Welcome to CS61B!

he rather extensive information on sections, Covid-19 ons, labs, initial assignments, and the presemester surall 2021 CS61B Piazza site.

oday. In (or preferably before) lab this week, get a ccount from `https://inst.eecs.berkeley.edu/webacct`.

n remotely to one of the instructional servers eley.edu, where X is ashby.cs, derby.cs, cedar.cs, cory.eecs,

homepage (https://inst.eecs.berkeley.edu/ cs61b/fa21) l distribution site for assignments, lecture slides, course uch else.

l be recorded and screencast. The recordings should able in the bCourses Media Gallery sometime after the

Crowding

se not to take this course please drop it as soon as the benefit of others (the add/drop deadline is 18 -6 September if you wish to avoid a fee).

Stanley will not hold us all, which is why there are both nline lectures. Lecture seating is on a first-come-first-. Definitely not ideal, but we hope that after the first hose of you who prefer in-person lectures will be able

Texts

o readers currently on-line (see the website).

without printed versions, but might want to print out tions for exams (since we don't allow computers in tests).

r first part of the course only) is *Head First Java*. It's but has the necessary material.

Course Organization I

illustrate.

ortant: exercise of programming principles as well as ty details go there. Generally we will give you homework ing them.

important, but it's reasonably easy to get full credit: see fit and *turn it in!* You should get points for just reasonable effort into it.

ojects are *really* important! Expect to learn a lot. Projects efforts (that's for later courses).

Course Organization II

is part of the course. Programming takes place in a *environment:*

diting, debugging, compilation, archiving versions.

, I keep it simple: Emacs + gjdb + make + git, (docin one of the readers and on-line). But we'll look at lab.

allenging: better to stay on top than to cram.

Projects, 50%; HW, 10%

ell us!

Pandemic Considerations

's responsibility to look out each other.

er, in particular, this means adhering to certain incontices mandated by the University.

e wearing masks indoors, as well as staying home when

ve the mask mandate; if anyone refuses, I can and will simply end the day's lecture, and you'll all have to rely e slides for the material.

Academic Dishonesty

cidence of academic dishonesty seems to have increased
rs.

t, this is our fault: the mimimum GPA threshold policy
ors puts people under a lot of stress,

s, we can't afford to tolerate cheating. The Course Info
ourse homepage contains our policy on cheating and the
impose; please read them.

p with the course and starting assignments early, you
ny perceived need to cheat.

ourse is not curved, so you are not disadvantaged by
's dishonesty.

Programming, not Java

rn *programming,* not Java (or Unix, or Windows, or...)

principles span many languages

connections.

+y *vs.* (+ x y)) is superficial.

hon, and Scheme have a lot in common.

u use GUIs, text interfaces, or embedded systems, im-
s are the same.

For next time

Chapter 1 of *Head First Java*, plus §1.1–1.9 of the on-line
Reference, available on the class website.

erview of most of Java's features.

ooking at examples on Friday.

mber the questions that come up when you read some-
ign:

s? We might have made a mistake.

to ask at the start of lectures, by email, or by Piazza.

Acronyms of Wisdom

DBC

RTFM

uick Tour through the First Program

ould write

```
l first program
o, world")
```

```
l first program.
 N. Hilfinger */
Hello {
 greeting. ARGS is ignored. */
atic void main(String[] args) {
em.out.println("Hello, world!");
```

Commentary

```
l first program.
 N. Hilfinger */
Hello {
 greeting.  ARGS is ignored.  */
atic void main(String[] args) {
em.out.println("Hello, world!");
```

nts can either start with '//' and go to the end of the
n Python), or they can extend over any number of lines,
y '/*' and '*/'.

he '//' comments, except for things that are supposed
ed, and our style checks will flag them.

multiline kind of comment includes those that start with
re called *documentation comments* or *doc comments*.

on comments are just comments, having no effect, but
s interpret them as providing documentation for the
ollow them. They're generally a good idea and our style
re them.

Classes

```
l first program.
 N. Hilfinger */
ello {
greeting. ARGS is ignored. */
tic void main(String[] args) {
m.out.println("Hello, world!");
```

on and variable in Java is contained in some *class*.

ke Python's classes, but with (of course) numerous dif-
detail.

n turn, belong to some *package*. The `Hello` class belongs
mous package.

ned packages later,

Methods (Functions)

```
l first program.
 N. Hilfinger */
ello {
greeting. ARGS is ignored. */
tic void main(String[] args) {
m.out.println("Hello, world!");
```

aders in Java contain more information than those in
specify the *types* of values *returned* by the function
parameters to the functions.

oid has no possible values; the *main* function here re-
. The type `String` is like Python's `str`. The trailing '`[]`'
of. Arrays are like Python lists, except that their size
created.

takes a list of strings and returns nothing.

med "main" and defined like the example about are spe-
e what get called when one runs a Java program (in
ain function is essentially anonymous).

Selection

```
l first program.
 N. Hilfinger */
ello {
greeting. ARGS is ignored. */
tic void main(String[] args) {
.out.println("Hello, world!");
```

$\mathcal{E}.N$ means "the thing named N that is in or that applies
identified (or computed) by $\mathcal{E}$."

.out" means "the variable named '`out`' that is found in
ed '`System`'."

stem.out.println" means "the method named '`println`'
o the object referenced by the value of variable '`System.out`'."

Access

```
l first program.
 N. Hilfinger */
ello {
greeting. ARGS is ignored. */
tic void main(String[] args) {
.out.println("Hello, world!");
```

ed entity in Java has *access permissions* indicating what
e may mention it.

, *public* classes, methods, and variables may be referred
else in the program.

es refer to them as *exported* from their class (for
arialbles) or package (for classes).

Access

```
l first program.
 N. Hilfinger */
ello {
greeting. ARGS is ignored. */
tic void main(String[] args) {
.out.println("Hello, world!");
```

ods and variables are "one-of" things.

hod is just like an ordinary Python function (outside of
a function in a Python class that is annotated `@staticmethod`.

iable is like a Python variable defined outside of any
iable selected from a class, as opposed to from a class

les are local variables (in functions) or instance vari-
ses), and these are as in Python.