CS 334 Lecture 8
\[ e \colon= x \mid y \mid \sin \pi e \mid \cos \pi e \mid e^x \mid e^e \mid \frac{e+e}{2} \]

\[ f(x, y) = \frac{x+y}{2} \quad f(x, y) = \sin \pi x \]

If \( x, y \in [-1, 1] \), then \( f(x, y) = [-1, 1] \)

\[
\begin{array}{cccc}
(-1,-1) & (-1,1) \\
(1,-1) & (1,1)
\end{array}
\]

\[ f(x, y) \begin{cases} 
-1 \to \text{black} \\
0 \to \text{neutral} \\
1 \to \text{white}
\end{cases} \]
Imperative Features in ML

```c
void f() {
    int x, y;
    X = y + 3;
}
```

3 refers to contents of memory location y (R-Value)
refers to location for variable x in memory (L-Value)
\[ X \quad \text{Value} \quad Y \quad \text{Value} \quad Z \quad X \quad \text{Value} \quad R \quad \text{Value}. \]

\[
\text{ML}

\cdot \text{LV values & RV values are distinct.}
\cdot \text{\texttt{val a = 3}; can't be changed.}
\]
Reference Cells

counter

```
int ref: pointer to a memory location holding an int.
val c = ref v;
!c
C := V;
```

1. Type system distinguishes mutable vs. immutable.
2. Easy to reason about side effects. Side effects are explicit in code.
Type - set of computational entities that share a common property.

int: \{ \ldots, 2, -1, 0, 1, 2, \ldots \}

String: \{ "Wally", "", "Steve" \}

int \rightarrow \text{bool}: \{ \text{all fins from int to bool} \}

\text{non-ex/}: \{ 1, \text{false}, \lambda x. 3 \}
Types in PLs
- dictate how data is represented & manipulated.

3 Primary Uses

0. Documentation.
1. Prevent Type Errors.

Ex: 3+ true "Duncan" + "Wally"
- value/functional is used inconsistently with what it represents.
3) Compiler Optimization

```scala
type Student = { name: String, year: int };

Student (name = "Betsy", year = 2021);
```

```scala
type Prof = { course: String list, name: String };

Prof (name = "Steve", course = [...]);
```

```scala
fun getName (x) = #name (x);
fun getName (x: Prof) = #name (x) + offset in memory.
```
Ensuring Type Safety

-no program is allowed to perform a type error.

Dynamically - Lisp, Python, Javascript,

Cons: [0 | car | cdr]

Atom: [1 | 1 | Data]

type of

Atom: 0 - int
1 - symbol
2 - number
3 - function

(car x) - is x a cons cell?
(+ a z) - is + a function.
-is a an int?
-is b an int?
Statically - (compile time)
- ML, Java

\[ f(x) \text{ - compiler checks that} \]
\[ f : A \rightarrow B \]
\[ \text{and } x : A \]
\[ e + 2 \text{ - does expression } E \]
\[ \text{produce an } \texttt{int} ? \]

Coverage
\[ (\text{if } \ldots ( + 5 3 ) ( + \text{ mod } 3 )) \]
All Programs

Dynamically Error-Free

Statically Checkable

Static checking is conservative

Hybrid Checking
- array bounds
- null ptrs
- C&L
- Cus13.