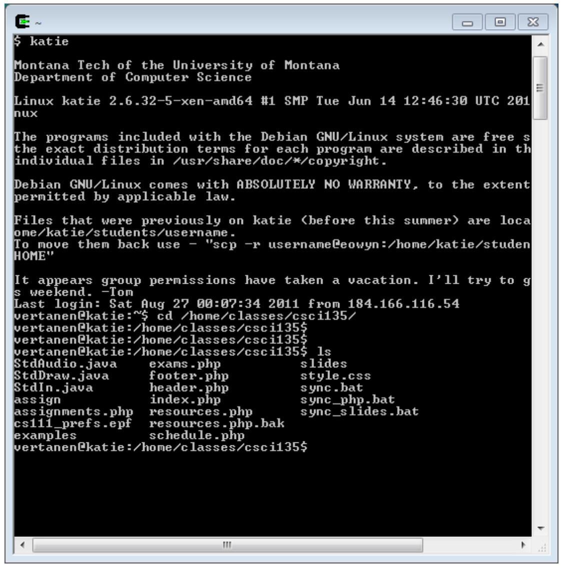
Survivor: CSCI 135

Interfacing with your computer

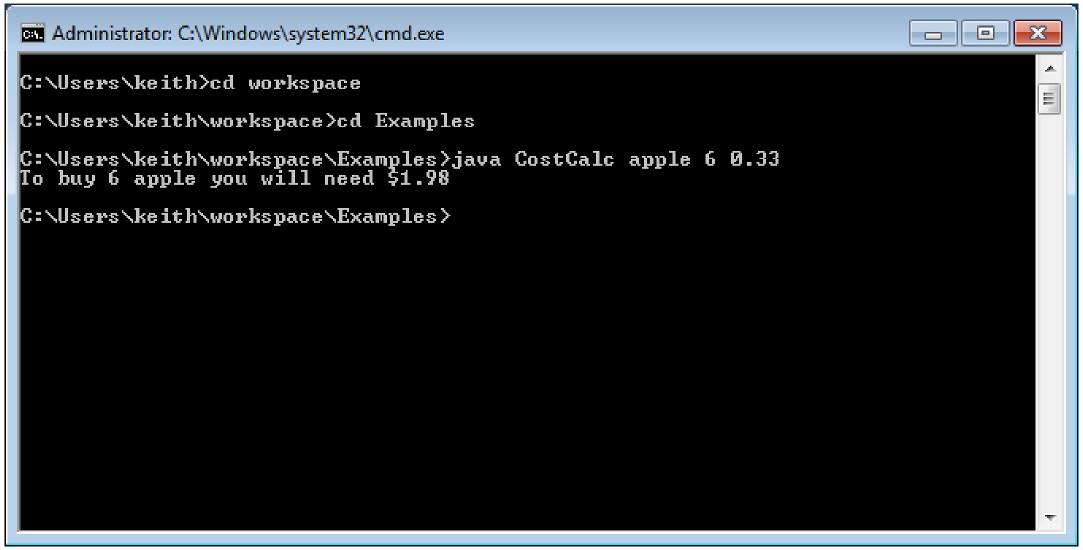
- GUI (graphical user interfaces)
  - Today: predominant interaction method
  - Windows, buttons, mouse
  - Advantages
    - Easier for novices
    - No commands to remember
    - Rich input and output capabilities

Interfacing with your computer

- Command line interface (CLI)
  - Originally the only option
  - Input by typing commands
  - Advantages:
    - Can be faster for experts than a GUI
    - Easier to automate tasks
    - Easier to hook programs together

