Tour of common optimizations
Simple example

```plaintext
foo(z) {
    x := 3 + 6;

    y := x - 5

    return z * y
}
```
foo(z) {
    x := 3 + 6;
    y := x - 5
    return z * y
}
Another example

\[ x := a + b; \]

\[ \ldots \]

\[ y := a + b; \]

\[ z + (a + b) \]

\[ z + x \]
Another example

\[ x := a + b; \]
\[ \ldots \] \[ \} \text{ only if } x, a, b \text{ not modified!} \]
\[ y := a + b; \] \[ x \]

\[ \]
Another example

\[ a := 0 \]

\[
\text{if} \ (\ldots) \ {\text{\{}} \\
\quad a := \ b; \ \ t := x \\
\} \ \text{else} \ {\text{\{}} \\
\quad t := a + b \} \\
\]

\[
\ldots \ \ t
\]

\[ y := a + b; \]
Another example

```plaintext
if (...) {
    x := a + b; \( t \)
} else {
    t := a + b
}
...

y := a + b; \( t \)
```

Partial Redundancy Elimination (PRE)
Another example

\[
x := y \\
\ldots \\
z := z + y
\]
Another example

\[
x := y \\
\ldots \\
z := z + x \quad \text{\{ x, y not modified \}} \quad \text{\copy prop}
\]
Another example

\[
\begin{align*}
x & := y \\
\ldots \\
z & := z + y
\end{align*}
\]

What if we run CSE now?

\[
\begin{align*}
x & := E \\
\ldots \\
(E) & \rightarrow x
\end{align*}
\]
Another example

\[ x := y \]
\[ \ldots \]
\[ z := z + y \times \]

What if we run CSE now?
Another example

\[ x := y**z \]

\[ \ldots \]

\[ x := \ldots \]
Another example

• Often used as a clean-up pass

\[ x := y^{**}z \]
\[ \ldots \]  \[ \} \text{if } x \text{ is not used} \]
\[ x := \ldots \]  \[ \text{dead assignment elim} \]
\[ \text{(unused assignment elim)} \]

\[ x := y \]
\[ z := z + x \]  \[ \rightarrow \]  \[ x := y \]
\[ z := z + y \]  \[ \rightarrow \]  \[ z := z + y \]
Another example

if (false) {
    ...
}

}
Another example

```java
if (false) {
    ...
}
```

dead code elim (unreachable code elim)

Another common clean up opt
Another example

• In Java:

```java
a = new int [10];
for (index = 0; index < 10; index ++) {
    a[index] = 100;
}
```
Another example

• In “lowered” Java:

```java
a = new int[10];
for (index = 0; index < 10; index++) {
    if (index < 0 || index >= a.length()) {
        throw OutOfBoundsException;
    }
    a[index] = 0;
}
```
Another example

• In “lowered” Java:

```java
a = new int [10];  // 1
for (index = 0; index < 10; index ++)
     if (index < 0 || index >= a.length()) {
         throw OutOfBoundsException;
     }
     a[index] = 0;
```

- Branch folding
- unreachable code elim
- Kinda like CP
- If we assume stmt 0 acts like a.length := 10
Another example

\[ p := \&x; \]
\[ *p := 5 \]
\[ y := x + 1; \]

5
Another example

\[
p := \&x; \\
*p := 5 \\
y := x + 1; \quad 6
\]

\[
x := 5; \\
*p := 3 \\
y := x + 1; \quad ???
\]
Another example

\[
\begin{align*}
\text{for } j := 1 \text{ to } N & \quad k := b[j] \\
\text{for } i := 1 \text{ to } M & \\
\quad a[i] := a[i] + b[j]k
\end{align*}
\]

\[
\begin{align*}
\text{for } (i = 0; i < 10; i++) & \\
\quad a(c[i]) & = 5
\end{align*}
\]
Another example

for j := 1 to N
    for i := 1 to M
        a[i] := a[i] + b[j]

\[ t := b[j] \]

Loop invariant
Code motion
Another example

\[
\text{area}(h, w) \{ \text{ return } h \times w \} \\
\]

\[
\begin{align*}
h & := \ldots; \\
w & := 4; \\
a & := \text{area}(h, w) \\
\end{align*}
\]

\[
\begin{align*}
h & \times w \times 4 \\
h & \ll 2
\end{align*}
\]
Another example

```java
area(h,w) { return h * w }

h := ...;
w := 4;
a := area(h,w)
```

Many "silly" opts become important after in-lining
Optimization themes

• Don’t compute if you don’t have to
  – unused assignment elimination

• Compute at compile-time if possible
  – constant folding, loop unrolling, inlining

• Compute it as few times as possible
  – CSE, PRE, PDE, loop invariant code motion

• Compute it as cheaply as possible
  – strength reduction

• Enable other optimizations
  – constant and copy prop, pointer analysis

• Compute it with as little code space as possible
  – unreachable code elimination