AI INSTITUTE #AIISC
UNIVERSITY OF SOUTH CAROLINA

UNIVERSITY OF SOUTH CAROLINA

CSCE 240: Advanced Programming Techniques

Lecture 5: Function Calls, User-defined Types, Memory Management, HW 2 (Given), Prog 1 (Start)

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

24TH JANUARY 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 5

- Introduction Section
  - Recap of Lecture 4
  - TA and SI Updates
- Main Section
  - Concept: Function calls
  - Concept: User defined types
  - Concept: Memory management
  - Home work2 – given
  - Programming assignment 1 - Start
- Concluding Section
  - About next lecture – Lecture 6
  - Ask me anything

Introduction Section

Recap of Lecture 4

- We experienced peer review on home work - FileBasedCalculator
- Discussed the concepts of mixed types
- Discussed formatted printing

Main Section

C/C++ Compilation Process

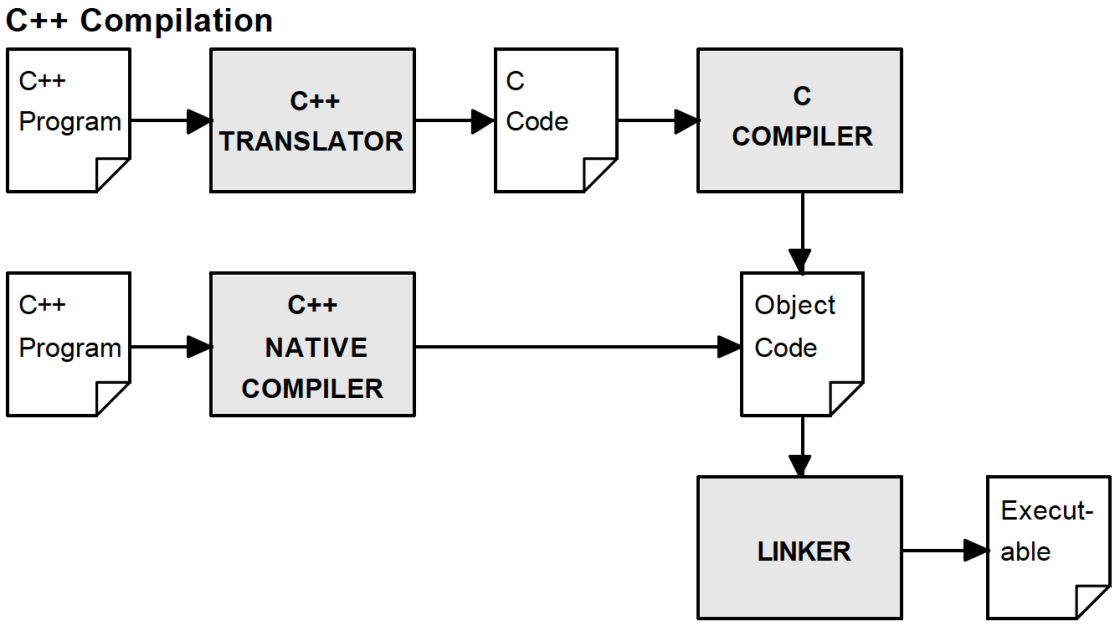

Figure courtesy: C++ Essentials by Sharam Hekmat

Concept: Function Calls

Functions Basics

- A block of code
  - Main objective: to encapsulate functionality for reuse
  - Others:
    - improve programmer productivity
    - reduce effort to understand code by hiding unnecessary details
- Concept of a function signature
  - Combination of **return type**, **function name** and **data type of arguments**
    - **void add** (**int** arg1, **int** arg2)
    - **int add** (**int** arg1, **int** arg2)
    - **void add** (**int** arg1, **float** arg2)
    - **double add** (**int** arg1, **int** arg2)
    - …
  - All functions in a program should have a unique function signature

Functions Basics

- Types
  - Returns nothing (void) v/s a value
  - Pre-defined v/s user-defined

- Steps in creating and using a function
  - Defining: the functionality
  - Calling: invocation
  - Declaring: the signature // Optional if code consists of only one file, invocation happens after defining

Concept: User Defined Types

User Defined Types

- When we start solving real world problems, we often use a set of information together
  - Examples:
    - Name = {title, first-name, middle-name, last-name, suffix}
    - Address = {Street name, Number, City, State, Zip code}
  - Need not be of the same type as a language's pre-determined / basic data types
- May be of the same or different basic types

Why User Defined Types ?

- Ease of writing and maintaining code
  - Abstraction helps in communication
  - Code is easier to understand
  - Code becomes easier to test
- No impact on code's executional performance
- C++ concepts
  - Struct
  - Class (Object Oriented Programming)

Code Illustration

- Problem
  - Represent person details
- Example: **Function:** demoStruct()

Concept: Memory Management

Types of Memory Allocation

- Static: when size is known at compile time
  - Basic data types
  - Arrays
- Dynamic: when size not known ahead of time
  - User defined types
  - Linked list

Advantages and Disadvantages of Dynamic Allocation

- Advantages
  - Does not waste memory
  - Does not ask user for information (size, data type) user may not know
  - Eases supporting of user-defined types
- Dis-advantages
  - Requires developer to be careful with allocation and de-allocation
  - Code is often complex compared to static allocation