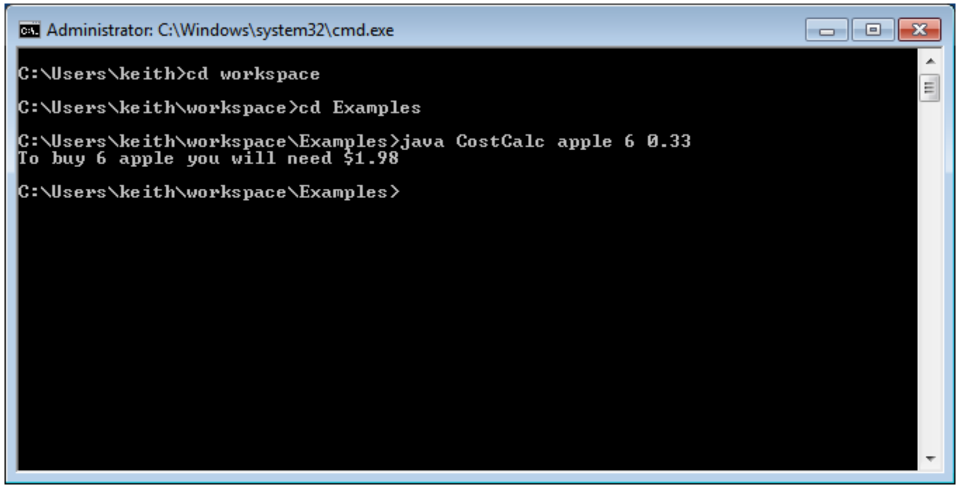


```
$ katie

Montana Tech of the University of Montana
Department of Computer Science

Linux katie 2.6.32-5-xen-amd64 #1 SMP Tue Jun 14 12:46:30 UTC 201
nux

The programs included with the Debian GNU/Linux system are free s
the exact distribution terms for each program are described in th
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

Files that were previously on katie (before this summer) are loca
ome/katie/students/username.
To move them back use - "scp -r username@eowyn:/home/katie/studen
HOME"

It appears group permissions have taken a vacation. I'll try to g
s weekend. -Tom
Last login: Sat Aug 27 00:07:34 2011 from 184.166.116.54
vertanen@katie:~$ cd /home/classes/csci135/
vertanen@katie:/home/classes/csci135$
vertanen@katie:/home/classes/csci135$
vertanen@katie:/home/classes/csci135$ ls
StdAudio.java     exams.php             slides
StdDraw.java      footer.php            style.css
StdIn.java        header.php            sync.bat
assign            index.php             sync_php.bat
assignments.php   resources.php         sync_slides.bat
cs111_prefs.epf   resources.php.bak
examples          schedule.php
vertanen@katie:/home/classes/csci135$
```



```
C:\Users\keith>cd workspace

C:\Users\keith\workspace>cd Examples

C:\Users\keith\workspace\Examples>java CostCalc apple 6 0.33
To buy 6 apple you will need $1.98

C:\Users\keith\workspace\Examples>
```

Starting a command shell



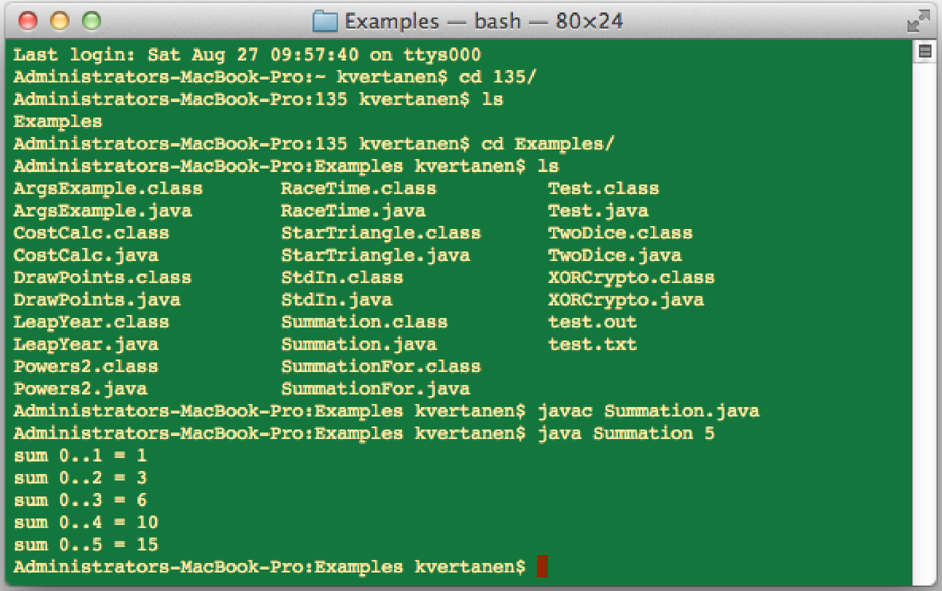
Administrator: C:\Windows\system32\cmd.exe

```
C:\Users\keith>cd workspace

C:\Users\keith\workspace>cd Examples

C:\Users\keith\workspace\Examples>java CostCalc apple 6 0.33
To buy 6 apple you will need $1.98

C:\Users\keith\workspace\Examples>
```

