Programming Paradigms
Names, Scopes, and Bindings (Part 4)

Prof. Dr. Michael Pradel
Software Lab, University of Stuttgart
Summer 2020
Overview

- Object lifetime and storage management
- Scopes
- Aliasing and overloading
- Binding of referencing environments
Aliasing and Overloading

**Aliasing**: Two more more names refer to the same object

```
name1 → object1
name2 → object1
```

**Overloading**: A name refers to two more objects

```
name1 → object1
name1 → object2
```
#include <stdio.h>

void half(double& a)
{ // argument passed by reference
    a = a / 2;
}

int main( int argc, const char* argv[] )
{
    double n = 5.0;
    double *p = &n; // pointer to value stored in n
    
    half(n);
    half(*p);
    
    printf("%f\n", n);
}
#include <stdio.h>

void half(double& a) {
   // argument passed by reference
   a = a / 2;
}

int main(int argc, const char* argv[]) {
   double n = 5.0;
   double *p = &n; // pointer to value stored in n
   half(n);
   half(*p);
   printf("%f\n", n);
}

Result: 1.250000
Overloading: Example

class Overloading{
    void foo() {}
    void foo(int n) {}
    void foo(String s) {}

    public static void main(String[] args) {
        Overloading o = new Overloading();
        o.foo(...);
    }
}
Overloading: Example

Three methods, all with name “foo”

Resolution of name depends on arguments