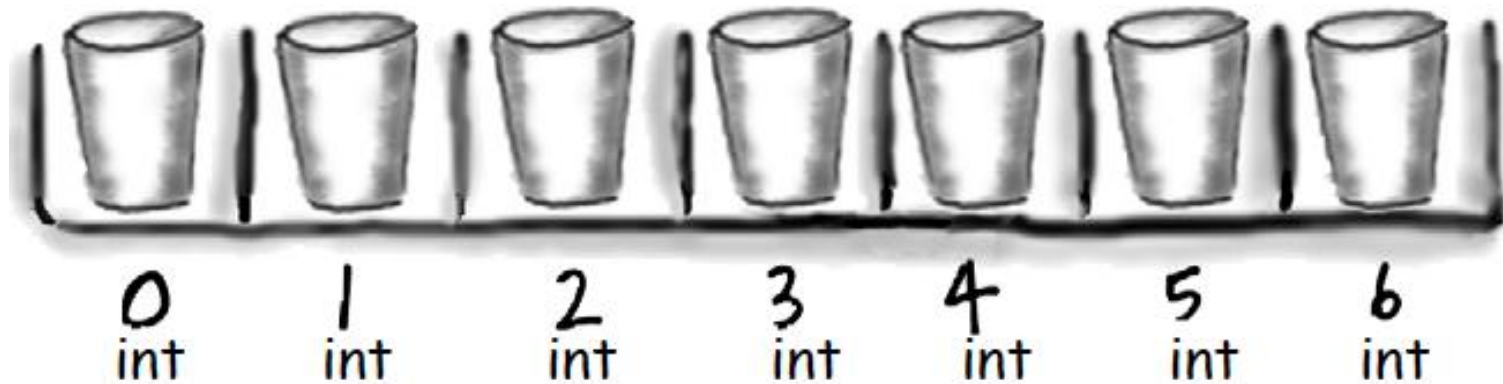


# Command line, standard input, and arrays

```
Administrator: C:\Windows\system32\cmd.exe
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
0.31114780342463055
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>
```



# Overview

- Learning to use the **command line**
- **New ways to get input** into your programs:
  - Read information **from user**
  - Read information **from a file**
  - Read information **from another program**
- **New way to store things**
  - **Arrays**: store multiple things under one name
  - e.g. `args[0]`, `args[1]`, `args[2]`