Windows 7

Start → type "cmd"

All Programs → Accessories → Command Prompt



Examples — bash — 80×24

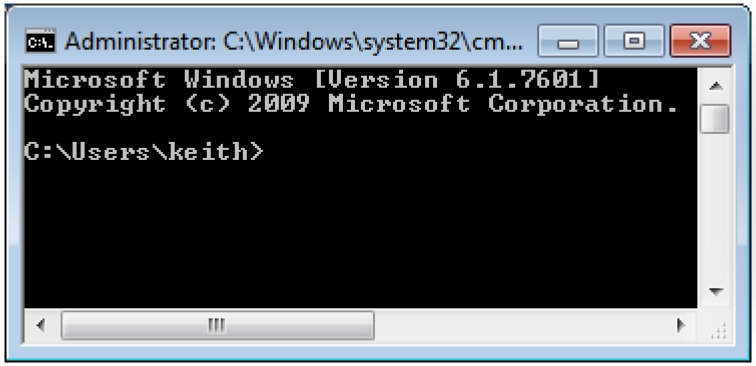
```
Last login: Sat Aug 27 09:57:40 on ttys000
Administrators-MacBook-Pro:~ kvertanen$ cd 135/
Administrators-MacBook-Pro:135 kvertanen$ ls
Examples
Administrators-MacBook-Pro:135 kvertanen$ cd Examples/
Administrators-MacBook-Pro:Examples kvertanen$ ls
ArgsExample.class       RaceTime.class          Test.class
ArgsExample.java        RaceTime.java           Test.java
CostCalc.class          StarTriangle.class      TwoDice.class
CostCalc.java           StarTriangle.java       TwoDice.java
DrawPoints.class        StdIn.class             XORCrypto.class
DrawPoints.java         StdIn.java              XORCrypto.java
LeapYear.class          Summation.class         test.out
LeapYear.java           Summation.java          test.txt
Powers2.class           SummationFor.class
Powers2.java            SummationFor.java
Administrators-MacBook-Pro:Examples kvertanen$ javac Summation.java
Administrators-MacBook-Pro:Examples kvertanen$ java Summation 5
sum 0..1 = 1
sum 0..2 = 3
sum 0..3 = 6
sum 0..4 = 10
sum 0..5 = 15
Administrators-MacBook-Pro:Examples kvertanen$
```

Mac

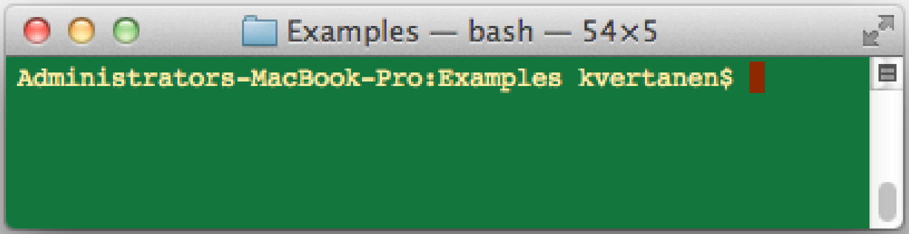
Spotlight → type "terminal"

Go → Applications → Utilities → Terminal

Getting around the command line



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.

