Dealing with shared memory

Thread1

```
amount = 30;
if(account.getBalance() > amount){
    account.withdraw(amount);
}
```

Thread2

```
amount = 20;
if(account.getBalance() > amount){
    account.withdraw(amount);
}
```

The account has 40 dollars in it.
Who gets the money?

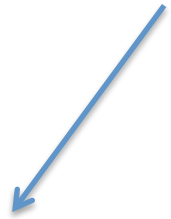
Thread 1 manages to execute its code first.

Thread1

```
1  amount = 30;
2  if(account.getBalance() > amount){
3        account.withdraw(amount);
4  }
   //account now has 10 dollars
```

Thread2

Unable to withdraw!



```
5  amount = 20;
6  if(account.getBalance() > amount){
7        account.withdraw(amount);
8  }
```

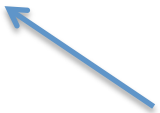
Or maybe Thread 2 executes its code first.

Thread1

```
5 amount = 30;
6 if(account.getBalance() > amount){
7       account.withdraw(amount);
8 }
```

Thread2

```
1 amount = 20;
2 if(account.getBalance() > amount){
3       account.withdraw(amount);
4 }
  //account now has 20 dollars
```



Unable to withdraw!

Thread1

```
1  amount = 30;
2  if(account.getBalance() > amount){
       //balance is still 40!
7      account.withdraw(amount);
8  }
```

Thread2

```
3  amount = 20;
4  if(account.getBalance() > amount){
5      account.withdraw(amount);
6  }
```

Both successfully withdraw! The account now has -10 dollars.

- Every possible interweaving of code can occur at some point.
- Bugs might only happen once in 1000 tries or more.

Racing Cars

Car 1

```
int count = 0;
for(int i = 0; i < finishers.length; i++) {
        if (finishers[i]) count++;
}

finishers[myIndex] = true;
```

Car 2

```
int count = 0;
for(int i = 0; i < finishers.length; i++) {
        if (finishers[i]) count++;
}

finishers[myIndex] = true;
```

Count is zero for both.

Car 1

```
finishers[myIndex] = true;


int count = 0;
for(int i = 0; i < finishers.length; i++) {
        if (finishers[i] && i!=myIndex)
                count++;
}
```

Car 2

```
finishers[myIndex] = true;


int count = 0;
for(int i = 0; i < finishers.length; i++) {
        if (finishers[i] && i!=myIndex )
                count++;
}
```

Count is 1 for both. No one wins!

Can we fix it?

Yes. Take CS110!