Parallel processing & sockets

YO DAWG, I HEARD YOU LIKED PROCESSORS...

...SO WE PUT PROCESSORS IN YOUR PROCESSOR SO YOU CAN PROCESS WHILE YOU PROCESS!
Terminology

CPU = area of computer that does thinking
Core = processor = a thinking unit
Program = code = instructions on what to do
Thread = parallel process = an independent part of the program/code
Program = string, thread = 1 part of that
Review: CPUs

In the 2000s, computing too a major turn: multi-core processors (CPUs)
Review: CPUs
Review: CPUs
Review: CPUs

The major reason is due to heat/energy density.
Review: CPUs
This trend will almost surely not reverse.

There will be new major advancements in computing eventually (quantum computing?)

But “cloud computing”, which has programs that “run” across multiple computers are going nowhere anytime soon.
Parallel: how

So far our computer programs have run through code one line at a time.

To get multiple parts running at the same time, you must create a new thread and give it a function to start running.

To do this we extend the Thread class:

```java
public class ParallelProcessing extends Thread {
```
Parallel: how

Then we need to override the run() method and create one:

```java
Thread counter = new Thread(new ParallelProcessing());
counter.start();

@Override
public void run()
{
    // do something!
}
```

(See: ThreadUpdate.java)
Parallel: basics

The major drawback of distributed computing (within a single computer or between) is **resource synchronization** (i.e. sharing info)

This causes two types of large problems:
1. Conflicts when multiple threads want to use the same resource
2. Logic errors due to parts of the program having different information
1. Resource conflict

Siblings anyone?

EVERY SHOWER STALL IN THE BATHROOM OCCUPIED?

BACK TO BED IT IS
1. Resource conflict

Public bathroom?

All your programs so far have had 1 restroom, but some parts of your program could be sped up by making 2 lines (as long as no issues).
1. Resource conflict

We will actually learn how to cause minor resource conflicts to ensure no logic errors.

This is similar to a cost of calling your forgetful relative to remind them of something.

This only needs to be done for the important matters that involve both of you (e.g. when the family get-together is happening).
2. Different information

If you and another person try to do something together, but not coordinated... disaster

![Comic strip with dialogue and characters discussing a confusing situation.]

It is really confusing!!!

Four

No Three
2. Different information

Each part of the computer has its own local set of information, much like separate people.

Suppose we handed out tally counters and told two people to count the amount of people.
2. Different information

However, two people could easily tally the number entering this room...

Simply stand one by each door and add them.

Our goal is to design programs that have these two separate parts that can be done simultaneously (which tries to avoid sharing parts).
Parallel: how

To avoid having threads fight over memory, we can designate it so only one thread can be in a method at a time.

This is done by attaching `synchronized` to it:

```java
synchronized public static void increment()
{
    count++;
}
```

(see: SychronizedThreading.java)
Parallel: how

However, as you may have noticed: using threads is much slower than the normal loop.

Threads take a fair amount of resources to manage, so they need to do substantial parts of the program to be worth while.

However, the more synchronized it is... the less efficient the threads are.
A major issue when using parallel processing is to be able to design your program in as many (semi-)independent parts as possible. Each (semi-)independent part can be run on its own thread and easily double or triple the speed of your program. This puts a greater emphasis on designing your program well to take advantage of threads.
Sockets

ARE YOU COMING TO BED?

I CAN’T. THIS IS IMPORTANT.

WHAT?

SOMEONE IS WRONG ON THE INTERNET.
Sockets

Communicating over the internet is actually very similar to using files...

We need to make an object/variable to represent our connections, then:

1. Open socket
2. Send data using socket variable
3. Close socket
Since computers run multiple programs at the same time, we need to specify which program is supposed to receive our data.

This is done using a port number, which is a semi-arbitrary number.

You can think of this like the name on a snail mail (who in the house it belongs to) (See: Client.java and Server.java)
Done

TAKE BASIC CODING COURSE

EXPERT PROGRAMMER