Lecture 01 - Getting Started

CS 1342 - SMU

Mark Fontenot, PhD

May 21, 2019

Agenda

- Data Types and Variables
- Practice with Control Structures

Data Types and Variables

Variable Declarations

```
1   int x;
2   int y = 10;
3   int z = f();
```

- **Line 1**: Variable Declaration
- **Line 2**: Variable Declaration and Initialization
- **Line 3**: z is declared and initialized with the value returned from function f.
- it is always a good idea to initialize your variables when you declare them.

This Isn't Math Class

- `x = 10;` is not equivalent to `10 = x;`
  - `=` is assignment operator, not statement of equivalence.
  - assignments always have destination on the left of `=` and source on the right.
- Numeric data types are not unbounded
  - `-3000000000000` is NOT a valid int value in C++.
  - See page 25 of Halterman for bounds.

Variable Name Best Practices

- use camel case for naming regular variables
    - `finalGrade`, `calculatedValue`
- only use generally accepted/known abbreviations; full words are better
    - `calVal` ?? “California’s Value” or “calculated value”?
- names of constants should be all upper case
    - `PI`, `NUMBER_STUDENTS`
    - sometimes we use the _ for space in constant names so they are easier to read

Character Data Type

```
char letterGrade = 'A';
letterGrade = 'B';
```

- a **char** stores a single ASCII Character
- ‘Under the hood’, it is stores as an integer value.
  - ‘A’ is 65
  - ‘a’ is 97
- See Halterman Table 3.4 for ASCII code to character mapping table.

Escape Sequences

- Remember those **escape sequences** from Java??
    - `\n \t \0`, etc.
    - They exist in C++ as well.
    - `\n` (new line) and `\0` (null character) are the ones you'll use most frequently
    - `\\` if you want to print a single backslash

```
cout << "Hello\nWorld";
```

Practice with Control Structures

Problem 1 - Seeing Stars

Ask the user to enter an integer representing the number of stars they’d like to print to the screen. Then, print that many stars (asterisks) to the screen.

Problem 2 - Seeing Organized Stars

Modify your solution to Problem 1 such that it ensures the number of stars entered by the user falls between 1 and 200 inclusive.

Then, modify it so that it will not print more than 10 stars per line. For example, if the user entered ‘33‘, your program would print 3 lines containing 10 stars and 1 line containing 3 stars.

Problem 3 - More Organized Now

You're going to print some fun figures for the user based on a dimension they enter. Allow the user to enter a the height for the figure and then print a square of asterisks that using the value entered as the number of lines of stars as well as the number of stars on each line.

For example: assume the user enters 5. Your program would print:

```
*****
*****
*****
*****
*****
```

Function Basics

Parts Of a Function Definition

1. Name
2. Return Type
3. Parameter List
4. Body

```
void printSquareOfStars(int val)
{
    //Code here
}
```

Some differences with Java

- Functions must be declared above where they can be called; they DO NOT need to be fully defined.
- So, then, what is the difference between a function declaration and function definition?
    - glad you asked…

```
void printSquareOfStars(int val); //prototype
int main() {
    int x;
    cin >> x;
    printSquareOfStars(x); //call
}
void printSquareOfStars(int val){ //definition
    //some code here
}
```

Parameters

- **Parameters** are values that allow the function to be more generic and work for different data inputs.
- *Imagine a world where calculators had 1 button for every possible angle for which you could find the sin, cos, and tan. How big would that calculator be?*
- Nomenclature:
    - **argument -** the thing that is in the function call
    - **parameter -** the thing that is in the header of the function definition
- **Pass by Value**: When a function is **called**, the *argument* is copied into the *parameter*.

Problem 4 - Functions of Stars

Convert Problem 3 into a program that has a function that prints the figure.

More Problems

Problem 5 - Filled in Triangle

Add a function to your previous solution that will print a triangle similar to the following based on the height value entered by the user. Name your function printRightTriangle.

Assuming the user entered 5:

```
*
**
***
****
*****
```

Problem 6 - Empty Parallelogram

Add another function to your solution … yada yada yada It should print an empty parallelogram based on the value entered.

Assuming the user entered 5:

```
    *****
   *   *
  *   *
 *   *
*****
```

Problem 7 - The Challenged Problem

Add another function to … blah blah blah. It should print a striped diamond based on the value entered. Assuming the user entered 5

```
    *
   * *
  *****
 *     *
*********
 *     *
  *****
   * *
    *
```