C:\Users\keith>
```



```
Administrators-MacBook-Pro:Examples kvertanen$
```

Action	Windows	Mac OS / Unix
Move into a folder	`cd myfolder`	`cd myfolder`
Move into parent folder	`cd ..`	`cd ..`
Move into a folder, absolute folder	`cd \Users\keith`	`cd /Users/keith`
List files in current folder	`dir`	`ls`
Compile program in current folder	`javac Prog.java`	`javac Prog.java`
Run a compiled program	`java Prog`	`java Prog`
See what is in a text file	`type Prog.java`	`more Prog.java`
Auto-complete filenames	`<tab key>`	`<tab key>`
Last command	`<up arrow>`	`<up arrow>`

Input via command line

- Input via args[] array
  - Tedious to enter lots of input
  - Impossible to have interactive user input
  - e.g. What we need for a number hunting game

```
% java NumberHunt
Guess a number between 1-100? 50
Ice cold.
Guess a number between 1-100? 20
Getting warmer.
Guess a number between 1-100? 10
Hot.
Guess a number between 1-100? 5
Getting warmer.
Guess a number between 1-100? 15
Hot.
Guess a number between 1-100? 12
You nailed it!
It took you 6 guesses.
```

Standard input class

- Allows input from user or from a file
- Download `StdIn.java`
  - Place in same directory as your program
  - Refresh Eclipse project to make it show up

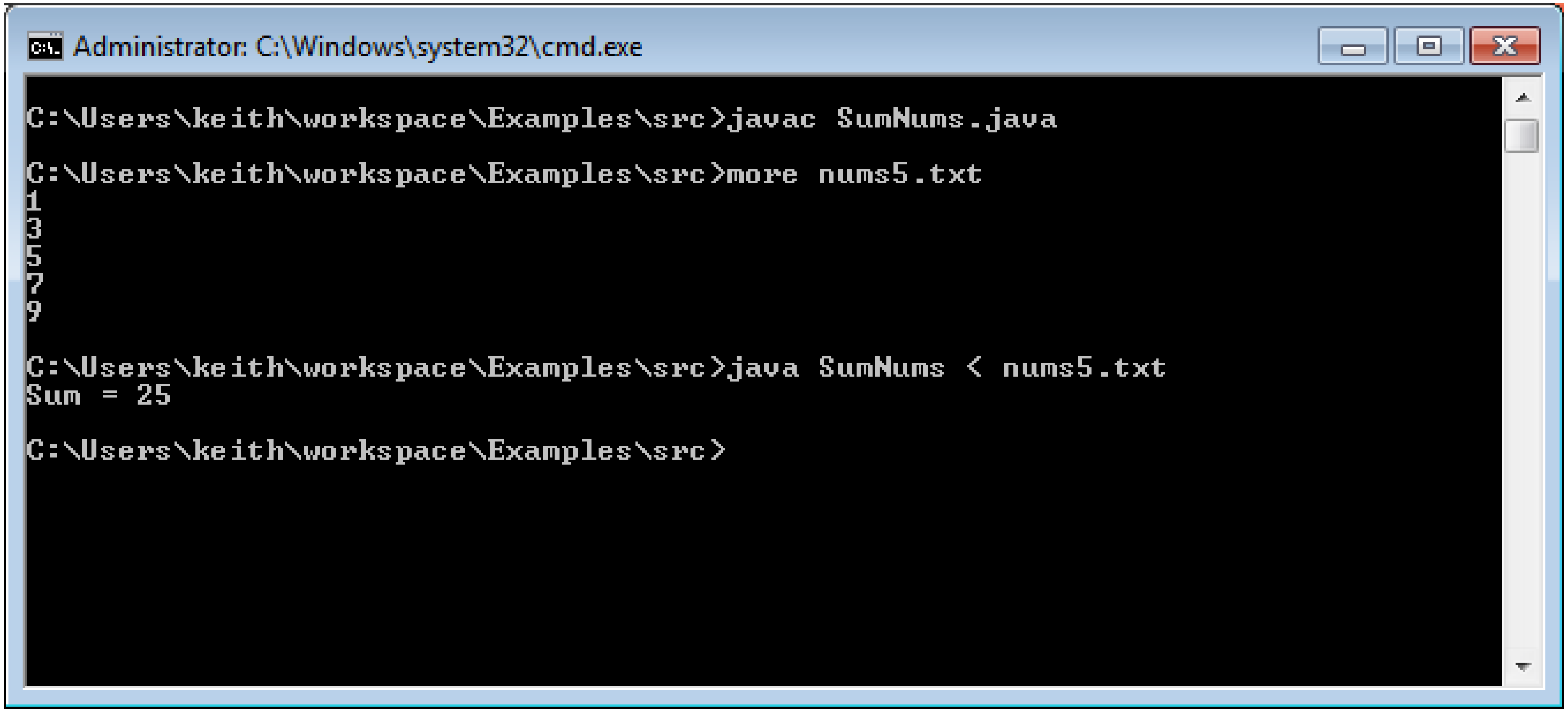
```java
public class AddTwo
{
   public static void main(String [] args)
   {
      System.out.print("Enter first integer: ");
      int num1 = StdIn.readInt();

