CS2124 Midterm Coverage - 25S

Basics

Lec 1-3; rec01,2; hw01,2

- Types
  - static typing
  - Common types: primitive types, strings and vectors
  - Default values for non-primitives
- Conditions / if / else
- Looping: for loop, while, do-while, ranged for. Also break and continue.
- Console I/O
- File input: open and closing files. Testing if open works. Reading files.
- Functions / parameter passing / return types. Default parameter values
  - Use of by-value, by-reference and by-constant-reference with ranged for
- Defining types: structs
  - Filling a vector with instances of a struct.

OOP Basics

Lec 4,5 / rec03 / hw03

- Encapsulation and data hiding
  - meaning of public / private.
- Constructors
  - Initialization lists
    - order of member variable initialization
  - Default constructor
  - Used with emplace_back
- Methods
  - const methods
  - getters / setters
- Nested classes
- Delegation
- Overloading output operator
- Difference between the keywords `struct` and `class`

Pointers and Dynamic Memory

Lec 6,7 / rec04,5 / hw04,5

- Association
- Addresses
- address-of operator: &
- Pointer variables and strict typing
- dereference operator: `*`
- `nullptr`
- `this`
- arrow operator: `->`
- operator precedence: dot vs. asterisk
- Issues: dangling pointers and memory leaks
- `new` / `delete`
- pointers and `const`

Copy Control and Vector implementation

Lec 8-11; rec05,6;

- **Copy control**:
  - destructor,
  - copy constructor,
  - assignment operator
- dynamic arrays: allocating and freeing up.
- pointer arithmetic: **`p[k] == *(p+k)`** for all integer values of k and pointers p.
- Index operator: **`operator[]`**
  - Overloading methods based on const
- keyword: **`explicit`** and implicit conversions
- ranged for support, i.e. **`begin()`** and **`end()`** methods.

Operator Overloading

Lec 5, 8-12; rec07;

- Implement as member or non-member
- Restrictions on operator overloading
  - Arity, associativity, precedence
  - Operators must already exist for primitives
  - Operators cannot be overloaded if only arguments are primitives
- Some of the operators that we have overloaded:
  - `<<,  >>, =, +=, ==, <, <=, >, >=. != +, ++, --, []`

Not covered

- Conversion operators, e.g. `operator bool`
- Cyclic Association
- Separate compilation