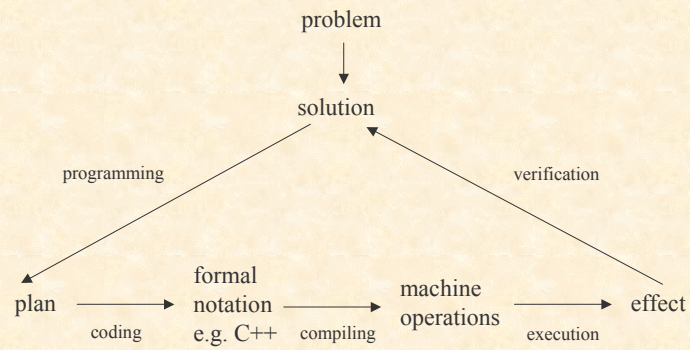


PROGRAMMING AND PROGRAM STRUCTURE

□ What is Programming



WHAT MAKES A GOOD PROGRAM

- Is the program correct ?
- Is the program easy to modify ?
- Is the program readable ? (e.g. Comments)

WHAT IS C++ ?

□ What is C++ ?

- In the 1960s a programming language named BCPL (Basic Combined Programming Language) has been used primarily in Europe.
- From BCPL, another language arose with its name abbreviated to B
- In the late 1960s and early 1970s, Dennis Ritchie at AT&T Bell Labs adopted features from the B language to develop a new language named C. C as a system programming language was used to reprogram the UNIX operating system.
- In 1985 Bjarne Stroustrup at AT&T Bell Labs, a new programming language. To the C language he added features for data abstraction and object oriented programming. Instead of naming the language D, the Bell labs group named it C++.

WHY C++ ?

□ Why C++ ?

- A "better" C
- Backwards compatible (with C)
- Not state-of-the-art in terms of programming languages but becoming state-of-the-industry
- Includes most recent advantages in programming: Object Oriented and Abstract Data Types
- Small but powerful

C++ IS AN IMPERATIVE LANGUAGE

- ❑ Runs as **ordered** sequences of instructions

Step 1 → Step 2 → Step 3 → Step 4 → etc

Programs are executed by carrying out each instruction in the specified order

!!!Changing the order changes the meaning of the program!!!

Example:

not the same as

- | | | | |
|---------------|------|---------------|------|
| 1. Go forward | then | 1. Go forward | then |
| 2. Go left | then | 2. Go right | then |
| 3. Go forward | then | 3. Go forward | then |
| 4. Go right | then | 4. Go left | then |
| 5. Go left | | 5. Go right | |

IMPERATIVE PROGRAMMING

- ❑ Problems decomposed into series of steps which can be represented by sequence of statements written in C++

When a program is run each statement is executed in sequential order

- ❑ Two additional concepts
 - control structures - to control order in which instructions are executed
 - idea of state - a program goes from one state to the next with the goal of going from beginning state to end state

CONTROLLING EXECUTION

□ Imperative programs provide:

- Sequences - allows order of instructions to be specified
- Iteration - allows a group of instructions to be repeated. The program remains at the same 'depth'
- Recursion - allows a group of instructions to be repeated. The program changes in 'depth' - embedded
- Selection - allows instructions to be executed depending on a particular condition

Iteration and recursion may be **bounded** or **unbounded**

Example: "Each week go to play tennis."

"During July, go to swim every Thursday."

STATE

□ Lists of instruction on their own are not very useful unless they are able to manipulate something

The idea of **state** provides values that can be manipulated by instructions

□ The execution of C++ programs is described by a **state machine** model

- Instructions cause the model to go from one state to another
- The **initial state** represents the conditions at the beginning of the program before computation has started (input)
- The **final state** represents the conditions at the end of the program (output and result)

□ The state holds all the information describing the current status of the computation

PROGRAMS AND FILES

- ❑ The source code of a program may be split up into a number of files - this aids software development
 - Each file normally contains a logically related collection of declarations and definitions

- ❑ For example, a program could consist of the following source files:

```
main.cc
part1.cc
calc.cc
calc.h
```

- ❑ The definitions are placed in the three cc files, while declarations required for calc.cc are in calc.h

COMPOUND STATEMENT

- ❑ C++ statements may be grouped together in a compound statement using the operators {},

```
{
    statement 1
    statement 2
    statement 3
}
```

e.g.

```
{
    Go right
    Go forward
    Go left
}
```

Compound statements are also known as code blocks

A FIRST C++ PROGRAM

```
/* This statement is required to be able to do I/O */
#include <iostream.h>

// This is the main function. There must be one and only
// one function of this name in a program

int main ()
{
    // Print to the computer's output display
    cout << "Hello world!" << endl;
    return 0;
}
```