2: Let's Write a Program: Prime Numbers

java Primes U to print prime numbers through U.
va Primes 101
3 5 7 11 13 17 19 23 29
37 41 43 47 53 59 61 67 71
79 83 89 97 101

prime number is an integer greater than 1 that has no than itself other than 1.
> 1 is prime iff gcd$(p, x) = 1$ for all $0 < x < p$.)

$N/k \geq \sqrt{N}$, for $N, k > 0$.
N then N/k divides N.

ential divisors up to and including the square root.

Testing for Primes

```
 boolean isPrime(int x) {

lse;

sDivisible(x, 2);  // "!" means "not"


{ is divisible by any positive number >=K and < X,
 1. */
 boolean isDivisible(int x, int k) {
          // a "guard"
lse;
% k == 0)  // "%" means "remainder"
ue;
(k < x && x % k != 0)
)ivisible(x, k+1);
```


Iteration

is *tail recursive*, and so creates an *iterative process*.

Algol family" production languages have special syntax
. Four equivalent versions of `isDivisible`:

```
se;
 k == 0)
e;

ivisible(x, k+1);
```

```
while (k < x) { // !(k >= x)
  if (x % k == 0)
    return true;
  k = k+1;
  // or k += 1, or (yuch) k++
}
return false;
```

```
x) {
 == 0)
rue;

;
```

```
for (int k1 = k; k1 < x; k1 += 1) {
  if (x % k1 == 0)
    return true;
}
return false;
```


Administrivia

sure you have obtained a Unix account. If you have y (i.e., since today) signed up for concurrent enrollment us your name, email, and SID. After we have a chance , you will be able to use WebAcct, as Lab #1 specifies.

e Wednesday (end of Wednesday at midnight). Usually, Friday midnight of the week they occur. It is especially set up your central reppository.

e not to take this course after all, please tell CalCentral at we can adjust the waiting list accordingly.

up; due next Friday at midnight. You get credit for any but we suggest you give the problems a serious try.

Plan

```
rimes {
l primes up to ARGS[0] (interpreted as an
, 10 to a line. */
c void main(String[] args) {
s(Integer.parseInt(args[0]));


l primes up to and including LIMIT, 10 to
*/
ic void printPrimes(int limit) {
very integer, x,  between 2 and LIMIT, print it if
e(x), 10 to a line. }*/


 X is prime */
ic boolean isPrime(int x) {
{ X is prime )*/;
```


Thinking Recursively

check `isDivisible(13,2)` by *tracing one level*.

```
s divisible by
>=K and < X,
 */
oolean isDivisible...

;
 == 0)


isible(x, k+1);
```

nents aid understanding.
nt!

- Call assigns `x=13, k=2`
- Body has form '`if (k >= x)` S_1 `else` S_2'.
- Since $2 < 13$, we evaluate the first `else`.
- Check if $13 \bmod 2 = 0$; it's not.
- Left with `isDivisible(13,3)`.
- Rather than tracing it, instead use the *comment:*
- Since 13 is *not* divisible by any integer in the range 3..12 (and $3 > 1$), `isDivisible(13,3)` must be *false*, and we're done!
- Sounds like that last step begs the question. Why doesn't it?

Cautionary Aside: Floating Point

lide, we had

```
 = (int) Math.round(Math.sqrt(x));
k1 = k; k1 <= limit; k1 += 1) {
```

at this would check all values of k1 up to and including
oot of x.

ng-point operations yield *approximations* to the corre-
hematical operations, you might ask the following about
round(Math.sqrt(x)):

ys at least $\lfloor\sqrt{x}\rfloor$, where $\lfloor z\rfloor$ is the largest integer $\leq z$?
e might miss testing $\sqrt{x}$ when x is a perfect square.)

ıs, the answer is "yes" for IEEE floating-point square

nple of the sort of detail that must be checked in edge

Using Facts about Primes

used the Useful Facts from an earlier slide. Only have
divisors up to the square root.

ıent the iterative version of isDivisible:

```
f X is divisible by some number >=K and < X,
hat K > 1, and that X is not divisible by
ber >1 and <K. */
tic boolean isDivisible(int x, int k) {
 = (int) Math.round(Math.sqrt(x));
k1 = k; k1 <= limit; k1 += 1) {
 k1 == 0)
n true;

lse;
```

litional (blue) condition in the comment?

nal Task: printPrimes (Simplified)

```
primes up to and including LIMIT. */
 void printPrimes(int limit) {
```


Simplified printPrimes Solution

```
primes up to and including LIMIT. */
 void printPrimes(int limit) {
 = 2; p <= limit; p += 1) {
sPrime(p)) {
ystem.out.print(p + " ");


.println();
```


printPrimes (full version)

```
primes up to and including LIMIT, 10 to
/
 void printPrimes(int limit) {


 = 2; p <= limit; p += 1) {
sPrime(p)) {
ystem.out.print(p + " ");
 += 1;
 (np % 10 == 0)
  System.out.println();


10 != 0)
n.out.println();
```