CS 121 – Intro to Programming:Java - Lecture 7
Announcements

Next OWL assignment up, rather large, due Friday (no programming assignment due this week)

Next book chapter up at the end of the week

Exams back next Thursday. Grades should be up sooner
While loops - Java’s other important looping construct

Here is the standard example

```java
int j = 1; // control variable
while (j <= 5){ // while test
    System.out.println(j); // body - 2 stmts
    j = j + 1;
}
```

Loop executes until test fails
int n = 0;
    while (n < 5){
        System.out.println(n);
        n = n + 1;
    }

What does this loop do?

And this:
int n = 7;
    while (n > 3){
        System.out.println(n);
        n = n - 1;
    }
What about this loop?

```java
int n = 1;
while (n != 10){
    System.out.println(n);
    n = n + 2;
}
```
public class SplitTest{

public static void main(String[] args){
    int num = 300000;
    int splitCount = 0;
    while (num > 1){
        num = num/2;
        splitCount++;
    }
    System.out.println("number of splits: "+splitCount);
}
}

----->
number of splits: 18
From an old midterm - what does this method do (assume in SimpleDice class)?

```java
public int huh(int big){
    int count = 0;
    int total = 0;
    while(total < big){
        count = count + 1;
        total = total + throwDice();
    }
    return count;
}
```
Palindromes - the same forwards and backwards

otto


Live dirt up a sidetrack carted is a putrid evil

A man a plan a canal, Panama!

Able was I ere I saw Elba

Remarkable was I ere I saw Elba, Kramer

And so forth..
left ->

<- right
otto

amanaplanacanalpanama

Center: position = 10
public class Palindrome{
    String letters;

    public Palindrome(String s){
        letters = s;
    }

    public boolean palCheck(){
        int left = 0; // position of first char
        int right = letters.length() - 1; // position of last char
        boolean ok = true; // assume a palindrome
        while(left < right){
            if (letters.charAt(left) != letters.charAt(right))
                ok = false;
            left++;
            right--;
        }
        return ok;
    }
}
What happens with this proposed palindrome?

acbdefghijklmonpqrtuvwxyzthequickbrownfoxj
umpedoverthelazyxrtmmmmmmmaiwazicfehwopp
Java’s `break` statement - kicks you out of loop

```java
public boolean palCheck()
{
    int left = 0; // position of first char
    int right = letters.length() - 1; // position of last char
    boolean ok = true; // assume a palindrome
    while(left < right){
        if (letters.charAt(left) != letters.charAt(right))
            ok = false;
        if (!ok)
            break; // here is the break stmt
        left++;
        right--;
    }
    return ok;
}
```
Java’s `switch` statement - a generalized conditional statement

```java
public void rateLetter(char ch) {
    switch(ch){
        case 'a':
            case 'e':
            case 'i':
            case 'o':
            case 'u':
                System.out.println("vowel");
                break;
        case 'y':
            System.out.println("vowel- consonant");
            break;
        default:
            System.out.println("consonant");
    }
}
```
Features:

1) You can gang cases

2) You’ve got to break out of a case, either with a break statement, or via some other mechanism (such as a return statement) - otherwise you fall through.

3) The case values must be of type int or char

4) General syntax:

```java
switch(val)
{
    case x:
    case y:  {blah blah; break;} // x or y gets you here
    case z:  blah2; break;
    default: do-something-else..;  }
```
The game of Craps

The basic game of craps is very simple. On the first roll of two dice (the come-out roll), the shooter wins by rolling either a 7 or 11 (a natural). Rolling craps (2, 3, or 12) loses. Any other number (4, 5, 6, 8, 9, or 10) is called the point.

Now to win, the point number must be rolled before a 7. If a 7 is rolled before the point number, the shooter loses; if the point comes up first, the shooter wins.
Some games:

7

4 4

4 5 6 7

4 5 6 8 9 10 11 4

4 5 6 8 9 10 11 7

3

11

10 9 8 2 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 10
// runs 100,000 craps games - how many wins?

class CrapsTester
{
    public static void main(String args[])
    {
        public int gameCount = 100000;
        Craps C = new Craps();
        int count = 0;
        for (int i = 0; i < gameCount; i++)
            if (C.crapsRun())
                count++;
        System.out.println((double)count / gameCount);
    }
}
public class SimpleDice{

public int tossDie(){
    return (1+ (int)(Math.random()*6));
}

public int throwDice(){
    return (tossDie() + tossDie());
}
}
public class Craps
{
    SimpleDice d = new SimpleDice();

    public boolean crapsRun()
    {
        ....
        true means: you win, false, you lose
public boolean crapsRun()
{
    int point = 0;
    int cur;
    cur = d.throwDice();
    switch(cur) {
    case 7:
    case 11: return(true);
    case 2:
    case 3:
    case 12: return(false);
    default: point = cur;
    }
    while (true)
    {
    cur = d.throwDice();
    if (cur == point)
    return(true);
    else if (cur == 7)
    return(false);
    }  
}
public boolean crapsRun2()
{
    int point = 0;
    int cur; boolean done = false; boolean result = false;
    cur = d.throwDice();
    switch(cur){
    case 7:
    case 11: {done = true; result = true; break;}
    case 2:
    case 3:
    case 12: {done = true; result = false; break;}
    default: point = cur;
    }
    if (done) return result; else
    while (true)
    { cur = d.throwDice();
        if (cur == point) return(true);
        else if (cur == 7)
            return(false);}
}
The “123” problem. (also: the “98.6” problem)

These are Strings - but they represent numbers

At times we run into them as Strings, but we want to interpret them as numbers

Java has a system of wrapper classes (8 in all, one for each primitive type)

Most prominent: Integer, Double, Character - these shadow int, double, char

Integer num = new Integer(23); turns 23 into the object 23

If k is an Integer, int num = k; will do the conversion automatically (but there’s a caveat..)

These classes come with important utility methods:

double r = Double.parseDouble("35.7");

int number = Integer.parseInt("123");
import javax.swing.JOptionPane;

public class SickorWell{

    public static void main(String[] args){
        String myTemp;
        double temp;
        myTemp = JOptionPane.showInputDialog("Enter your temperature");
        temp = Double.parseDouble(myTemp);
        if(temp > 98.6)
            JOptionPane.showMessageDialog(null,"you are sick!");
        else
            JOptionPane.showMessageDialog(null,"you are well!");
    }
}