Advanced Programming in C++

فرشاد حکیم پور

Exercise

- Break the stack implementation (see slides 5 to 9) to declaration, and implementation part (similar to slide 20)
- Implement a queue class

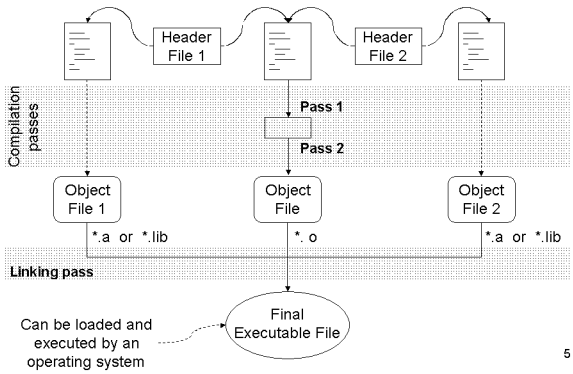
Header Files

- C header files contain *declarations*
- Conventional naming ".h" (old fashion: ".hxx" or ".hpp")
- A (C or C++) software library provides you with header files.
- Header files can be included in your program using "#include" preprocessor directive

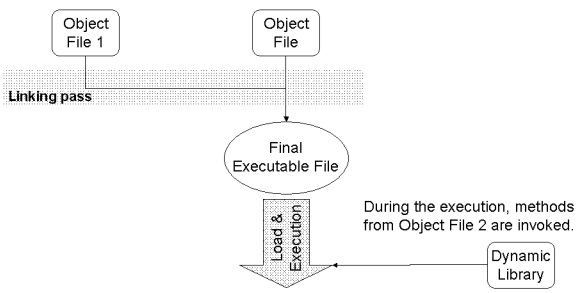
#include directive

- *Directives* start by # (hash-)sign such as:
  - `#include`
  - `#define`
  - `#ifndef`
  - `#endif`
- Directives are interpreted by a pre-compiler
- Include directive inserts the header file inside your program

Compilation Process



Static vs. Dynamic Link



Compilation phases

- Lexical Analysis
- Preprocessing
- Syntactic Analysis (parsing)
- Semantic Analysis
- Code generation
- (Code optimization)
- (Linker)
- (Loader)

7

#include directive

- "`#include`" finds the file and insert it inside the current program file
- Different compilers may have different rules for finding the header files
- Example:

```
#include <iostream>
#include <iostream.h>
#include "iostream.h"
```

8

<iostream>

- Streams defined in <iostream>:
  - `std::cout`
  - `std::cin`
  - `std::cerr`
- `std::endl` may be used for flushing the output and moving to the next line.

9

"std" Namespace

- Namespaces are used to organize variables and functions names.

```
#include <iostream>
// Namespace for standard C++ libraries
using namespace std;
// or 'using std::cout;'
int main()
{
  cout << "Hello......." << endl;
  return 0;
}
```

10

Example Client Program

```
#include <iostream>
#include "D:\Source\cpp\DataStructures\stack.h"
using std::cout;
using std::endl;

int main() {
    Stack s = Stack::Stack();
    s.push('+');
    s.push('+');
    s.push('C');
    while (s.noOfElements() > 0)
        cout << s.pop();
    cout << endl;
}
```

Bad practice

11

Header File

```
class Stack
{
private:
    static const int stackSize = 10;
    char stackArray[stackSize];
    int idx;
public:
    Stack();
    bool push(char elem);
    char pop();
    int noOfElements();
};
```

12

Header File

```
#ifndef _STACK_H
#define  _STACK_H
class Stack
{
private:
   static const int stackSize = 10;
   char stackArray[stackSize];
   int idx;
public:
   Stack();
   bool push(char elem);
   char pop();
   int noOfElements();
};
#endif   /* _STACK_H */
```

13

Implementation

```
#include "D:\Source\cpp\DataStructures\stack.h"
Stack::Stack() {  idx = 0; }
bool Stack::push(char elem) {
    bool success = false;
    if (idx < stackSize)
    {
        stackArray[idx] = elem;
        idx++;
        success = true;
    }
    return success;  }
char Stack::pop() {
    char result = '\0';
    if (idx > 0)
    {
        idx--;
        result = stackArray[idx];
    }
    return result;  }
int Stack::noOfElements() { return idx; }
```

14