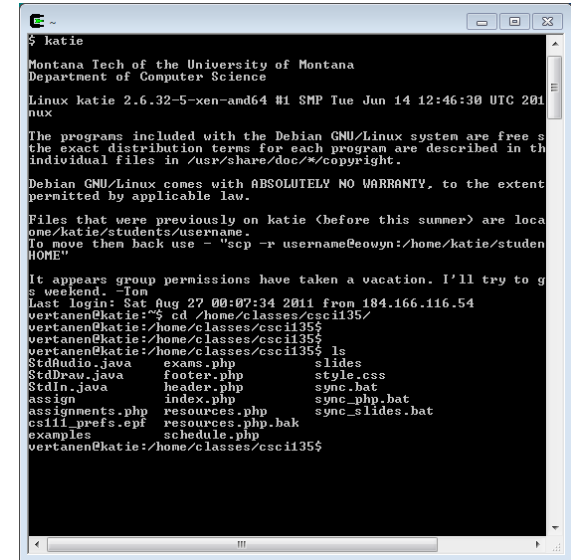
# Interfacing with your computer

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

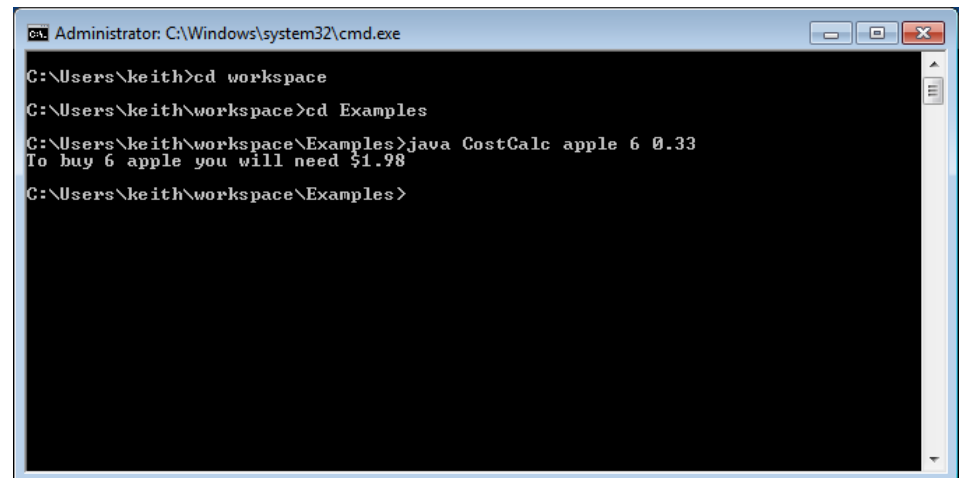


# Interfacing with your computer

- **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**

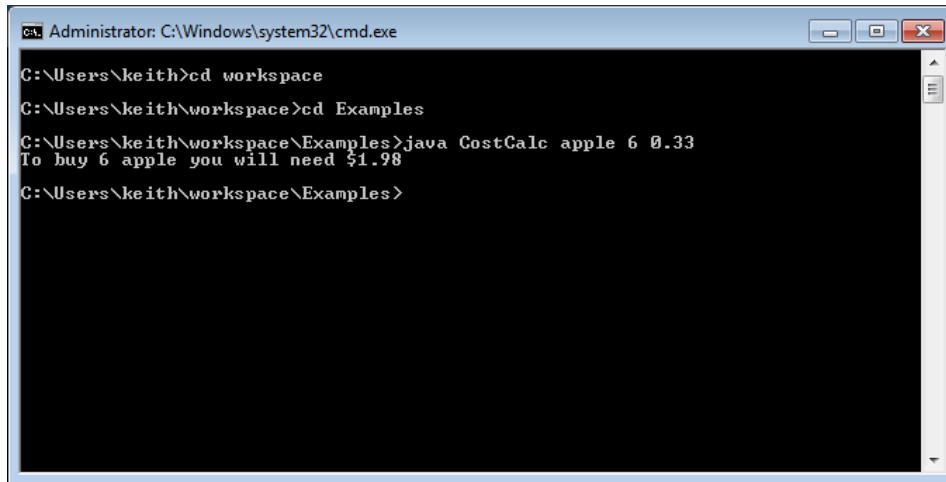


```
~  
$ katie  
Montana Tech of the University of Montana  
Department of Computer Science  
Linux katie 2.6.32-5-xen-amd64 #1 SMP Tue Jun 14 12:46:30 UTC 2011  
nux  
The programs included with the Debian GNU/Linux system are free s  
the exact distribution terms for each program are described in th  
individual files in /usr/share/doc/*/copyright.  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the exten  
permitted by applicable law.  
Files that were previously on katie (before this summer) are loca  
one/katie/students/username.  
To move them back use - "scp -r username@oovyn:/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/cscii35/  
vertanen@katie:/home/classes/cscii35$  
vertanen@katie:/home/classes/cscii35$ 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  
cscii_prefs.epf  resources.php.bak  
examples          schedule.php  
vertanen@katie:/home/classes/cscii35$
```



```
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>
```

