Programming with C and C++

CSC-101 (Lecture 4)

Dr. R. Balasubramanian

Professor
Department of Computer Science and Engineering
Mehta Family School of Data Science and Artificial Intelligence
Indian Institute of Technology Roorkee
Roorkee 247 667

bala@cs.iitr.ac.in
https://faculty.iitr.ac.in/cs/bala/

Motivation

From Numbers to Code

Trivia:

- Think of your favourite number. Can you represent it in binary form?
- How would you store the number π in a computer, given it's an irrational number with infinite decimal expansion?

Mathematical Patterns in Programming

- Fibonacci Series: 0, 1, 1, 2, 3, 5, 8, 13, ...
- Prime Numbers: 2, 3, 5, 7, 11, 13, ...

Question: If you were to instruct a computer to generate these sequences, where would you start?

Puzzles and Logic

Puzzle:

- Consider the equation: $2x + y = 100$. Given that $x, y \in \mathbb{N}$, how many solutions can you think of?

Question:

- How might a computer systematically find all these solutions?
- Computers can't directly understand algebra, but they can execute commands based on logical and mathematical operations.
- How do you think a computer might solve the equation $3x - 4 = 11$, where $x \in \mathbb{Z}$?

Geometry in Programming

Problem:

- Consider a circle with radius r. Can you write a mathematical expression for its area and circumference?

Question:

- How would you instruct a computer to calculate these properties for a given radius? Think of the functions and constants involved.

Basics of C

Defining a Function

Mathematics:

- Defined relation between sets.
- Ex.: $f(x) = x^2$
- Input: x
- Operation: Squaring
- Output: x^2

C Programming:

- Encapsulates a set of instructions.
- Ex.: `int square(int x){ return x * x;}`
- Input: `x`
- Operation: Multiplication
- Output: `x * x`

Function Characteristics

Mathematics:

- Deterministic: One input has one specific output.
- Predictable: You can always determine the outcome with the given input.
- Represents a rule or a transformation.

C Programming:

- Deterministic: Functions will produce the same result with the same input.
- Predictable: If coded correctly, no surprises.
- Encapsulates a behaviour or task.

The Power of this Analogy

- Both paradigms focus on the transformation: Converting inputs into desired outputs.
- By understanding one, we have a head start on understanding the other.
- Just as we manipulate mathematical functions to solve problems, we can manipulate C functions to program solutions.

Mathematics is the gate, and programming is the key.

Another Analogy: Ingredients and Variables

Kitchen Recipe:

- Ingredients: Flour, Eggs, Sugar
- Quantities: 1 cup, 2 eggs, 0.5 cup
- Essential to achieve the final dish.

C Programming:

- Variables: ‘int’, ‘char’, ‘float’
- Values: ‘5’, “a”, ‘3.14’
- Essential to store and manipulate data.

Steps and Instructions

Kitchen Recipe:

- Sequential steps: Mix, Bake, Serve
- Each step affects the outcome.
- Order is crucial.

C Programming:

- Sequential instructions: Initialize, Compute, Print
- Each instruction performs a task.
- Execution order is paramount.

Outcomes and Outputs

Kitchen Recipe:

- Desired Outcome: A tasty cake
- Result of following steps with ingredients.

C Programming:

- Desired Output: Correct results, no errors.
- Result of executing instructions with variables.

The Essence of the Analogy

- Both paradigms are about transformation: Ingredients to a dish, Variables to an output.
- Just as you’d troubleshoot a recipe, you debug a program.
- The beauty of creation: Crafting a dish or crafting a solution.

Programming, like cooking, is an art. Mastery comes with practice and understanding.

A Simple Task: Finding the Area of a Rectangle

Task:

- Calculate the area of a rectangle with a length of 5 units and a width of 7 units.
- Mathematically: $\text{Area} = \text{length} \times \text{width}$

C Program:

```
#include<stdio.h>

int main() {
    int length = 5;
    int width = 7;
    int area = length * width;
    printf("Area = %d\n", area);
    return 0;
}
```

Dissecting the Program: Preprocessor Directive

What is this?

```
#include<stdio.h>
```

- It’s a preprocessor directive.
- Tells the compiler to include the standard input-output header file.
- Enables the use of ‘printf’ and other I/O functions.
- Think of it as importing a toolkit before starting work.

Dissecting the Program: The main() Function

Function:

```
int main() {
```

- It’s the entry point for every C program.
- The execution starts from here.
- Contains the key instructions to run.
- Returns an integer to the operating system upon completion.
- This is similar to a mathematical function having both input and output.

Dissecting the Program: Variables

Variable Declaration:

```
int length = 5;
int width = 7;
```

- Variables ‘length’ and ‘width’ store the dimensions of the rectangle.
- ‘int’ denotes the integer data type.
- Variables are initialized with given values.
- They play a role similar to constants in algebra.

Dissecting the Program: Calculating Area

Calculation:

```
int area = length * width;
```

- Multiplication operation calculates the area.
- The result is stored in the ‘area’ variable.
- Direct application of the mathematical formula.

Dissecting the Program: Printing the Result

Printf:

```
printf("Area = %d\n", area);
```

- Displays the result on the console.
- ‘%d’ is a placeholder for integers.
- It will be replaced by the value of ‘area’.
- It’s like evaluating a function with a given value.

Dissecting the Program: The Return Statement

Return Statement:

```
return 0;
```

- Signifies the successful termination of the program.
- Returns an integer value (0 in this case) to the operating system.
- Like concluding a mathematical operation or proof.

Thank You and Keep Coding!

"Don’t be pushed around by the fears in your mind. Be led by the dreams in your heart."

- Roy T. Bennett