      System.out.print("Enter second integer: ");
      int num2 = StdIn.readInt();

      int sum = num1 + num2;
      System.out.println("Sum = " + sum);
   }
}
```

Standard input class

- Reading from a file via redirection
  - Need to do from command line
    - Can't redirect file (easily) inside Eclipse
- Goal: Sum all integers in a file
  - Keep reading numbers until End Of File (EOF)
    - EOF can be sent by hitting ctrl-z or ctrl-d in Eclipse

```
public class SumNums
{
   public static void main(String [] args)
   {
      int sum = 0;
      while (!StdIn.isEmpty())
      {
         sum += StdIn.readInt();
      }
      System.out.println("Sum = " + sum);
   }
}
```

Reading from a file



```
C:\Users\keith\workspace\Examples\src>javac SumNums.java

C:\Users\keith\workspace\Examples\src>more nums5.txt
1
3
5
7
9

C:\Users\keith\workspace\Examples\src>java SumNums < nums5.txt
Sum = 25

C:\Users\keith\workspace\Examples\src>
```

StdIn.java

```
public class StdIn

boolean     isEmpty()       true if no more values, false otherwise
int         readInt()       read next int
double      readDouble()    read next double
long        readLong()      read next long
boolean     readBoolean()   read next boolean
char        readChar()      read next char
String      readString()    read next String
String      readLine()      read rest of line (until carriage return)
String      readAll()       read the rest of the text
```

```
this is an example text file
1.23 3.45
10 20
the
end
```

Combining programs

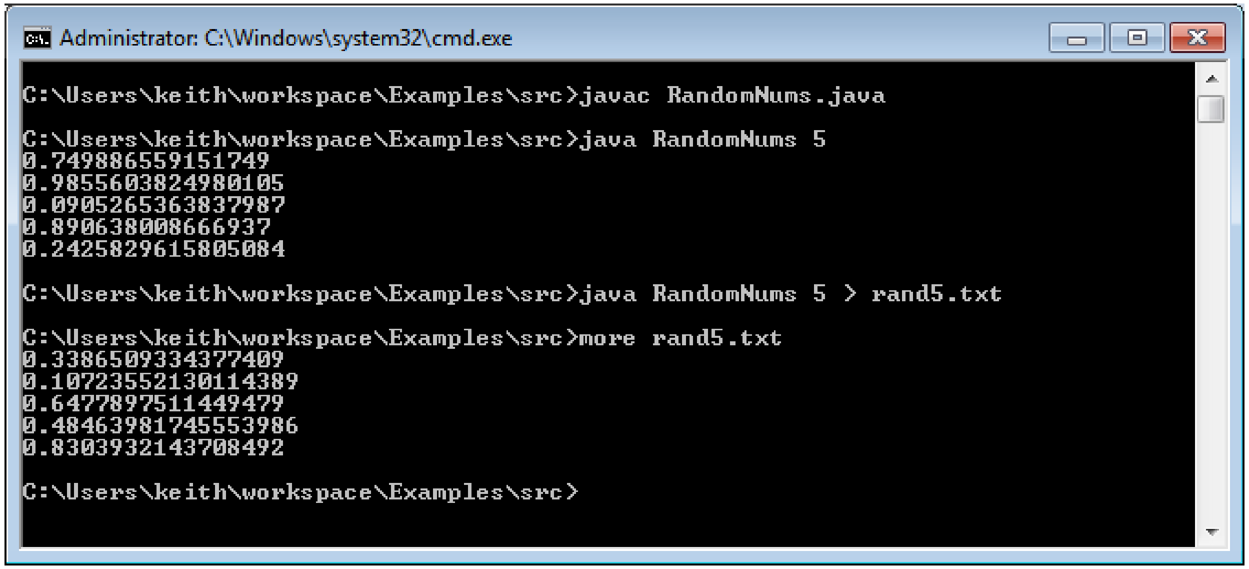
- Output can also be redirected
  - To a file (for later review) via redirection
  - Directly to another program via piping
- Example:
  - First program generates random numbers
  - Second program averages the numbers

Combining programs