# Starting a command shell



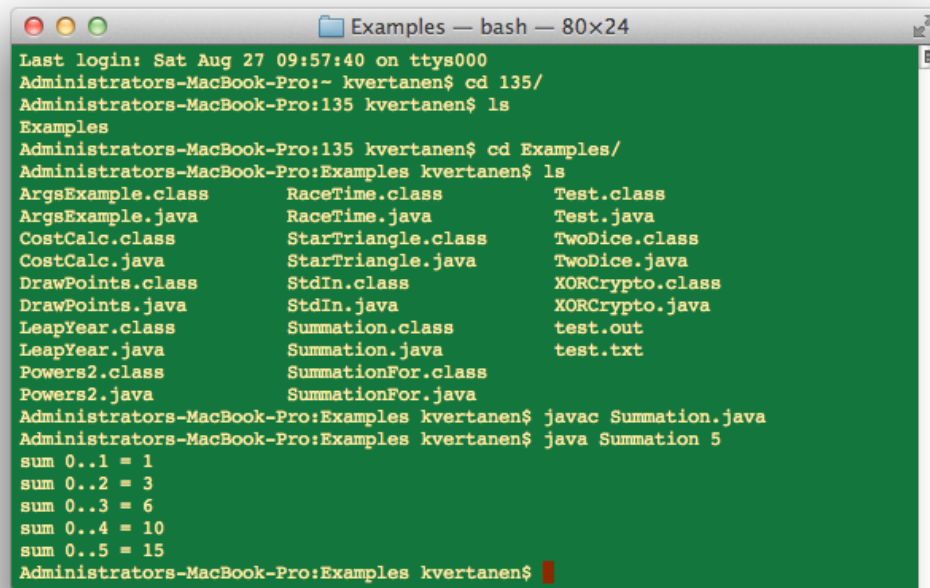
A screenshot of a Windows Command Prompt window titled "Administrator: C:\Windows\system32\cmd.exe". The window shows the following commands and output:

```
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

Start → type "cmd"

All Programs → Accessories →  
Command Prompt



A screenshot of a Mac Terminal window titled "Examples — bash — 80x24". The window shows the following commands and output:

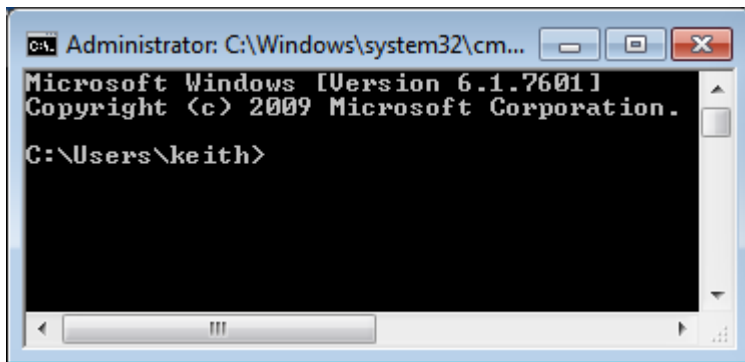
```
Last login: Sat Aug 27 09:57:40 on ttys000
Administrators-MacBook-Pro:~ kvertanen$ cd /
Administrators-MacBook-Pro:~ kvertanen$ ls
Examples
Administrators-MacBook-Pro:~ kvertanen$ cd Examples/
Administrators-MacBook-Pro:Examples kvertanen$ ls
ArgsExample.class      RaceTime.class        Test.class
ArgsExample.java       RaceTime.java         Test.java
CostCalc.class         StarTriangle.class    TwoDice.class
CostCalc.java          StarTriangle.java     XORCrypto.class
DrawPoints.class       StdIn.class           XORCrypto.java
DrawPoints.java        StdIn.java            test.out
LeapYear.class         Summation.class       test.txt
LeapYear.java          Summation.java
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"

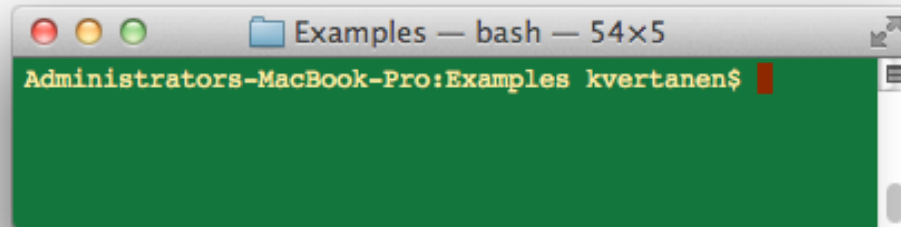
Go → Applications → Utilities  
→ Terminal

# Getting around the command line



```
Administrator: C:\Windows\system32\cm...
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.

C:\Users\keith>
```



```
Examples — bash — 54x5
Administrators-MacBook-Pro:Examples kvertanen$
```

