Programming Languages

Lecture 3

Robb T. Koether

Hampden-Sydney College

Fri, Aug 30, 2019

1. Programming Languages
2. Building an Executable Program
3. Example
4. Program Organization
5. Examples

Outline

1. Programming Languages
2. Building an Executable Program
3. Example
4. Program Organization
5. Examples

High-Level Languages

C++ Code

```
cin >> cost;
if (cost > 100.0)
{
    discount = 0.10 * cost;
    price = cost - discount;
}
else
    price = cost;
cout << price;
```

- Instructions use some English words (if, else, while, etc.)
- Expressions are written in something resembling standard algebraic notation.

Outline

1. Programming Languages
2. Building an Executable Program
3. Example
4. Program Organization
5. Examples

Program Translation

- A computer is capable of interpreting and executing only machine language (numerical) instructions.
- Humans (almost always) write programs in high-level languages such as C++.
- Therefore, a C++ program must be *translated* into machine language in order to be run.
- The translation process is referred to as compiling.

Compilers

- A compiler is a program that translates (compiles) a high-level language such as C++ into machine language.
- The compiler must first check the syntax of the source code.
- A syntax error is a violation of the grammar rules of the language.
- If there are syntax errors, then error messages are displayed and the process stops.

Assemblers

- Most compilers produce assembly code as an intermediate step.
- Assembly code is an intermediate-level language that is still somewhat readable, but also very close to machine language.
- An assembler translates the assembly code into machine language.
- If there are no syntax errors, then the assembler produces the object code (machine language, mostly).

Linkers

- A linker combines the object code produced by the compiler with library functions (e.g., the square root function) needed by the program.
- If the linker is unable to locate a function, it reports a link error and the process stops.
- If there are no link errors, the linker produces a fully functioning program.

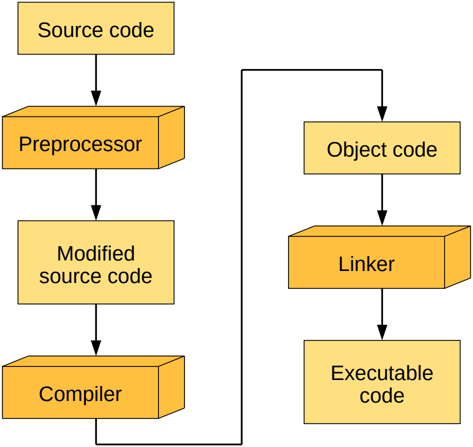
Loaders

- The loader copies the program into a suitable location in main memory and begins execution.
- A run-time (or logical) error is an error that occurs during execution.
- Typically, run-time errors cannot be detected by the compiler or the linker before execution.
- Division by zero would be a run-time error.

Division by Zero

```
a = 0;
b = 2/a;
```

Program Translation



Outline

1. Programming Languages
2. Building an Executable Program
3. Example
4. Program Organization
5. Examples

High-Level Languages

C++ Code

```
int c = 3*(10 + 20);
```

- C++ code to add 10 and 20 and multiply by 3.

Assembly Language

Assembly Code

```
li   $4,10
addi $5,$4,20
li   $4,3
mult $5,$4
mflo $5
```

- Each instruction represents a single basic operation at the machine level (add, multiply, assign, etc.)
- Mnemonics are used instead of words.

Machine Language

Machine Code

Hex	Binary	Assembly
2404000A	00100100000001000000000000001010	li $4,10
20850014	00100000100001010000000000010100	addi $5,$4,20
24040003	00100100000001000000000000000011	li $4,3
00A40018	00000000101001000000000000011000	mult $5,$4
00002812	00000000000000000010100000010010	mflo $5

- Each instruction represents a single basic operation.
- Each instruction is written entirely numerically.

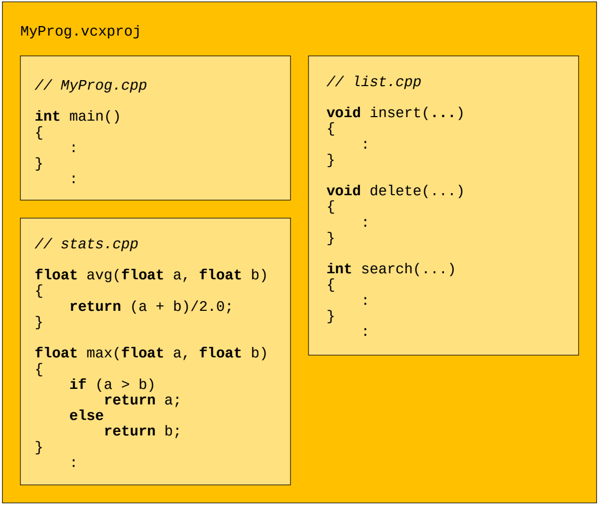
Outline

1. Programming Languages
2. Building an Executable Program
3. Example
4. Program Organization
5. Examples

The C++ Program Hierarchy

- A token is the smallest meaningful unit.
  - Number (`123`), variable (`cost`), operator (`+`), etc.
- A statement is a group of tokens that express a simple task.
  - `price = cost - discount;`
- A function is a group of statements that accomplish a task requiring several steps.
  - For example, insert a new name into a list of names.
- A file contains a collection of related functions.
  - For example, all functions that manage lists of names.
- A program consists of a collection of related files.

The C++ Program Hierarchy



MyProg.vcxproj

```
// MyProg.cpp

int main()
{
    :
}
    :
```

```
// stats.cpp

float avg(float a, float b)
{
    return (a + b)/2.0;
}

float max(float a, float b)
{
    if (a > b)
        return a;
    else
        return b;
}
    :
```

```
// list.cpp

void insert(...)
{
    :
}

void delete(...)
{
    :
}

int search(...)
{
    :
}
    :
```

Outline

1. Programming Languages
2. Building an Executable Program
3. Example
4. Program Organization
5. Examples

Examples of C++ Programs

- Example
  - `GradeReport.cpp`
  - `GradeStats.cpp`