```
public class RandomNums
{
   public static void main(String [] args)
   {
      int num = Integer.parseInt(args[0]);
      for (int i = 0; i < num; i++)
         System.out.println(Math.random());
   }
}
```

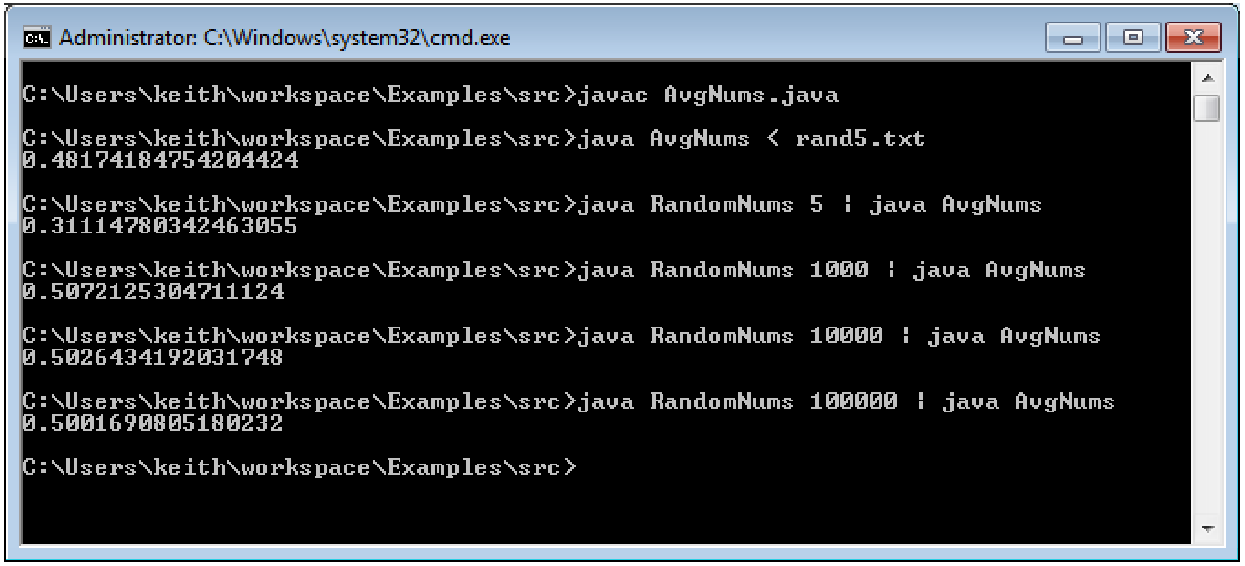
```
public class AvgNums
{
   public static void main(String [] args)
   {
      double sum   = 0.0;
      long   count = 0;
      while (!StdIn.isEmpty())
      {
         sum += StdIn.readDouble();
         count++;
      }
      System.out.println(sum / count);
   }
}
```

Averaging random numbers



```
C:\Users\keith\workspace\Examples\src>javac RandomNums.java

C:\Users\keith\workspace\Examples\src>java RandomNums 5
0.749886559151749
0.9855603824980105
0.09052653638379 87
0.8906380086669 37
0.2425829615805084

C:\Users\keith\workspace\Examples\src>java RandomNums 5 > rand5.txt

C:\Users\keith\workspace\Examples\src>more rand5.txt
0.3386509334377409
0.10723552130114389
0.6477897511449479
0.48463981745553986
0.8303932143708492

C:\Users\keith\workspace\Examples\src>
```

Redirecting program **output to a file** using **>** followed by the output filename.



