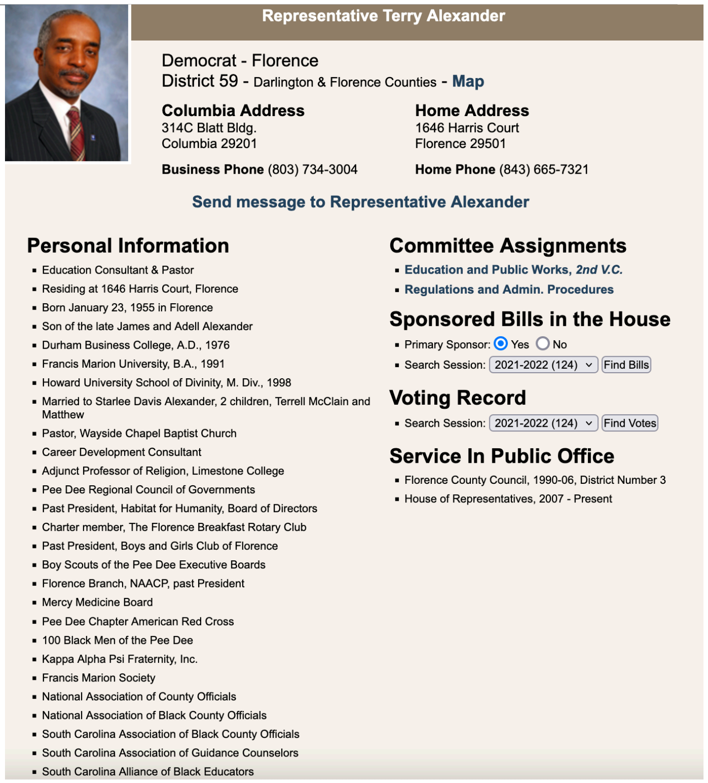
User: “Tell me about the rep”

System: …

…

User: q

- Contact Information (Type-I1)
- Personal Information (Type-I2)
- Committee Assignments (Type-I3)
- Sponsored Bills in the House (Type-I4)
- Voting Record (Type-I5)
- Service in Public Office (Type-I6)



Representative Terry Alexander

Democrat - Florence
District 59 - Darlington & Florence Counties - Map

Columbia Address
314C Blatt Bldg.
Columbia 29201

Home Address
1646 Harris Court
Florence 29501

Business Phone (803) 734-3004

Home Phone (843) 665-7321

Send message to Representative Alexander

Personal Information
- Education Consultant & Pastor
- Residing at 1646 Harris Court, Florence
- Born January 23, 1955 in Florence
- Son of the late James and Adell Alexander
- Durham Business College, A.D., 1976
- Francis Marion University, B.A., 1991
- Howard University School of Divinity, M. Div., 1998
- Married to Starlee Davis Alexander, 2 children, Terrell McClain and Matthew
- Pastor, Wayside Chapel Baptist Church
- Career Development Consultant
- Adjunct Professor of Religion, Limestone College
- Pee Dee Regional Council of Governments
- Past President, Habitat for Humanity, Board of Directors
- Charter member, The Florence Breakfast Rotary Club
- Past President, Boys and Girls Club of Florence
- Boy Scouts of the Pee Dee Executive Boards
- Florence Branch, NAACP, past President
- Mercy Medicine Board
- Pee Dee Chapter American Red Cross
- 100 Black Men of the Pee Dee
- Kappa Alpha Psi Fraternity, Inc.
- Francis Marion Society
- National Association of County Officials
- National Association of Black County Officials
- South Carolina Association of Black County Officials
- South Carolina Association of Guidance Counselors
- South Carolina Alliance of Black Educators

Committee Assignments
- Education and Public Works, *2nd V.C.*
- Regulations and Admin. Procedures

Sponsored Bills in the House
- Primary Sponsor: [✓ Yes] No
- Search Session: 2021-2022 (124) Find Bills

Voting Record
- Search Session: 2021-2022 (124) Find Votes

Service In Public Office
- Florence County Council, 1990-06, District Number 3
- House of Representatives, 2007 - Present

Announcements

- Chatbots – Event on March 18, 2022
  - Collaborative Assistants for Society (CASY) – in person and virtual event on campus
  - 9:30 am – 1:00 pm; talks and student use-cases
- Details and registration info: https://casy.aiisc.ai
- Looking for a panelist from class

Concluding Section

Lecture 16: Concluding Comments

- We looked at the c++ standard library
  - Many types of functionality
  - String, I/O, Mathematical libraries most commonly used
- Remember that many implementations of C++ standard library, usually based on different OS or hardware
  - Implements changing specs
- Be ready to implement one's own (rather than reuse), if necessary, for performance

About Next Lecture – Lecture 17

Lecture 17: C++ Standard Libraries

- No class next week
- Code testing strategies
- Start of PA #4
- Will give HW #5

Mar 3 (Th)	**C++ standard library**	**Prog 3 - end Semester - Midpoint**
Mar 8 (Tu)		Spring break – No class
Mar 10 (Th)		Spring break – No class
Mar 15 (Tu)	Testing strategies	Prog 4 - start
Mar 17 (Th)	Advanced: Pointers	HW 5 due