

CS 31 Discussion 1A, Week 10

Zengwen Yuan (zyuan [at] [cs.ucla.edu](mailto:zyuan@cs.ucla.edu))
Humanities A65, Friday 10:00—11:50 a.m.

Today's focus

- Class
- Final review

Class

- Same as struct, except that its member functions and member variables are private by default.
- Core concept of Object-oriented Programming (OOP).
- Object: an instance of a class
- Encapsulation: member variables are private; use public accessor (getter) and mutator (setter) functions

Class: constructor

- Constructor has a name that matches the class name and without any return type; not even `void`.
- Constructors cannot be called explicitly as if they were regular member functions. They are only executed once, when a new object of that class is created.
- The *default constructor* is the constructor that takes no parameters.
- Overloading constructors — a constructor can also be overloaded with different versions taking different parameters; don't forget the *default constructor*!

Class: destructor

- Destructors fulfill the opposite functionality of *constructors*: they do the necessary cleanup needed by a class when its lifetime ends.
- A destructor is a member function very similar to a default constructor: it takes no arguments and returns nothing, not even `void`.
- It also uses the class name as its own name, but preceded with a tilde sign (`~`).

Class: member variable initialization

- Usually:
 - `Rectangle::Rectangle (int x, int y) { width=x; height=y; }`
- An initializer list is an alternate, concise syntax for constructors. This is done by inserting, before the constructor's body, a colon (:) and a list of initializations for class members.
 - `Rectangle::Rectangle (int x, int y) : width(x) { height=y; }`
 - `Rectangle::Rectangle (int x, int y) : width(x), height(y) { }`

Class: dynamic allocation

- new operator

```
MyClass *p;
p = new MyClass; // default constructor called
```

dynamic variable

- delete operator

```
delete p; // the memory space of p is marked recyclable
```

- Caveat: *dangling pointer*. Dereferencing it is dangerous and leads to undefined behavior. One way to avoid this is to set `p` to `nullptr/NULL` after using `delete`.

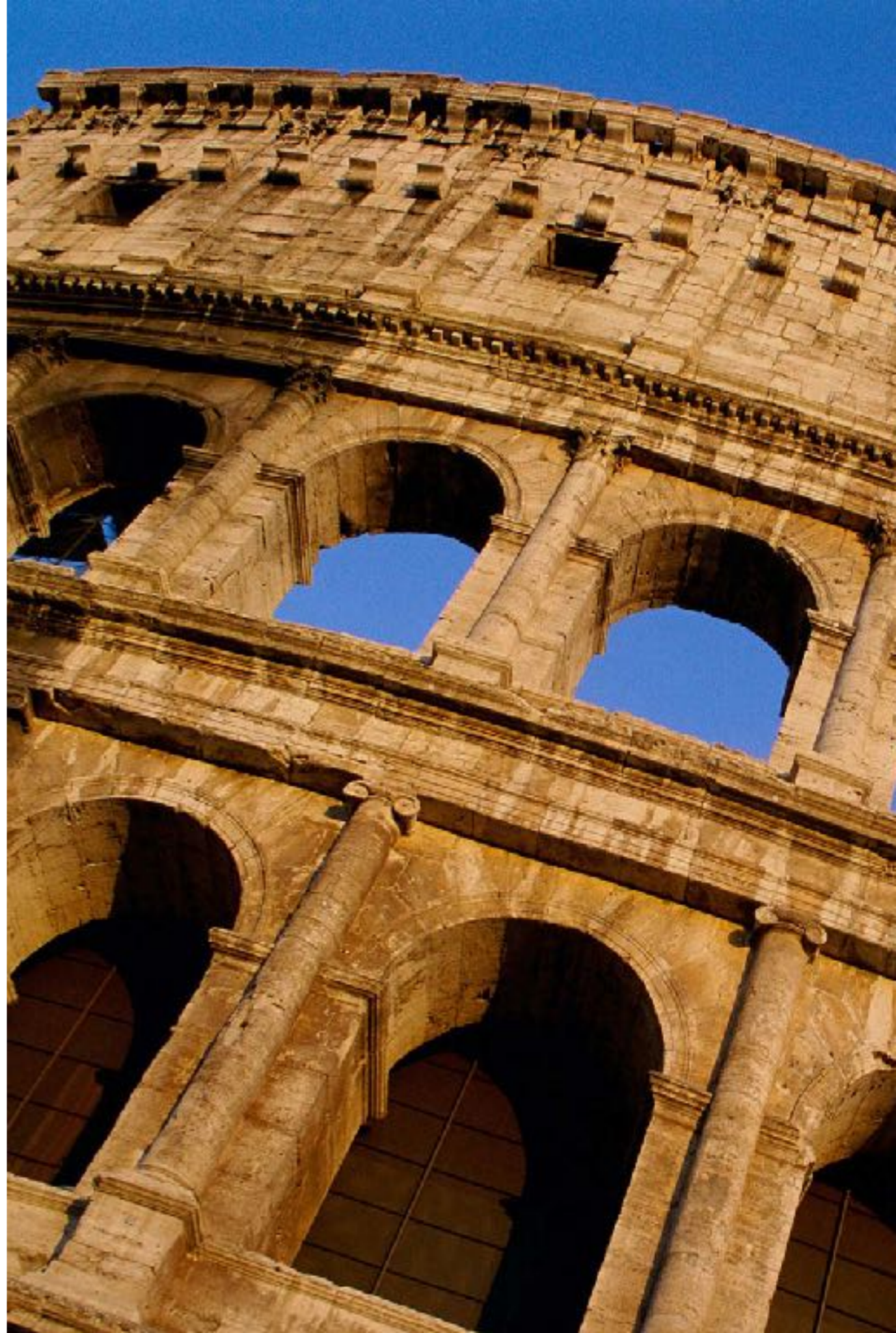
Class: the *this* pointer

- The *this* pointer is a predefined pointer that points to the calling object.
- access member variables even when they are shadowed by local variables.
- pass the current object into a function that takes an argument of its class.

Class: pointers to objects

expression	can be read as
<code>*x</code>	pointed to by x
<code>&x</code>	address of x
<code>x.y</code>	member y of object x
<code>x->y</code>	member y of object pointed to by x
<code>(*x).y</code>	member y of object pointed to by x (equivalent)
<code>x[0]</code>	first object pointed to by x
<code>x[1]</code>	second object pointed to by x
<code>x[n]</code>	(n+1)th object pointed to by x

Final Review



Resources (from UPE)

- Slides: <https://goo.gl/DhAcfN>
- Practice problems: <https://goo.gl/C2vncz>
- Other TA's practice problems and samples

Final Reminder

- The final exam will be closed book, closed notes, except for two 8½"×11" sheets of paper (4 sides). The final is Saturday, December 3,
- Bring a No. 2 pencil to the final. If you're the kind of person who asks questions during an exam, please sit in the front row or in an aisle seat.
- Don't forget the course evaluation on MyUCLA. Let me know what I did right and where I can improve! :)

Good luck!