CS11600-1
Intro to Programming II

Instructor:
Catalin Dumitrescu <cldumitr@cs.uchicago.edu>

TAs:
Irina Matveeva <matveeva@cs.uchicago.edu>
Zhimin Xie <zxie@cs.uchicago.edu>

Last Time

- Class Policies (syllabus)

- Tentative Schedule (syllabus)

- Introduction (lecture online)
  - Computer organization
  - Algorithms, programming languages
  - Interpreters & compilers

Today

- Introduction to C/C++ (non-OO -> procedural)

  - C/C++ Program Structure

  - Statements
    - simple (type-safe) I/O operations (*Section 2.4*)
    - control flow (*Section 2.6*)

  - Notions about
    - Source Files
    - Preprocessing (*Section 2.2*)
    - Compilation and Linking Processes

What is C++?

- A better C ...
- Supports data abstraction
- Supports object-oriented programming
- Supports generic programming

B. Stroustrup -- "*The C++ Programming Language*" -- Introduction

What is C?

Programming Paradigms

- **Procedural**
  - "*Decide which procedures you want; use the best algorithm you can find*"
- **Modular**
  - "*Decides which modules you want; partition the program so that data is hidden within modules*"
- **Data Abstraction**
  - "*Decide which types you want; provide a full set of operations for each type*"
- **Object Oriented**
  - "*Decide which classes you want; provide a full set of operations for each class; make commonality explicitly by using inheritance*"
- **Generic**
  - "*Decide which algorithms you want; parameterize them so that they work for a variety of suitable types and data structures*"

Program Structure

unit : external_declaration
| unit external_declaration

external_declaration : function_definition
| declaration

function_definition : *declarator* compound_statement

declaration : declaration_specifiers ';'
| declaration_specifiers init_declarator_list ';'

Program Structure (2)

compound_statement : '{' '}'
| '{' *statement_list* '}' | '{' declaration_list '}'
| '{' declaration_list *statement_list* '}'

declaration_list : declaration
| declaration_list declaration

statement_list : statement
| *statement_list* statement

Example *(e1.cc)*

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

int
main ()
{
   cout << "Hello World!" << endl;
   return 0;
}
```

Statements

- Declaration
- Expressions
- if
- switch
- while
- do … while
- for
- break & continue
- goto
- return
- Block instructions

Declaration

- Variables
  - int anInt = 20;
  - double aDouble = 20.5;
  - string aString = "abc";

- Types
  - typedef int integer;
  - typedef struct point_st { int x; int y; } point;
  - class point { int x; int y; };

Expression

- Arithmetic
  - 4 + 5
- Logic
  - 3 == 4
- Assignment
  - a = 2
- ...

if

if (expression)
    statement | compound_statement

Example:

```
if (1 + 2 == 3)
  cout << "3" << endl;
```

switch

switch (expression)
{
case expression: statement_list break;
default: statement_list break;
}

Example:

```
int a = 2;
switch (a) {
  case 2: cout << "2" << endl; break;
  default : cout << "not 2" << endl; break;
}
```

while

while (expression)
statement | compound_statement

Example:

```
int a = 1;
while (a <= 10)
  a = a + 2;
```

do ... while

do
statement | compound_statement
while (expression);

Example:

```
int a = 1;
do {
  a = a + 2;
} while (a < 10);
```

for

for (expression; expression; expression)
statement | compound_statement

Example:

```
for (int a = 1; a < 10; a = a + 2)
  cout << a << endl;
```

break & continue

```
while (expression)
   {
      ...
      break;
      ...
      continue;
      ...
   }
```

~~goto~~

goto label;

...

label:

return

return expression;

Example:

```
int main ()
{
  return 0;
}
```

Example (e2.cc)

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

int main () {
 long int i, t, t1, t2;

 for (i = 1; i <= 10; i++) {
   switch (i) {
      case 1: t = t1 = 0; break;
      case 2: t = t2 = 1; break;
      default: t = t2 + t1; break;
      }
   t1 = t2;
   if (i > 2)
      t2 = t;
  }
 cout << "The 100th term in the sequence is: " << t << endl;
 return 0;
}
```