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

# 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

```
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);
    }
}
```

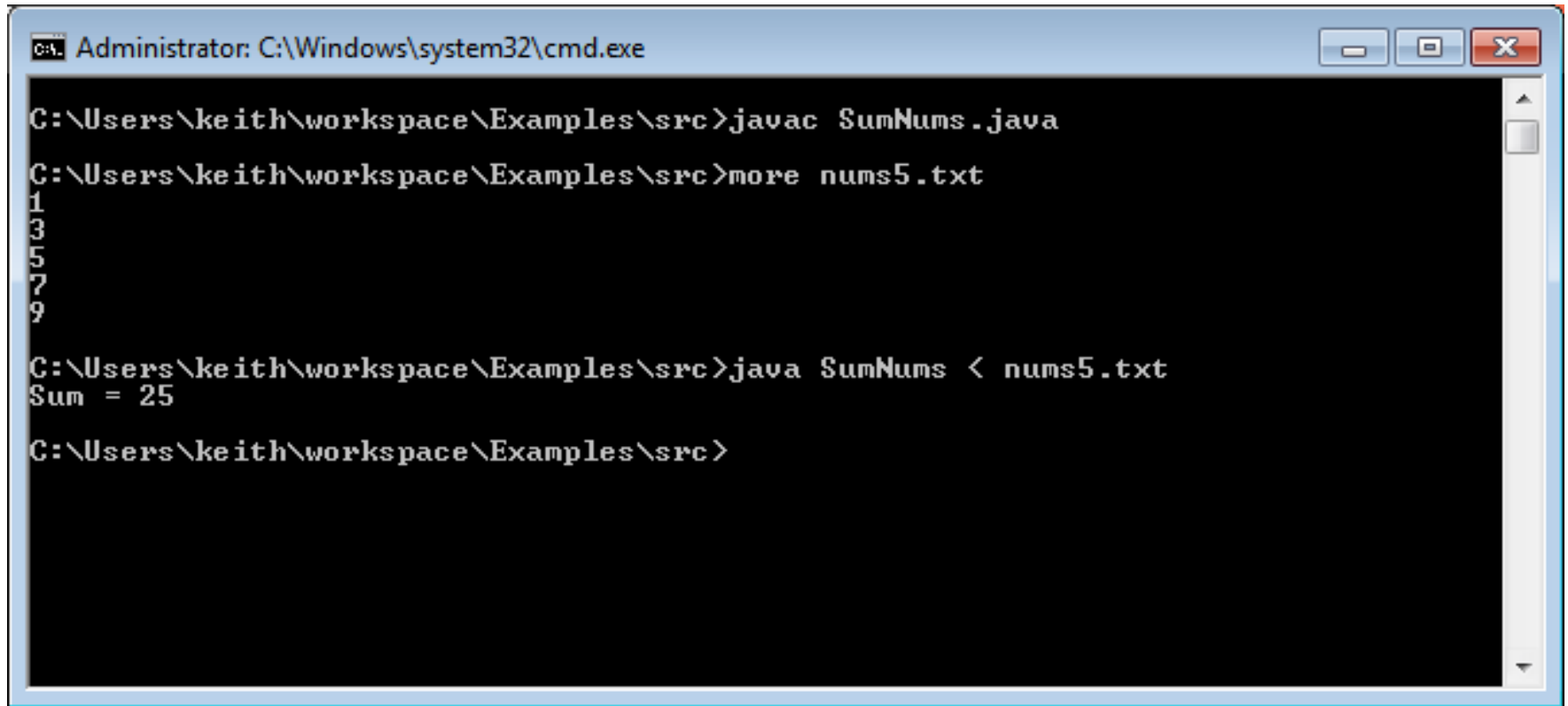


# Standard input class

- 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:\Windows\system32\cmd.exe
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

```
Administrator: C:\Windows\system32\cmd.exe
C:\Users\keith\workspace\Examples\src>javac RandomNums.java
C:\Users\keith\workspace\Examples\src>java RandomNums 5
0.749886559151749
0.9855603824980105
0.0905265363837987
0.890638008666937
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.

