### Exercise 13.1 pg. 500

1. Assign a value to the base case and starting point of recursive.
2. The function for the specific value of N and the other is the initialized value of the starting point.
3. Recursion usually takes longer than incrementing and testing a loop control variable, also a method call ties up memory this not freed until the method completes its task.
4. Easier to understand and provides automatic objects at each function call.
5. 2 \* raise (2,4)–> 2\*2\*raise(2,3)–> 2\*2\*2\*raise(2, 2)–> 2\*2\*2\*2\*raise(2,1)–> 2\*2\*2\*2\*2\*raise(2,0)–>2\*2\*2\*2\*2\*1= 32
6. It will go on forever because n will never reach n

### Projects

//project 13-1

**public** **class** testclass {

**public** **static** **int** gcd(**int** a, **int** b){

**if** (b == 0)

**return** a;

**else**

**return** *gcd*(b,a%b);

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

System.***out***.println(*gcd*(20,6));

}

}

**public** **class** project132 {

**private** **static** String s ="";

**public** **static** String backwards(String Str,**int** pos){

**if** (Str.length()< pos)

**return** **null**;

s = s+ Str.charAt(Str.length()- s.length() -1);

*backwards*(Str, pos +1);

**return** Str.substring(0, Str.length() -s.length()) + s;

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

System.***out***.println(*backwards*("hello", 3));

}

}

### Labs

//© A+ Computer Science - www.apluscompsci.com

//Name - Josh Martin

//Date -03/19/2015

//Class - Ap Computer Science

//Lab - Recursion Lab 1

**import** java.awt.Graphics;

**import** java.awt.Color;

**import** java.awt.Font;

**import** javax.swing.JPanel;

**public** **class** RecursiveCircles **extends** JPanel

{

**public** RecursiveCircles()

{

setSize(640,480);

setBackground(Color.WHITE);

setVisible(**true**);

}

**public** **void** paintComponent( Graphics window )

{

**super**.paintComponent(window);

window.setColor(Color.ORANGE);

window.setFont(**new** Font("TAHOMA",Font.BOLD,12));

window.drawString("Lab21a",20,40);

window.drawString("Drawing Circles Using Recursion ",20,80);

drawCircles(window,10,10);

}

**public** **void** drawCircles(Graphics window, **int** x, **int** y)

{

//base case

window.setColor(Color.BLUE);

**if** (x <100){

window.drawOval(90+x, 90+y, 20+x, 20+x);

drawCircles(window, x+10, y+10);

}

}

}

//© A+ Computer Science - www.apluscompsci.com

//Name - Josh Martin

//Date - 03/19/2015

//Class - Ap Computer Science

//Lab - Recursion Lab 2

**import** **static** java.lang.System.\*;

**public** **class** RecursionFunOne

{

**public** **static** **int** countOddDigits(**int** num)

{

**int** count =0;

**if** (num % 2 == 1 )

count ++;

**if**( num > 0)

**return** *countOddDigits*(num/10) + count;

**return** count;

}

}

//© A+ Computer Science - www.apluscompsci.com

//Name -Josh Martin

//Date -

//Class -

//Lab - Lab 3

**import** **static** java.lang.System.\*;

**public** **class** RecursionFunTwo

{

**public** **static** **int** countChickens(String word)

{

**int** index =word.indexOf("chicken");

**if**(index != -1)

{

**if**(index == 0)

word= word.substring(7);

**else**

word = word.substring(0,index )+ word.substring(index+7);

**return** 1 + *countChickens*(word);

}

**return** 0;

}

}

**public** **void** countAts(**int** r, **int** c)

{

// Lab 4

//add in recursive code to count up the # of @s connected

//start checking at spot [r,c]

**if**(r >= 0 && c >= 0 && ( r <atMat.length) &&( c < atMat[0].length) && (atMat[r][c] == '@'))

{

atCount++;

atMat[r][c] = '0';

countAts(r,c +1);

countAts(r,c-1);

countAts(r+1,c);

countAts(r-1,c);

}

}