AI INSTITUTE #AIISC
UNIVERSITY OF SOUTH CAROLINA

UNIVERSITY OF SOUTH CAROLINA

CSCE 240: Advanced Programming Techniques
Lecture 12: Constructors and Destructors

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

15TH FEBRUARY 2024

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 12

- Introduction Section
  - Recap of Lecture 11
- Main Section
  - Concept: Constructors
  - Concept: Destructors
  - Home work #4
  - Discussion: Project, Programming Assignment #3
- Concluding Section
  - About next lecture – Lecture 13
  - Ask me anything

Introduction Section

Recap of Lecture 11

- Looked at Errors
- Looked at Exception Handling
- Examples of Exceptions
  - In C++, Java, Python
  - Creating new exception handlers in C++

Main Section

Concept: Constructors

Constructor - What is It?

- Special function in every class
  - Always has the same name as the class itself
  - Does not have an explicit return type
  - Multiple constructors possible per class

- **Purpose**: Used to initialize objects of that class

Specification

```
class PersonName {
        string firstName;
        string lastName;

public:
        PersonName();
        PersonName(string);
        PersonName(string, string);
…
```

https://github.com/biplav-s/course-adv-proglang/blob/main/sample-code/CandC%2B%2B/Class7and8_C%2B%2B_OO/src/headers/PersonName.h

Observation: Constructor

- Declaration is usually public
- What happens if a constructor is **private** ?
  - Could declare, but no object can be declared
- A program cannot explicitly call constructors like other member functions
  - Implicitly called by instantiating a class

```
PersonName p1;
```

Constructor - What is It?

- Special function in every class
  - Always has the same name as the class itself
  - Does not have an explicit return type
    Multiple constructors possible per class

- **Purpose**: Used to initialize objects of that class

Usage

```
PersonName p1;
PersonName p2("Joginder");
PersonName p3("Joginder", "Singh"
```

Implementation

```
PersonName::PersonName() {
        firstName = "default-Maria";
        lastName = "default-Wang";
}

PersonName::PersonName(string first) {
        firstName = first;
        lastName = "default-Wang";
}

PersonName::PersonName(string first,
                       string last) {
        firstName = first;
        lastName = last;
}
```

https://github.com/biplav-s/course-adv-proglang/blob/main/sample-code/CandC%2B%2B/Class7and8_C%2B%2B_OO/src/implem/PersonName.cpp

Order of Calling Constructors in Hierarchy

- Parent then Child or Child before Parent ?

- **Parent first**

```
*** DEMO of Grand Child Class ***

Testing: data member –

  DEMO of Constructor – Parent Class ***

         DEMO of Constructor – Another Child Class ***

                  DEMO of Constructor – GrandChild Class ***
```

Discussion: Example - DayOfYear

```
class DayOfYear
{
public:
    DayOfYear(int monthValue, int dayValue);
                    //Constructor initializes month & day
    void input();
    void output();
    …
private:
    int month;
    int day;
}
```

```
// Example usage 1:
DayOfYear date1(7, 4), date2(5, 5);
// What happens?

// Example usage 1:
DayOfYear date1, date2
date1.DayOfYear(7, 4);
date2.DayOfYear(5, 5);
// What happens?
```

Example Credit: Absolute C++, Savitch

Concept: Destructors

Destructors - What is It?

- Special function in every class
  - Always has the same name as the class itself but prefixed with ~
  - Does not have an explicit return type
  - Does not take an argument
  - Maximum one destructor per class

- **Purpose**: Used to cleanup before removing objects of that class
  - Common usage: freeing memory allocated by the object's data members before the object is destroyed
  - Common usage: Close files, streams

Implementation

```
PersonName::~PersonName() {

}
```

https://github.com/biplav-s/course-adv-proglang/blob/main/sample-code/CandC%2B%2B/Class7and8_C%2B%2B_OO/src/implem/PersonName.cpp

Order of Calling Destructors in Hierarchy

- Parent then Child or Child before Parent ?

- **Child first !**

```
*** DEMO of Grand Child Class ***

…
The grandchild's location is: AnotherChild:default-location
                  Demo of Destructor - GrandChild Class ***
         DEMO of Destructor - Another Child Class ***
  DEMO of Destructor - GrandChild Class ***
```

Full Example

Program: Class9and10_C++_OOAdv.cpp **Argument**: 3

```
*** DEMO of Grand Child Class ***

Testing: data member -

  DEMO of Constructor - Parent Class ***

         DEMO of Constructor - Another Child Class ***

                  DEMO of Constructor - GrandChild Class ***

The grandchild's name is: Parent:default-name
The grandchild's location is: AnotherChild:default-location

                  Demo of Destructor - GrandChild Class ***

         DEMO of Destructor - Another Child Class ***

  DEMO of Destructor - GrandChild Class ***
```

Discussion: Using Constructors / Destructors Effectively

- Remember: Create automatically if none provided by developer
- Constructor: initialization of data members
- Destructor: clean-up
- Remember the order, use it productively but do not overly depend on it.

