Eraser Review Notes

Key Idea: Eraser does not necessarily detect races. Eraser detects dangerous locking practices (often leading to races)

Simple Data Race:

```
v = 0
def f():
    v = v + 1
```

when f is run by multiple threads, v can end up with an unexpected value (such as 1)

when analyzing concurrency, useful to split program into reads and writes

ex:

```
def F():
    x ← v    (read v)
    v ← x+1  (write v)
```

Now the race is obvious

T_1:	T_2:
read v	
	read v
write v=1	
	write v=1

Eraser will warn us of this problem because variable v is not protected by any locks!

Correct Locking:

```
def f():
  Lock(l1)
  v = v+1
  Unlock(l1)
```

Question: Eraser slows down a program, which can prevent races from happening. Why is this okay?

Answer: As long as two Threads access a shared variable (even at completely different times) without a lock, Eraser will detect it.

ex.

T_1:	T_2:
v = v+1	
⋮ (time)	⋮
	v = v+1

Eraser will still notice that v is accessed without a lock, even though no race actually happens

Eraser False Positives:

read-only variables:

T_1:	T_2:
x = v	[redacted]
	y = v

Eraser solves this by waiting until a variable is read-write shared before issuing warnings

Initialization:

T_1:	T_2:
x = new object()	.
x.y = "foo"	.
init(x_lock)	.
	// all accesses to x now happen // with lock held

Eraser solves this by waiting until another thread accesses the variable before checking its locks.

Private Lock Implementations:

Eraser adds its routines by checking a program for calls to the pthread library (a common C-Library)

If a program uses other locking schemes, Eraser won't detect this.

Benign Races:

ex:

```
if (x == null) {
    lock(l)
    if (x == null)
        x = new object()
    unlock(l)
}
```

This code is correct.

Why would a programmer do this?

performance: Locking is expensive

Eraser False Negatives:

```
if (today == Xmas)
    V=V+1
else {
    Lock(l)
    V=V+1
    Unlock(l)
}
```

Eraser will only detect bad locking practice on Xmas!

Code needs to execute for Eraser to detect problems.

2010 Quiz 1 problem 5:

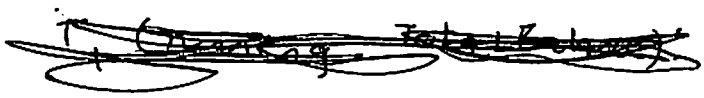
A. Each account is a shared variable as is protected by its own lock when accessed. Eraser will not issue a warning.

B. Now Eraser will issue a warning b/c TotalBalance is reading an account variable without a lock.

C. True. Transfer maintains the same total amount between accounts when run to completion.

D. False

ex:



T_1 (running Total Balance):

total = total + change(a, 0)

T_2:

Transfer(a, b, 100)

total = total + change(b, 0)

total will end up with a balance too large.

E. False ex:

T_1:

change(a, -100)

T_2:

TotalBalance()

change(b, 100)

This problem is a good example of another Eraser false negative.

Eraser issued no warnings, but we still had a bug!