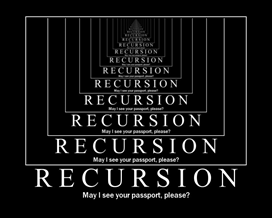
Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Period \_\_\_\_\_\_\_



Recursion Exercises

Practice #1

Use the following method puzzle to answer problems 1 - 3.

/\* precondition: base and limit are nonnegative numbers

public int puzzle(int base, int limit)

{

if ( base > limit )

return -1;

else if ( base == limit )

return 1;

else

return base \* puzzle(base + 1, limit);

}

1. Identify the base case(s) of method puzzle if base >= limit*( base < limit ) and ( base == limit )*

2. Identify the recursive case(s) of method puzzle if base < limit *base < limit*

3. Show what would be displayed by the following calls.

a) System.out.println(puzzle(14,10)); -1

b) System.out.println(puzzle(4,7)); 840 *-1*

c) System.out.println(puzzle(0,0)); 1

*1*

In problems 4 - 6, show the output that will be displayed by the call show(123);

|  |  |  |
| --- | --- | --- |
| 4. | 5. | 6. |
| public void show ( int n )  {  if (n > 0)  show (n/10);  System.out.print (n%10);  }  *123* | public void show ( int n)  {  System.out.print (n%10);   if (n>0)  show (n/10);  }  *321* | public void show ( int n)  {  System.out.print (n%10);  if (n>0)  show (n/10);  System.out.print (n%10);  }*321123* |
| 0 | 12, 1, 0 | 12, 1, 0, 0, 0 |

7. Complete the code to recursively evaluate the 8. What does this code count?

sum: sum = 1 + 1/2 + 1/3 +…+1/n, n ≥ 1.

public double sum(int n) // n>=1 public int count (int n)

{ {

if (\_*n == 1*\_\_\_\_\_\_\_\_\_\_) if (n==1)

return \_*1*\_\_\_\_\_\_\_\_\_\_\_\_; return 1;

return 1 + count(n–1);

return \_\_*1/n*\_\_\_\_ + sum(\_*n-1*\_\_\_\_\_); }

} \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Recursion Exercises

Practice #2

Evaluate each recursive function foo for the given parameter values.

1. //x is nonnegative

public int foo (int x){ foo(3)

if (x == 0)

return (x + 1);

return 1 + foo (x-1);

}

2. public int foo (int x, int y) foo(6,5)

{

if (x <= 0)

return (0);

if (y >= x)

return 1 + foo(y,x);

return 2 + foo(x-3,y-1);

}

3. public int foo(int x) foo(foo(foo(foo(foo(foo(3)))))))

{

if (x < 5)

return x\*x + 1;

if (x == 5)

return x\*x - 3;

return foo(x-2);

}

Show the output of each recursive method.

4. 5.

|  |  |
| --- | --- |
| public void p(int n)  {  p(n - 1);  System.out.print (n);  p(n - 1);  }  . . .  p(3);  System.out.println(); | int[] list = new int[10];  // list contains { 4, -1, 5, 1, 8, 3, -2, 1, 6, 7 };  public int calc(int[] list, int first, int last)  {  if ( first > last )  return 0;  return list[first] + calc(list,first+1,last);  }  . . .  System.out.println(″Result:″ + calc (list, 2, 7)); |

6. What happens when you call doSomething(3)?

public static void doSomething(int value){

if (value > 0){

doSomething(value-1);

doSomething(value-1);

System.out.print(″ ″ + value);

}