Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

Session: Structure of a Simple C++ Program

Quick Recap of Some Relevant Topics

- Dumbo model of computing
- Notion of instructions
- Notion of memory
  - Reading and writing values in memory
- Notion of input and output
- Intuitive idea of program

Overview of This Lecture

- Understand structure of a simple C++ program
  - Some common features
  - Significance of features
  - Example program

- There's more to structure of C++ programs
  - We'll see these later in the course

Our Friendly C++ Program

IIT Bombay

```
// file add_two_numbers.cpp
#include <iostream>
using namespace std;
// this program reads two integers
// and calculates the sum
int main() {
   int A, B, C;
   cout << "Give two numbers";
   cin >> A >> B;
   C = A + B;
   cout << "Sum is" <<  C;
   return 0;
}
```

- A program is a sequence of directives, declarations and instructions
- Written according to some *rules*
  Computer languages also have grammars!
- Stored in one (or more) files
  e.g. add_two_numbers.cpp

Our Friendly C++ Program ...

IIT Bombay

```
// file add_two_numbers.cpp
#include <iostream>
using namespace std;
// this program reads two integers
// and calculates the sum
int main() {
   int A, B, C;
   cout << "Give two numbers";
   cin >> A >> B;
   C = A + B;
   cout << "Sum is" <<  C;
   return 0;
}
```

Compiler Directives

Compiler Directives

IIT Bombay

- Instructions to compiler used when compiling your program to machine language

```
#include <iostream>
```

- Include instructions from “iostream” header file
- Input/output handled as ‘streams’ of bytes
  - Input stream converted to internal representation of computer
  - Internal representation converted to stream of displayable characters
- “cin” (for keyboard input) and “cout” (for console output) work because of instructions in “iostream” header file

Compiler Directives

using namespace std;

- Names of objects in a program can be grouped in “namespaces”
  - Analogy: Names of students in CS101 grouped in divisions
- Each “namespace” can be given a name
  - Analogy: Each division can be given a name
- When referring to a name, we must indicate which “namespace” we are referring to
  - Analogy: Shyam from division E
  - C++ programs: myNameSpace::myVarName
- Can use same name in different namespaces
  - Shyam from division E different from Shyam from division A
  - myNameSpace1::myVarName different from myNameSpace2::myVarName
- “using namespace std” asks compiler to use a global namespace called “std”
  - myVarName refers to std::myVarName

Our Friendly C++ Program ...

IIT Bombay

```
// file add_two_numbers.cpp
#include <iostream>
using namespace std;
// this program reads two integers
// and calculates the sum
int main() {
   int A, B, C;
   cout << "Give two numbers";
   cin >> A >> B;
   C = A + B;
   cout << "Sum is" <<  C;
   return 0;
}
```

"main" function begins here

"main" function ends here

“main” function

IIT Bombay

- Literally, the **main** part of the program
- Operating system invokes this function when you run your compiled program
  - Can pass one or more parameters to this function
  - None passed in our example
- Operating system gets back control of computer when this function ends

Our Friendly C++ Program ...

IIT Bombay

```
// file add_two_numbers.cpp
#include <iostream>
using namespace std;
// this program reads two integers
// and calculates the sum
int main() {
  int A, B, C;
  cout << "Give two numbers";
  cin >> A >> B;
  C = A + B;
  cout << "Sum is" <<  C;
  return 0;
}
```

Variable declarations

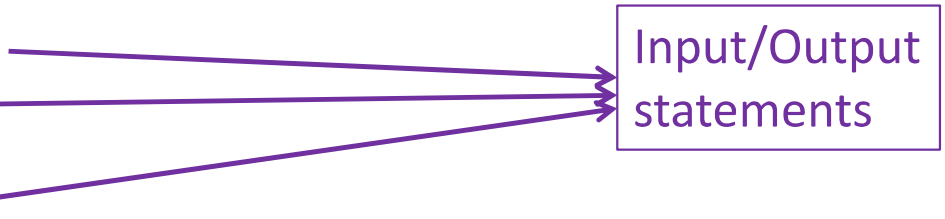
Variables and Declarations

- Variables
  - Recall the drawers of Dumbo
  - Basic computational objects
- Declarations
  - Each variable must be named
  - For each variable, we must indicate type of value it can store
- A variable must be declared before it is used
- Remember “using namespace std”
  - int A, B, C; really means int std::A, std::B, std:C;
  - Isn’t it convenient to use “using namespace std”?

Our Friendly C++ Program ...

