Chapter 10
Pointers

FORDHAM

CISC 5004 - Programming C++
Sam Kamens
Fall, 2022

Pointer

- A variable that contains the memory address of another variable

Vectors

- Implemented using a Dynamically Allocated Array
  - More powerful/flexible than regular arrays
  - Safer memory management

Vectors vs. Linked Lists

- Insert: Vectors
- Insert: Linked List
- Comparison of vectors and linked lists
- Standard collections may use either option internally
  - Be aware of the operations each one is optimized for

Call class member functions

- `this` pointer
  - Implicit argument in class functions
- Implemented as a pointer
  - Most often use -> to access
  - Example

Pointer Basics

- Pointer has a data type
  - Indicates what type of data is in the address that the pointer points to
  - Declaring and assigning pointers
- Dereferencing a pointer
  - dereference operator: `*`
- Null pointer
  - Indicates that a pointer points to nothing
  - `nullptr`
    - Historically `0` or `NULL`
  - Best practices
    - Always initialize a pointer to `nullptr` or assign it a correct value immediately
    - Alway check for null before trying to use a pointer
- Pointer example

Common pointer errors

- `*valPointer = &maxValue`
  - Syntax error
- `int* valPointer1, valPointer2;`
  - Only the first pointer will actually be a pointer type
  - I personally prefer to associate the * with the variable, not the type:
    - int *valPointer1
    - The textbook prefers the other way, and only declares one pointer per line
- Dereferencing an uninitialized or Null pointer

Pointer Operators: new

- The `new` operator
  - Allocates memory for a given type and returns a pointer to the allocated memory
  - If the type is a class, the constructor is executed
    - Arguments can be passed in to the constructur
    - Examples: 1, 2
- C equivalent: `malloc`
  - Almost never used in C++ coding

Member Access Operator

- Member access operator: `->`
  - “Syntactic Sugar”
    - `a->b` is equivalent to `(*a).b`
  - Examples

The delete operator

- `delete`
  - Deallocates (or frees) a block of memory
  - `delete pointerValue;`
- After the deletion, dereferencing the pointer leads to undefined behavior (often a program crash)
- Example
- C equivalent: `free`

Allocating deleting arrays of objects

- Syntax
  - `type* array = new type[size]`
  - `delete[] array`
    - Important! `delete array` leads to undefined behavior
- Examples

String functions with pointers

- Character pointers (`char *`) in C string library
  - `#include <cstring>`
- `strcmp()` and `strcpy()`
- String search functions
- String search and replace example

A first linked list

- Basic example
- Inserting
- Improved example
- Coding practice

Memory Regions

- Program's memory has four distinct regions
  - Code
    - Program instructions.
    - Generally speaking, this region is never changed
  - Static memory
    - Global or static variables
  - Stack
    - Local variables associated with a function call
    - "Stack frame"
  - Heap
    - Region where memory allocated by `new` is located
    - aka free store
- Illustration

Destructors

- Special class memory automatically called when a variable of that class is destroyed
  - Leaving scope
  - `delete`
- Uses
  - Deallocate memory allocated by the class
  - Release resources held
- Syntax
  - A function declaration with the class name (like a constructor) but prefixed with tilde: ~
  - Only one constructor can be defined for a class (unlike constructors)
- Linked List Destructor code
- Linked list destructor example

Memory Leak

- Occurs when a program accumulates memory that it doesn't release but also doesn't need anymore
  - Loss of access
  - Bad algorithm
  - Not freed in destructor
  - Builds up over time in long running programs (like your browser!)
- Illustration
- Garbage Collection
  - Some languages automatically free unused memory (e.g. Java)
  - C/C++ generally don't, for performance reasons

Copy Constructors

- The problem
- Automatically created constructor
  - Copies members variable values from old to new
  - “Shallow Copy”
  - Fine for simple members (int, etc.), but for pointers, this would mean that both classes point to the same memory location
- Solution: Programmer defined copy constructor
  - Syntax
  - “Deep Copy”
  - Example

Assignment Operator

- By default, assignment of objects works the same as the default copy constructor
- Assignment operator (=) can be **overloaded**
- Example

Rule of Three

- Classes typically have three types of functions implemented together (“the big three”)
  - Destructor
  - Copy constructor
  - Copy Assignment Operator
    - Note: It is possible to explicitly forbid copying or assignment
- General rule: if any of these are needed/used, define all three

Examples

- Managing an employee list using a vector
- Library Book Sorting
- Inventory
- Grocery List