PIC 10B Discussion
Week 3

Professor: Michael Lindstrom
TA: Thomas Tu
Homework Regrading

• See syllabus for full policy
• Written request by next Monday
• No guarantee of increase or change in grade
  • Rubric used will be exactly the same
  • Grade can go down as I have more time than the grader to catch mistakes
• Regrade is final
Exercise

• Students are pointing at a cat as it runs around the yard
  • If no one is pointing at the cat, it runs away (is deleted)

• Students randomly close/open their eyes
  • When closing eyes, stop pointing at the cat
  • When opening, pick a random student, and point to what they’re pointing to

• Repeat until the cat leaves
  • i.e. `~Cat()` is run

• Note to self: when an exercise feels contrived, it’s probably not worth doing
Exercise

• Object Banner is initialized with a const shared_ptr<string>
  • This represents an external resource
  • Create a weak pointer from it

• Whenever the display() method is called, print the string

• Banner should not prevent the external resource from being deallocated
  • Handle case in which the pointer is null
R-, GL-, PR-, X-, and L- Values

• Two main types: R-values and GL-values

• R-values: “right hand side”
  • x = 3; // 3 is on the right hand side
  • y = x; // x is on the right hand side

• GL-values: “generalized left hand side values”
  • x = 3; // x is on the left hand side
  • arr[0] = 15; // arr[0] is on the left hand side
R-, GL-, PR-, X-, and L-Values

• R-values have two forms:

• PR-values: “pure R-values”
  • Literals: 5, “hello”, 0x4c392a
  • Return types that are not objects/references
    • vec.size(); // returns an int

• X-values: “eXpiring values” – stored temporarily
  • Objects and certain function values
  • Return types that are references to R-values
    • obj.getNameReference(); // returns reference to string obj.name
  • Attributes of returned objects
    • getFirstElement().value
R-, GL-, PR-, X-, and L- Values

- GL-values can be X-values or L-values
- L-values – “left values”
  - Functions or objects
  - References to functions or objects
    - Ball* ballptr;
    - ballptr = &otherBall;
- X-values
  - getFirstElement().m = 5;
Move Semantics and && References

• Why do we have all these types?
  • Wanted more efficiency, eliminate unnecessary copies
  • Operator overloading: your next topic; we’ll come back to this later

• && references
  • Bind to r-values
  • `string && r_ref = “right”;`

• Allows you to distinguish gl-values from r-values
  • `Foo( Ball& gl_ball);`
  • `Foo( Ball&& r_ball);`
  • So you can make certain operations more efficient