IIT Bombay

```
// file add_two_numbers.cpp
#include <iostream>
using namespace std;
// this program reads two integers
// and calculates the sum
int main() {
   int A, B, C;
   cout << "Give two numbers";
   cin >> A >> B;
   C = A + B;
   cout << "Sum is" <<  C;
   return 0;
}
```



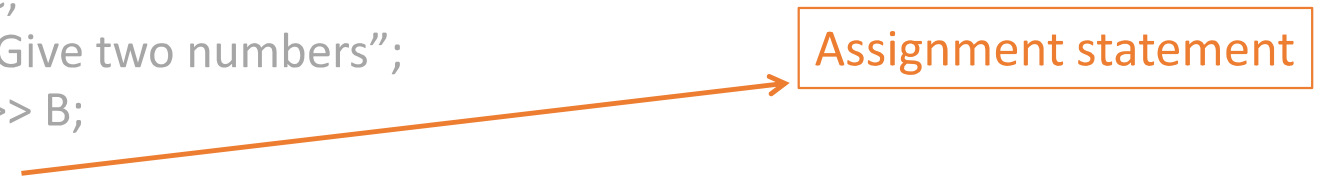
Input/Output Statements

- This is how we interact with the program
- “cin” allows input from keyboard
- “cout” displays/outputs on console
- Essential components of most programs we will write

Our Friendly C++ Program ...

IIT Bombay

```
// file add_two_numbers.cpp
#include <iostream>
using namespace std;
// this program reads two integers
// and calculates the sum
int main() {
   int A, B, C;
   cout << "Give two numbers";
   cin >> A >> B;
   C = A + B;
   cout << "Sum is" <<  C;
   return 0;
}
```



Assignment (and other) Statements

- This is where the program does the really interesting part
- C = A + B ;
  - A + B: Arithmetic expression
  - C: Destination of assignment
  - Instruction:

    Add values in variables A and B, and store it in variable C
- We'll see lots more examples of statements that allow us to do fantastic stuff with our programs

Our Friendly C++ Program ...

IIT Bombay

```
// file add_two_numbers.cpp
#include <iostream>
using namespace std;
// this program reads two integers
// and calculates the sum
int main() {
   int A, B, C;
   cout << "Give two numbers";
   cin >> A >> B;
   C = A + B;
   cout << "Sum is" <<  C;
   return 0;
}
```

Return control back to caller (here, OS), and pass the value 0

“return” Statement

IIT Bombay

- A function is called/invoked
  - By operating system (e.g. “main” function)
  - By another function
- “return” returns control to caller
  - In our example, the operating system
- Can also return a value to caller
  - Useful for returning result of computation
  - Useful for indicating error status

Our Friendly C++ Program ...

IIT Bombay

```
// file add_two_numbers.cpp
#include <iostream>
using namespace std;
// this program reads two integers
// and calculates the sum
int main() {
  int A, B, C;
  cout << "Give two numbers";
  cin >> A >> B;
  C = A + B;
  cout << "Sum is" <<  C;
  return 0;
}
```

A logical block of statements

Grouping Statements

- In C++, statements can be grouped by enclosing them within a pair of '{' and '}'
  - Each group is treated as one logical unit of the program
  - Useful for demarcating logical parts of a program
  - Each group can have its own variable declarations

Our Friendly C++ Program ...

IIT Bombay

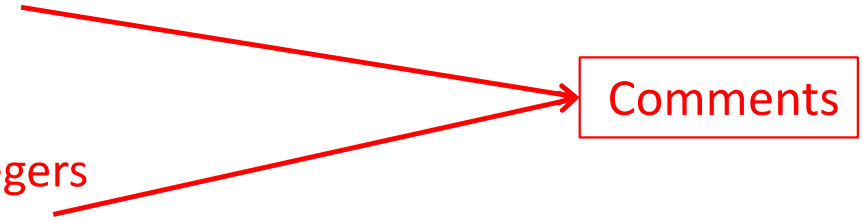
```
// file add_two_numbers.cpp
#include <iostream>
using namespace std;
// this program reads two integers
// and calculates the sum
int main() {
  int A, B, C;
  { cout << "Give two numbers";
  cin >> A >> B; }
  C = A + B;
  {cout << "Sum is" <<  C;
  return 0; }
}
```

Hierarchically grouped blocks

Our Friendly C++ Program ...

IIT Bombay

```
// file add_two_numbers.cpp
#include <iostream>
using namespace std;
// this program reads two integers
// and calculates the sum
int main() {
   int A, B, C;
   cout << "Give two numbers";
   cin >> A >> B;
   C = A + B;
   cout << "Sum is" <<  C;
   return 0;
}
```



Comments

- Essential for good readability of program
- Can appear anywhere in program
- Completely ignored by compiler
- Good programming practice
  - Insert adequate comments throughout your code
  - Make your program speak its story through good comments

Summary

IIT Bombay

- Structure of a simple C++ program
  - Compiler directives
  - “main” function
  - Variable declarations
  - Input/output statements
  - Assignment (and other) statements
  - “return” statement
  - Grouping statements into logical blocks
  - Comments