Welcome to Programming 1: C++ Foundations Review

Solidifying the Basics for Advanced Concepts

What We'll Review Today

- **Why This Review Matters**
- **The C++ Program Skeleton**
- **Variables & Data Types**
- **Input & Output (**cin **&** cout**)**
- **Operators: Arithmetic, Relational, and Logical**
- **Control Flow:** if**,** else if**,** else
- **Loops:** while**,** do-while**,** for
- **A Glimpse Ahead: What's Next in Programming 2**
- **Q&A**

Building on a Strong Foundation

- **Programming 1:** You learned the *words* and *grammar* of C++.
  - (Variables, simple commands, basic logic)
- **Programming 2:** You will learn to write *essays* and *stories*.
  - (Functions, Pointers, Arrays, Recursion - complex problem-solving)
- **Weak Foundation = Unstable Building.**
- **Goal:** Ensure everyone is confident with the basics before we add new floors.

The Hello World Template

- **Key Components:**
- #include <iostream>: Allows input/output operations.
- using namespace std;: Lets us use standard names (like cout, cin) without std::.
- int main(): The starting point of every C++ program. Execution begins here.
- return 0;: Indicates the program ended successfully.

```
#include <iostream>
using namespace std;

int main() {
    // Your code goes here

    return 0;
}
```

Storing Information in Memory

Data Type	Size (Typical)	Description	Example
int	4 bytes	Whole numbers	int age = 19;
double	8 bytes	Decimal numbers	double price = 9.99;
float	4 bytes	Decimal numbers (less precise)	float temperature = 36.6f;
char	1 byte	Single character	char grade = 'A';
string	Varies	Sequence of text	string name = "Alice";
bool	1 byte	True or False	bool isRaining = true;

1. **Key Concept: Declaration & Initialization:** `int score;` (declaration) `score = 100;` (assignment) or `int score = 100;` (initialization).

Communicating with the User

•cout (console out): Uses the insertion operator << to send data to the screen.

•cin (console in): Uses the extraction operator >> to get data from the user.

•endl: Inserts a newline and flushes the output buffer.

Speaker Notes:

```
#include <iostream>
using namespace std;

int main() {
    string name;
    int year;

    cout << "What is your name? "; // OUTPUT
    cin >> name;              // INPUT

    cout << "Hello " << name << "!" << endl;
    cout << "What year is it? ";
    cin >> year;

    cout << "You said the year is: " << year << endl;

    return 0;
}
```

Performing Operations

- **Arithmetic:** +, -, *, /, % (modulus - remainder)
  - int result = 10 % 3; // result is 1
- **Relational (Compare):** ==, !=, >, <, >=, <=
  - Return a bool value (true or false).
  - bool isEqual = (5 == 5); // isEqual is true
- **Logical (Combine Conditions):** && (AND), || (OR), ! (NOT)
  - if (age >= 18 && hasLicense) { ... }

Making Decisions in Your Code

```
int score;
cout << "Enter your test score: ";
cin >> score;

if (score >= 90) {
   cout << "Grade: A" << endl;
}
else if (score >= 80) {
   cout << "Grade: B" << endl;
}
else if (score >= 70) {
   cout << "Grade: C" << endl;
}
else {
   cout << "Grade: F" << endl;
}
```

- The program evaluates conditions from top to bottom.
- As soon as one condition is true, its block executes, and the rest are skipped.

Repeating Actions

while **Loop (Pre-test):** Checks condition *first*.

```
int count = 1;
while (count <= 5) {
   cout << "Count is: " << count << endl;
   count++; // Don't forget this! Avoid infinite loops.
}
```

do-while **Loop (Post-test):** Executes *at least once*.

```
int number;
do {
   cout << "Enter a positive number: ";
   cin >> number;
} while (number <= 0);
```


The Count-Controlled Loop

```
// Syntax: for(initialization; condition; update)
for (int i = 0; i < 5; i++) {
   cout << "i = " << i << endl;
}
```

Output:

```
i = 0
i = 1
i = 2
i = 3
i = 4
```

What's Next? Building on the Foundation

- **Functions:** Breaking down code into manageable, reusable blocks.
  - "Stop repeating yourself! Write it once, use it many times."
- **Arrays:** Storing and processing large collections of data (e.g., 100 test scores).
  - "This is where for loops become your best friend."
- **Pointers:** Understanding and directly manipulating memory addresses.
  - "The power and the peril of C++. This is a key differentiator."
- **Recursion:** Functions that call themselves to solve complex problems elegantly.
  - "A new way of thinking about repetition and problem-solving."