Source Files

Preprocessing

- Directives
  - #include
  - #define, #undefine
  - #if ... #endif
  - #pragma

#include

```
#include <iostream>

#include "my_header.h"
```

#define, #undefine

- **#define term 10**

- **#define min(x,y) ((x) > (y) ? (y) : (x))**

#if ... #endif

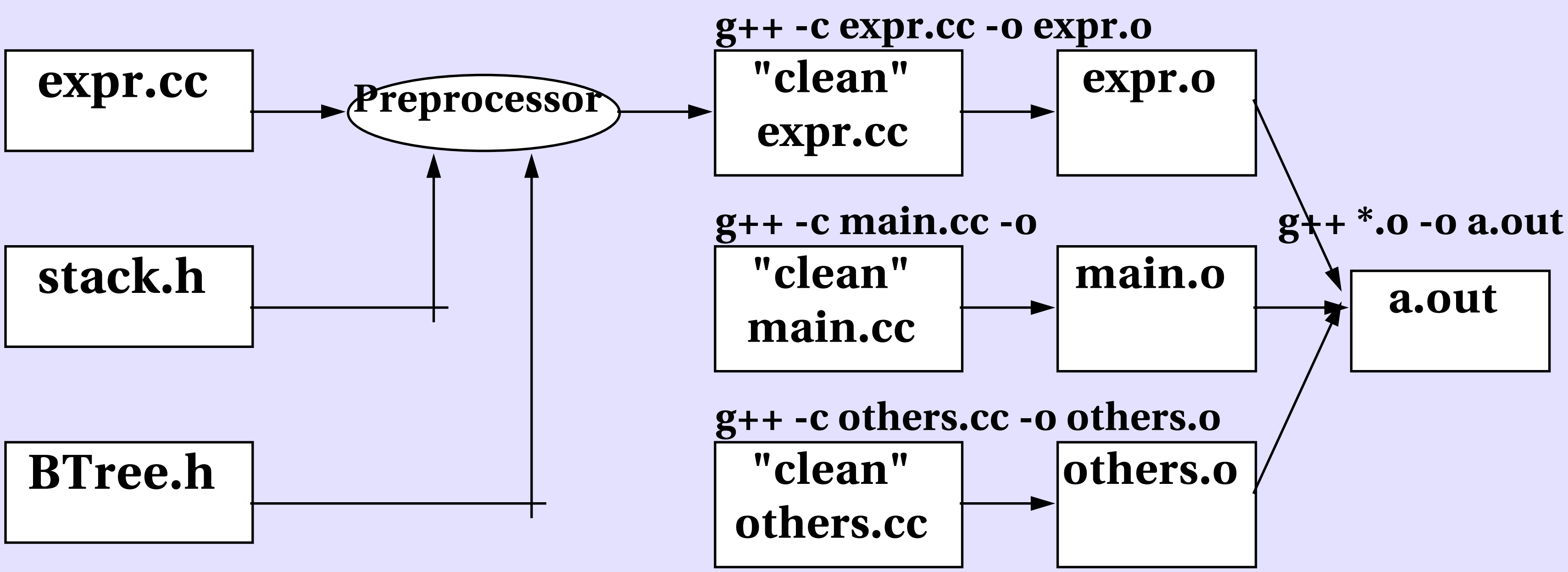
```
#ifdef DEBUG
    cout << "At " << i " t = " << t << endl;
#endif /* DEBUG */
```

#pragma

Example (e3.cc)

```
int main () {
  long int i, t, t1, t2;

#ifdef DEBUG
  int term;
  cout << "Input N:"; cin >> term;
#else
# define term 10
#endif /* DEBUG */

  for (i = 1; i <= term; i++) {
     switch (i) { ... }

#ifdef DEBUG
     cout << "At " << i " t = " << t << endl;
#endif /* DEBUG */

   }
  ..
}
```

Compilation & Linking



Linking

Homework

- Due date: January 10, 8:00pm
  - Problem 1/ Page 80 (Navigating C++)
  - Problem 3 / Page 80 (Navigating C++)
  - Write a C++ program that displays the Fibonnaci sequence up to the 100th term in reverse order

Next Time

- **Types (Section 2.1)**
- **Expressions (Section 2.5)**
- **Functions (Section 3.1)**
- **Arrays**