

CS11600: Introduction to Computer Programming (C++)

Lecture 7

Svetlozar Nestorov
University of Chicago

Outline

- Data encapsulation, ADT, API
- Struct and typedef
- Classes
- Define directives
- Private and public members
- Constructors
- Destructors

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

2

Data Encapsulation

- Abstract data types (ADT)
- Separate *interface* from *implementation*.
- All access to data is through the interface.
- Application program(ing) interface (API)
- Benefits
 - Modularity, portability
 - Data integrity
- Design methods

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

3

Struct

- Create new data type by grouping data members (of other types).
- General form:

```
struct newType {  
    Type1 member1;  
    Type2 member2;  
    ...  
};
```
- Notice ; after definition.

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

4

Struct Examples

```
struct Student {  
    int SID;  
    float gpa;  
    int age;  
};  
  
struct ListNode {  
    int data;  
    ListNode *next;  
};  
  
Student joe;  
Student jane = {11111, 4.0, 19};  
ListNode head;
```

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

5

Working with Structs

- Access data members:
`name.member` (name is a struct)
`name->member` (name is a pointer to a struct)
- Dynamic allocation of struct:
`new newType` (returns a pointer to a struct)
- Copy and assignment
`Student kim = jane;` (copy)
`Student *leo = new Student;`
`*leo = kim;` (assignment)

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

6

Typedef

- Define synonym name (alias) for a type:

```
typedef Type newName;
```

- Examples:

```
typedef int number;
typedef int[20] controls;
typedef double *doublePtr;
typedef ListNode *ListNodePtr;
struct ListNode {
    int data;
    ListNodePtr next;
}
```

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

7

Classes

- Basic building block of C++ programs
 - Equivalent to structs.

- General form:

```
class ClassName {
private:
    ...members...
public:
    ...members...
};
```

Can be used as Type

Internals (inside view)

data and/or functions

Interface (outside view)

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

8

Class Definition

- Often a class is defined in `.h` (or `.hh`) file and implemented in `.C` or `.cc` file.
- Implementation is definition of member functions.
- Start `.h` file with:

```
#ifndef ClassNameH
#define ClassNameH
```

Multiple #includes will
include .h file only once

- End `.h` file with:
- ```
#endif
```

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

9

## Class Implementation

- Outside of class definition, member functions are prefaced by `ClassName::`
- Access members (both data and functions) similar to struct.

`name.member` (name is an object)

`name->member` (name is a pointer to an object)

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

10

## Private and Public Members

- Private* data members and *private* member functions can be invoked only within class implementation, i.e. member function definitions.
- Public* members can be accessed anywhere.

```
class Calc {
public:
 void helper(...);
 int add(...);
};

int Calc::add(...) {
 ...
 helper(...);
};
```

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

11

## Constructors

- Initialize object when created.
  - Allocate memory for private data members if necessary.
- General form:  
`ClassName (...);`
- Member function with the same name as the class and no return type.
- Many constructors are allowed.
  - Argument lists must differ.
- Default constructor: either no arguments or all arguments have default values.

1/20/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

12

## Destructors

- Invoked when object is deleted:
  - Explicit: `delete`
  - Implicit: on function returns
- Should deallocate all dynamically allocated memory within object.
- General form:  
`~ClassName();`
- At most one destructor; no arguments, no return type.

## Linked List Example

- Simple singly linked list
- Class interface
- Class implementation