```
Administrator: C:\Windows\system32\cmd.exe
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
0.31114780342463055
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 **|**

# Enter the zombies...



<http://www.flickr.com/photos/purplemattfish/4075367455/>

# Zombie Apocalypse

Level: 0

```
. . ! . . . . . . . .  
. . . . . . . . . . .  
. . . . . . . . . . .  
. . . . * . . . . . . .  
. . . . . . . . . . .  
. . . . . . . . . . .  
. . . . . . . . . . .  
. . . . . . . . . . .  
. . . . . . . . . . .  
. . . . . . . . . . .  
. . . . . . . . . . .  
. . . . . . . . . . #
```

```
Direction? s  
You walked south  
Zombie went east
```

*How do I keep track of location of the person and the zombie?*

```
int personX = 0;  
int personY = 0;  
  
int zombieX = 0;  
int zombieY = 0;
```

*How do I detect when the person gets eaten?*

```
if ((personX == zombieX) && (personY == zombieY))  
{  
    System.out.println("Zombie got your braaaains!");  
    gameOver = true;  
}
```



# Extreme Zombie Apocalypse

Level: 0

```
. . ! . . . . . . . . .
. . . * . . . . . . . . .
. . . . . . . . . . . . . . .
. . . . * . . . . . . . . .
. . . . . . . . . . . . . . .
. . . . . . . . . . . . . . .
. . . . . . . . . . . . . . .
. . . . . . . . . . . . . . .
. . . . . . . . . . . . . . .
. . . . . . . . . . . . . . #
```

Direction? s

You walked south

Zombie went east

*What if we need to keep track of two zombies?*

```
int personX = 0;
```

```
int personY = 0;
```

```
int zombieX1 = 0;
```

```
int zombieY1 = 0;
```

```
int zombieX2 = 0;
```

```
int zombieY2 = 0;
```

```
...
```

```
if (((personX == zombieX1) && (personY == zombieY1)) ||
    ((personX == zombieX2) && (personY == zombieY2)))
```

```
{
```

```
    System.out.println("Zombie got your braaaains!");
```

```
    gameOver = true;
```

```
}
```

# Super Extreme Zombie Apocalypse

Level: 0

```
. . . ! . . . * . . . . .
. . . . . * . . . . . . .
. . . . . . . . . . . . .
. . . . . . . . . . . . .
. . . . . * . . . . . . .
. . . . . . . . . . . . .
. . . . . . . . . . . . .
. . . . . . . . . . . . .
. . . . . . . . . . . . .
. . . . . . . . . . . . .
. . . . . . . . . . . . #
```

```
Direction? s
You walked south
Zombie went east
```

*What if we need to keep track of three zombies?*

```
int personX = 0;
int personY = 0;

int zombieX1 = 0;
int zombieY1 = 0;

int zombieX2 = 0;
int zombieY2 = 0;

int zombieX3 = 0;
int zombieY3 = 0;

...

if (((personX == zombieX1) && (personY == zombieY1)) ||
    ((personX == zombieX2) && (personY == zombieY2)) ||
    ((personX == zombieX3) && (personY == zombieY3)))
{
    System.out.println("Zombie got your braaaains!");
    gameOver = true;
}
```

# Zombie Apocalypse: The Rising

You walked south  
Zombie went west  
Level: 5

```
. * . * .  
. . . * .  
! . * . .  
* . . . .  
. . * . #  
Direction?
```

*What if we want to add one zombie every time the player advances a level?*

**No good way to do this with simple variables!**

# Arrays to the rescue!

- We've already seen **arrays**:

```
public static void main(String [] args)
```

```
% java CostCalc bananas 12 0.21  
To buy 12 bananas you will need $2.52
```

identifier	meaning	value	type
args[0]	1 <sup>st</sup> thing on command line after Java class name	"bananas"	String
args[1]	2 <sup>nd</sup> thing on command line	"12"	String
args[2]	3 <sup>rd</sup> thing on command line after Java class	"0.21"	String
args.length	# of things on command line	3	int