```
C:\Users\keith\workspace\Examples\src>javac AvgNums.java

C:\Users\keith\workspace\Examples\src>java AvgNums < rand5.txt
0.48174184754204424

C:\Users\keith\workspace\Examples\src>java RandomNums 5 | java AvgNums
0.3111478034246305 5

C:\Users\keith\workspace\Examples\src>java RandomNums 1000 | java AvgNums
0.5072125304711124

C:\Users\keith\workspace\Examples\src>java RandomNums 10000 | java AvgNums
0.5026434192031748

C:\Users\keith\workspace\Examples\src>java RandomNums 100000 | java AvgNums
0.5001690805180232

C:\Users\keith\workspace\Examples\src>
```

Reading input from file using **<** followed by the filename.

Directly **piping output from one program to another** using pipe **|**

while loop

- while loop: common way to repeat code
  - Evaluate a `boolean` expression
  - If `true`, do a block a code
    - Go back to start of while loop
  - If `false`, skip over block

```
while (expression)
{
    statement1;
    statement2;
    …
}
```

while loop with multiple statements in a {} block

```
while (expression)
    statement1;
```

while loop with a single statement

while loop example 1

- Print out summations, 0 + 1 + 2 + … + N

```java
public class Summation
{
   public static void main(String [] args)
   {
      int  limit = Integer.parseInt(args[0]);
      int  i     = 1;
      long sum   = 0;

      while (i <= limit)
      {
         sum += i;
         System.out.println("sum 0..." + i +
                            " = " + sum);
         i++;
      }
   }
}
```

```
% java Summation 4
sum 0...1 = 1
sum 0...2 = 3
sum 0...3 = 6
sum 0...4 = 10
```

while loop example 2

- Print powers of 2 up to but not including limit

```java
public class Powers2
{
   public static void main(String [] args)
   {
      int  limit = Integer.parseInt(args[0]);
      long total = 1;
      while (total < limit)
      {
           System.out.println(total);
           total = total * 2;
      }
   }
}
```

```
% java Powers2 16
1
2
4
8
```

while loop

```
while (expression)
{
    statement1;
    statement2;
}
```

```
while (expression);
{
    statement1;
    statement2;
}
```

This semicolon is the entire body of the while loop!

Almost ***never*** what you want.

while loop

```
while (expression)
{
    statement1;
    statement2;
}
```

```
while (expression)
    statement1;
    statement2;
```

Only statement1 considered inside the while loop.

Java doesn't care about indentation. But I do (and so does your TA).

for loop

- for loop: another common type of loop
  - Execute an initialization statement
  - Evaluate a `boolean expression`
  - If `true`, do code block then increment
  - If `false`, done with loop

```
for (init; expression; increment)
{
    statement1;
    statement2;
    …
}
```

for loop versions

```
for (init; expression; increment)
{
    statement1;
    statement2;
    …
}
```

{} block version

```
for (init; expression; increment)
    statement1;
```

single line version

```
for (init; expression; increment);
{
    statement1;
    statement2;
    …
}
```

buggy version

for loop example

- Print out summations, 0 + 1 + 2 + … + N

```
public class SummationFor
{
   public static void main(String [] args)
   {
      int  limit = Integer.parseInt(args[0]);
      long sum   = 0;

      for (int i = 1; i <= limit; i++)
      {
         sum += i;
         System.out.println("sum 0..." + i +
                                   " = " + sum);
      }
   }
}
```

for loop anatomy

Declare and initialize a variable for use inside and outside the loop body

Condition which must be true to execute loop body

Changes the loop counter variable

Declare and initialize a loop control variable

