• **What is a type?**
  – The notion varies from language to language

• **Consensus**
  – A set of values
  – A set of operations on those values

• **Classes are one instantiation of the modern notion of type**
Consider the assembly language fragment

```
add $r1, $r2, $r3
```

What are the types of $r1, $r2, $r3?
• Certain operations are legal for values of each type

  – It doesn’t make sense to add a function pointer and an integer in C

  – It does make sense to add two integers

  – But both have the same assembly language implementation!
• A language’s type system specifies which operations are valid for which types

• The goal of type checking is to ensure that operations are used with the correct types
  – Enforces intended interpretation of values, because nothing else will!
• Three kinds of languages:

  – *Statically typed:* All or almost all checking of types is done as part of compilation (C, Java, Cool)

  – *Dynamically typed:* Almost all checking of types is done as part of program execution (Scheme)

  – *Untyped:* No type checking (machine code)
Competing views on static vs. dynamic typing

Static typing proponents say:
- Static checking catches many programming errors at compile time
- Avoids overhead of runtime type checks

Dynamic typing proponents say:
- Static type systems are restrictive
- Rapid prototyping difficult within a static type system
Types

• A lot of code is written in statically typed languages with an “escape” mechanism
  – Unsafe casts in C, Java

• People retrofit static typing to dynamically typed languages
  – For optimization, debugging

• It’s debatable whether either compromise represents the best or worst of both worlds
• The types in Cool are:
  – Class Names
  – SELF_TYPE

• The user declares types for identifiers

• The compiler infers types for expressions
  – Infers a type for every expression
Types

• *Type Checking* is the process of verifying fully typed programs

• *Type Inference* is the process of filling in missing type information

• The two are different, but the terms are often used interchangeably