# Arrays: creating many things

- **Arrays:** create many variables of same type
- **Goal: Ten variables of same type**
  - e.g. To hold the values 0-9

```
int a0, a1, a2, a3, a4, a5, a6, a7, a8, a9;  
a0 = 0;  
a1 = 1;  
a2 = 2;  
a3 = 3;  
a4 = 4;  
a5 = 5;  
a6 = 6;  
a7 = 7;  
a8 = 8;  
a9 = 9;
```

# Arrays: creating many things

- **Arrays:** create many variables of same type
- **Goal: Ten variables of same type**
  - e.g. To hold the values 0-9

```
int [] a = new int[10];  
a[0] = 0;  
a[1] = 1;  
a[2] = 2;  
a[3] = 3;  
a[4] = 4;  
a[5] = 5;  
a[6] = 6;  
a[7] = 7;  
a[8] = 8;  
a[9] = 9;
```

# Arrays: accessing elements

- **Arrays:** we can use a variable as the index!
  - Makes code shorter, cleaner, less buggy

```
int N = 10;           // size of array
int [] a;            // declare array
a = new int[N];      // create array

for (int i = 0; i < a.length; i++) // initialize array elements
    a[i] = i;        // to be 0 - 9
```



**1<sup>st</sup> element of array is a[0].  
We count from zero in  
computer science!**

# Arrays: easy to extend

- **Arrays:** can hold lots and lots of data
  - Same code, but now holds 100,000 integers:

```
int N = 100000;           // size of array
int [] a;                 // declare array
a = new int[N];           // create array

for (int i = 0; i < a.length; i++) // initialize array elements
    a[i] = i;             // to be 0 - 9
```



# Arrays: loading data from file

```
4  
fee  
fi  
fo  
fum
```

*4words.txt*

"There are going to be 4  
words to read in"

- Read words into array
- Print out words in reverse order

```
% java Backwards < 4words.txt  
fum fo fi fee
```

# Arrays: loading data from file

```
4  
fee  
fi  
fo  
fum
```

```
% java PrintBackward < 4words.txt  
fum fo fi fee
```

```
public class Backwards  
{  
    public static void main(String [] args)  
    {  
        int num = StdIn.readInt();  
        String [] words = new String[num];  
        for (int i = 0; i < num; i++)  
            words[i] = StdIn.readString();  
        for (int i = num - 1; i >= 0; i--)  
            System.out.print(words[i] + " ");  
        System.out.println();  
    }  
}
```

# Super Extreme Zombie Apocalypse

## What if we need to keep track of three zombies?

```
Level: 0
. . . ! . . . . . . . . . .
. . . . . . . . . . . . . *
. . . . . . . . . . . . . .
. . . . . . . . . . . . . *
. . . . . . . . . . . . . .
. . . . . . . . . . . . . *
. . . . . . . . . . . . . .
. . . . . . . . . . . . . .
. . . . . . . . . . . . . .
. . . . . . . . . . . . . #
Direction? s
You walked south
Zombie went east
```

```
int personX = 0;
int personY = 0;
final int NUM_ZOMBIES = 3; // constant defining # of zombies

int [] zombieX = new int[NUM_ZOMBIES]; // declare & create x-pos array
int [] zombieY = new int[NUM_ZOMBIES]; // declare & create y-pos array

// Set random initial location for each zombie (they can overlap)
for (int i = 0; i < NUM_ZOMBIES; i++)
{
    zombieX[i] = (int) (Math.random() * 10); // set i-th zombie's x-pos
    zombieY[i] = (int) (Math.random() * 10); // set i-th zombie's y-pos
}

...

int i = 0;
while ((i < zombieX.length) && (!gameOver))
{
    if ((personX == zombieX[i]) &&
        (personY == zombieY[i]))
    {
        System.out.println("Zombie got your braaaains!");
        gameOver = true;
    }
    i++;
}
```

# Summary

- **Command line**
  - Redirect **output to a file**
  - Redirect **input from a file**
  - **Pipe output between programs**
- **Standard input**
  - Easy way to **read from user or file**
- **Arrays**
  - Allow easy **storage of similar data**
  - Crucial for developing more **advanced programs**