Code Illustration

- **Problem**: Get average of a set of random numbers
- Note
  - Numbers not known ahead of time
  - How much to allocate?
    - Too high: wasting space
    - Too low: program will behave incorrectly

Code Illustration

- Static allocation – **Function:** demoStaticMemoryAllocation()

- Dynamic allocation – **Function:** demoDynamicMemoryAllocation()

Class Exercise

Reading .csv File

	A	B	C	D
1	Name	Height (cms)	Weight (kgs)	BMI
2	P1	60	10	27.7777778
3	P2	100	30	30
4	P3	150	70	31.1111111
5	P4	200	90	22.5

```
Name ,Height (cms),Weight (kgs),BMI
P1,60,10,27.77777778
P2,100,30,30
P3,150,70,31.11111111
P4,200,90,22.5
```

Task A: write a program to read this file

Task B: when the number of rows are unknown

Task C: when the number of columns is unknown

Task D: when the number of rows and columns are unknown?

BMI = Weight in kgs / (Height in cm /100)^2)
= C2 / (B2/100)^2)

Source: https://www.cdc.gov/healthyweight/assessing/bmi/childrens_BMI/childrens_BMI_formula.html

Home Work 2

Programming Home Work (#2) – C++

- Write a program called GeometricPropertyCalculator.
  - The program reads an input file (called input.txt). Each line in the file contains dimensions of a geometric shape – rectangle, shape and triangle. Specifically:
    - For rectangle, it contains – RECTANGLE <length-in-cm> <breadth-in-cm>
    - For circle, it contains – CIRCLE <radius-in-cm>
    - For triangle, it contains – TRIANGLE <side-1-in-cm> <side-2-in-cm> <side-3-in-cm>
  - The user specifies the property to calculate as argument to the program: 1 for AREA and 2 for PERIMETER
  - The program writes output lines to an output file (called output.txt) for each shape that it reads and the property – AREA or PERIMETER.
    - For example, for RECTANGLE and property as AREA, the program should write – RECTANGLE AREA <calculated value>
  - Write GeometricPropertyCalculator in C++
    - It should support RECTANGLE, CIRCLE and TRIANGLE
    - It should support properties AREA and PERIMETER
    - If there is insufficient information, the program should give an error. E.g. TRIANGLE AREA “Not enough information to calculate”

Programming Home Work (#2) – C++

- Code guidelines
  - Have sub-directories in your folder
    - src sub-folder, (or code) for code
    - data sub-folder, for input.txt and output.txt
    - doc sub-folder, for documentation on what the code does or sample output

- Hint
  - Area
    - Rectangle: length x breadth
    - Circle: pi * r^2
    - Triangle: -
  - Perimeter
    - Rectangle: 2 * (length + breadth)
    - Circle: 2 * pi * r
    - Triangle: sum of all three sides

Discussion: Programming Assignment # 1 Course Project

Course Project – Building and Assembling of Prog. Assignments in Health

- **Project**: Develop collaborative assistants (chatbots) that offer useful information about diseases
- Specifically, use the CDC dataset on diseases at: https://wwwnc.cdc.gov/travel/diseases
  - For polio, it is: https://wwwnc.cdc.gov/travel/diseases/poliomyelitis
  - Each student will choose two diseases (from 47 available).
  - Each student will also use data about the disease from WebMD. Example for polio - https://www.webmd.com/children/what-is-polio
  - Programming assignment programs will: (1) extract data about a disease from two sites, (2) process it, (3) make content available in a command-line interface, (4) handle any user query and (5) report on interaction statistics.

Discussion: Nature and Simplifications

- Once you select a disease, the content is also fixed.
  - Enter selection in column F of spreadsheet
- Some simplifications
  - **Download local copy** v/s web query
  - **Read static content first**
  - **Handle a subset of content**
  - **Have default handling for questions** the chatbot does not understand
- Do project in a language you are most comfortable with
- Use all advanced programming concepts to simplify coding

Discussion: Chatbot Loop

- Input: from user (called utterance)
  - Problem specific query (i.e., about disease chosen)
  - Chitchat
  - Unrelated
- Output: from system (response)
  - Handle unrelated
  - Handle chitchat
  - Answer to query
- **Do it until user say over!**

Handling different data types

Show formatted content!

Core Programs Needed for Project

- Prog 1: extract data from the disease pages
- Prog 2: process it 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

Programming Assignment # 1

- Goal: extract data from the disease of choice
  - Language of choice: Any from the three (C++, Java, Python)
- Program should do the following:
  - Take disease as input
  - Read content about the disease
    - from the disease's URL from CDC and WebMD, OR
    - a local text version of the disease pages // Keep them as separate files with names <disease>-<source>.txt
  - Report statistics of content: lines, words, chars
  - Write content out in an output file formatted with indentation
- Code organization
  - Create a folder in your GitHub called "prog1-extractor"
  - 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 update Google sheet

Concluding Section

Lecture 5: Concluding Comments

- We discussed
  - the concept of functions
  - the concept of user-defined types
  - the concepts of static and dynamic memory allocation
- Discussed Home Work 2 (due Thursday, Jan 26)
  - Peer evaluation in class
- Discussed Programming Assignment #1 (due Thursday, Feb 2)

About Next Lecture – Lecture 6

Lecture 6: Object Oriented Concepts

- Home work 2 due
  - Peer evaluation in class
- Concepts: Classes and Objects
- Project: Chatbots Background