```
long sum = 0;

for (int i = 1; i <= limit; i++)
{
    sum += i;
    System.out.println("sum 0..." + i +
                       " = " + sum);
}
```

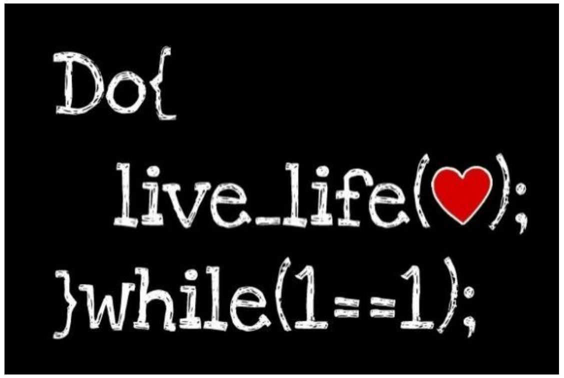
Loop body, executes 0 or more times

do while loop

- do while loop
  - Always executes loop body at least once
  - Do a block a code
  - Evaluate a boolean expression
  - If expression true, do block again

```
do
{
   statement1;
   statement2;
   …
}
while (condition);
```

do while needs this semicolon!



http://www.bhmpics.com/view-do_while_loop_for_life-1600x1200.html

do while example

- Draw random points in [0, 1)
- Stop when we draw one in interval [left, right]

```java
public class DrawPoints
{
    public static void main(String[] args)
    {
        double left  = Double.parseDouble(args[0]);
        double right = Double.parseDouble(args[1]);
        double point = 0.0;
        int    count = 0;

        do
        {
            point = Math.random();
            count++;
        }
        while ((point < left) || (point > right));

        System.out.println(count + " random draws");
    }
}
```

do while example runs

```
% java DrawPoints 0.1 0.2
9 random draws

% java DrawPoints 0.1 0.2
2 random draws
```

```
% java DrawPoints 0.1 0.11
74 random draws

% java DrawPoints 0.1 0.2
198 random draws
```

```
% java DrawPoints -0.2 -0.1
(never terminates!)

% java DrawPoints 0.2 0.1
(never terminates!)
```

- Infinite loop: possible for all loop types (while/for)
  – Eclipse, hit the red stop button
  – Command line, hit ctrl-c

Nested loops

- A loop inside another loop

```
public class StarTriangle
{
   public static void main(String[] args)
   {
      int limit = Integer.parseInt(args[0]);
      for (int i = 0; i < limit; i++)
      {
         for (int j = 0; j <= i; j++)
            System.out.print("*");
         System.out.println();
      }
   }
}
```

```
% java StarTriangle 4
*
**
***
****
```

Loop choice

- Does your loop need a counter variable?
  - e.g. Going from 0 to N or N to 0 in fixed steps
  - Use a for loop
  - Counter variable is local to loop
  - Harder to forget the increment/decrement
- Do you need an unknown number of loops?
  - Use a while loop
- Do you need to loop at least once?
  - Use a do while loop

Input/Output

Write a Java program that reads in a series of words from a file and places them in output sentences – this is a variation of MadLib, for anyone who has done these. The sentences are:

> We have an ___________ coming up next __________.
> I am going to start ______________ this weekend.
> But theses in-class ______________ and bonus ____________ really help.

The files will contain 5 words, and they should be placed in the sentences in order. The sentences should then be printed to the screen. There are two test files available on the class website, test1.txt and test2.txt. When I run my program from the command line as :

```
>> java mvandyne3 < test1.txt
```

the output should be:

```
We have an exam coming up next week.
I am going to start studying this weekend.
But these in-class reviews and bonus points really help.
```

If you like, you can write your own test file(s), and submit them with your code. Make sure you have StdIn.java in your directory.

VERY IMPORTANT: Name your program <yourusername>3.java
For example, my program would be named mvandyne3.java

Loops

Write a program that will drive anyone insane. You are to implement the “99 Bottles of Beer on the Wall” song. The user should be prompted for the number of bottles to start with, and your program should run from that number to 0 printing out the lyrics. For example, if the user ran my program and entered 99, the output should look like (with all the intermediate bottles printed out):

```
>> java mvandyne4
How many bottles of beer? 99

99 bottles of beer on the wall.
99 bottles of beer.
Take one down.
Pass it around.
98 bottles of beer on the wall.
…
1 bottle of beer on the wall.
1 bottle of beer.
Take one down.
Pass it around.
No more bottles of beer on the wall.
```

VERY IMPORTANT: Name your program <yourusername>4.java
For example, my program would be named mvandyne4.java