In-Class Exercise

Design of Constructors and Destructors

- Example setting: Calculator
  - Numbers and their operations
  - Operations: Addition, subtraction, division multiplication
  - Number types: natural numbers, whole numbers, rational numbers (fractions), irrational numbers, decimal numbers, binary, complex numbers, octal, hexadecimal, …
- Constructor considerations
- Destructor considerations

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
- **Prog 2: process it (extracted data) based on questions**
- Prog 3: make content available in a command-line interface
- Prog 4: handle any user query and
- Prog 5: report statistics on interaction of a session, across session

Content Reference: Queries for (Answers) Data We Have

- What does the (company) do? // Answers in Part 1
  - What is the (company's) business?
  - What are (company's) risk factors?
  - What does (company) own?
  - …
- Where does (company) operate? // Answers in Part 2
  - What has (company) disclosed?
- How is (company) structured? // Answers in Part 3
  - Who is (company's) CEO?
  - How much does (person) earn?
  - …
- What was in (company) statements? // Answers in Part 4
  - …

Concepts: 10-K, Parts, Items

Parts

- Part 1: Business Background and Risks
  - Item 1: Business
  - Item 2: Risk factors
  - Item 3: Properties
  - Item 4: Legal Proceedings
- Part 2: Operations and Disclosures
  - .. Market
  - .. Disclosures
- Part 3: Company Structure
  - Directors
  - Compensation
- Part 4: Financial Statements
  - Statements

Programming Assignment # 2

- Goal: **process extracted text based on questions**
  - Language of choice: Any from the three (C++, Java, Python)
- Program should do the following:
  - Take input from a local file which has content obtained from Prog#1 (when **company** name given as input)
  - Given an information type as input, the program will return its content
    - Examples: what is company's risk factors? What does company's CEO earn?
    - Input type can be given as command line argument.
      Examples:
      - prog2processor –t "what are **IBM's** risk factors?" // Tell about company
      - prog2processor –t "all information" // Get all info for a company
  - For demonstrating that your program works, have a file called "test_output.txt" showing the set of supported commandline options and output in the doc folder.

Concepts: 10-K, Parts, Items

Parts

- Part 1: Business Background and Risks
  - Item 1: Business
  - Item 2: Risk factors
  - Item 3: Properties
  - Item 4: Legal Proceedings
- Part 2: Operations and Disclosures
  - .. Market
  - .. Disclosures
- Part 3: Company Structure
  - Directors
  - Compensation
- Part 4: Financial Statements
  - Statements

- Code organization
  - Create a folder in your GitHub called "prog2-processor"
  - 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 to instructor and TA, and update Google sheet

Discussion

- Constructors to have
  - Benefits
- Destructors to have
  - Benefits

Discussion: Using Constructor in Project

- Company class
  - Initialization / customization of a company 10-K object
    - Website url for 10-k content
    - Initializing parsing rules
  - Allocating memory
  - Customizing content response
  - Reusing common services – logging, error handling
    - Initializing log file
    - Customizing error messages

Reminder: Student Assessment

A = [900-1000]
B+ = [850-899]
B = [800-849]
C+ = [750-799]
C = [700-749]
D+ = [650-699]
D = [600-649]
F = [0-599]

Tests	1000 points
• Course Project: programming assign.(5) and report, in-class presentation	600 points
• Class Participation and Home Work	200 points
• Quizzes and Exams	200 points
Total	1000 points

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.

Concluding Section

Lecture 12: Concluding Comments

- We looked at the concept constructor
- We looked at the concept of destructor

About Next Lecture – Lecture 13

Lecture 13: Review before Quiz

- Inheritance
- Polymorphism
- Errors/ Exception Handling
- Constructor/ Desctructor

Feb 6 (Tu)	OO – inheritance	Prog 2 - start
Feb 8 (Th)	Regex, OO - polymorphism	HW 3 due
Feb 13 (Tu)	Exceptions	
Feb 15 (Th)	OO – Constructor, Destructor	Prog 2 – end
Feb 20 (Tu)	Review: inheritance, Polymorphism	Quiz 1 – In class
Feb 22 (Th)	In class test	Prog 3 - start
Feb 27 (Tu)	In class Project Review: PA1 and PA2	
Feb 29 (Th)	OO – operators, access control	Prog 3 - end Semester - Midpoint