**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 1

\* This program will return n number of bunny ears considering bunnies have two ears, without multiplication or looping

\*/

**public** **class** BunnyEars

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

**int** ears = 0;

System.***out***.println("Bunny Ears (Recursive)\n======================This program will return n number of bunny ears\nconsidering bunnies have two ears recursively, without multiplication or looping");

System.***out***.println("\nEnter the number of bunnies:");

ears = sc.nextInt();

System.***out***.println("\nThe number of bunny ears for " + ears + " ears is: " + *bunnyEars*(ears) + " bunny ears.");

}//end main

**public** **static** **int** bunnyEars(**int** bunnies)

{

**if**(bunnies <= 0)

{

**return** 0;

}//end if

**return**(2 + *bunnyEars*(bunnies - 1));

}//end method bunnyEars

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 2

\* This program will return n number of bunny ears considering odd bunnies have two ears and even bunnies have 3, without multiplication or looping

\*/

**public** **class** BunnyEars2

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

**int** ears = 0;

System.***out***.println("Bunny Ears 2 (Recursive)\n======================\nThis program will return n number of bunny ears considering odd bunnies\nhave two ears and even bunnies have 3, without multiplication or looping.");

System.***out***.println("\nEnter the number of bunnies:");

ears = sc.nextInt();

System.***out***.println("\nThe number of bunny ears for " + ears + " ears is: " + *bunnyEars2*(ears) + " bunny ears.");

}//end main

**public** **static** **int** bunnyEars2(**int** bunnies)

{

**if**(bunnies == 0)

{

**return** 0;

}//end if

**else** **if**(bunnies % 2 == 0)

{

**return**(3 + *bunnyEars2*(bunnies - 1));

}//end else if

**else**

{

**return** (2 + *bunnyEars2*(bunnies - 1));

}//end else

}//end method bunnyEars2

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 3

\* We have triangle made of blocks. The topmost row has 1 block, the next row down has 2 blocks, the next row has 3 blocks, and so on. Without looping or multiplication.

\*/

**public** **class** Triangle

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

**int** rows = 0;

System.***out***.println("Triangle (Recursive)\n======================\nWe have triangle made of blocks. The topmost row has 1 block,\nthe next row down has 2 blocks, the next row has 3 blocks, and so on.");

System.***out***.println("\nEnter the number of rows:");

rows = sc.nextInt();

System.***out***.println("\nThe number of blocks in a " + rows + " row triangle is: " + *triangle*(rows) + " blocks.");

}//end main

**public** **static** **int** triangle(**int** rows)

{

**if**(rows == 0)

{

**return** 0;

}//end if

**return**(rows + *triangle*(rows - 1));

}//end method triangle

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 4

\* This program, given a non-negative int n, will return the sum of its digits

\*/

**public** **class** SumDigits

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

**int** number = 0;

System.***out***.println("Sum Digits (Recursive)\n======================\nGiven a non-negative integer n, this program will return the sum of its digits");

System.***out***.println("\nEnter a number that for the sum of its digits:");

number = sc.nextInt();

System.***out***.println("\nThe sum of the digits of " + number + " is: " + *sumDigits*(number));

}//end main

**public** **static** **int** sumDigits(**int** number)

{

**if**(number == 0)

{

**return** 0;

}//end if

**return**(number % 10 + *sumDigits*(number / 10));

}//end method sumDigits

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 5

\* This program, given a non-negative int n, will return the count of the occurrences of 7 as a digit

\*/

**public** **class** Count7

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

**long** number = 0;

System.***out***.println("Count 7 (Recursive)\n======================\nGiven a non-negative integer n, this program will\nreturn the count of the occurrences of 7 as a digit");

System.***out***.println("\nEnter a number to be checked for 7's:");

number = sc.nextInt();

System.***out***.println("\nThe occurances of the digit 7 in " + number + " is: " + *count7*(number));

}//end main

**public** **static** **int** count7(**long** n)

{

**if**(n < 1)

{

**return** 0;

}//end if

**else** **if** (n % 10 == 7)

{

**return** (1 + *count7*(n / 10));

}//end else if

**else**

{

**return**(*count7*(n / 10));

}//end else

}//end method count7

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 6

\* Given a non-negative int n, compute recursively (no loops) the count of the occurrences of 8 as a digit, except that an 8 with another 8 immediately to its left counts double, so 8818 yields 4.

\*/

**public** **class** Count8

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

**long** number = 0;

System.***out***.println("Count 8 (Recursive)\n======================\nGiven a non-negative int n, this program will compute recursively (no loops) the count of the occurrences of 8 as a digit,\nexcept that an 8 with another 8 immediately to its left counts double, so 8818 yields 4.");

System.***out***.println("\nEnter a number to be checked for 8's:");

number = sc.nextInt();

System.***out***.println("\nThe occurances of the digit 8 in " + number + " (considering the double condition) is: " + *count8*(number));

}//end main

**public** **static** **int** count8(**long** number)

{

**if**(number < 1)

{

**return** 0;

}//end if

**else** **if** (number % 10 == 8)

{

**if**((number / 10) % 10 == 8)//for when the digit to the left of it is 8, by removing the last eight/digit and checking the digit to the left of that last eight/digit

{

**return** (2 + *count8*(number / 10));

}//end if

**else**

{

**return** (1 + *count8*(number / 10));

}//end else

}//end else if

**else**

{

**return**(*count8*(number / 10));

}//end else

}//end method count8

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 7

\* Given base and n that are both 1 or more, compute recursively (no loops) the value of base to the n power

\*/

**public** **class** PowerN

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

**int** base = 0, exponent = 0;

System.***out***.println("PowerN (Recursive)\n======================\nGiven base and n that are both 1 or more, this program will compute\nrecursively (no loops) the value of base to the n power, considering n is 1 or greater");

System.***out***.println("Enter the base of the power:");

base = sc.nextInt();

System.***out***.println("Enter the exponent of the power:");

exponent = sc.nextInt();

System.***out***.println("\nThe number of " + base + " to the power of " + exponent + " is: " + *powerN*(base, exponent));

}//end main

**public** **static** **int** powerN(**int** base, **int** n)

{

**if**(n == 1)

{

**return** base;

}//end if

**else**

{

**return**(base \* *powerN*(base, n - 1));

}//end else

}//end method powerN

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 8

\* Given a string, compute recursively (no loops) the number of lowercase 'x' chars in the string.

\*/

**public** **class** CountX

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

String userInput;

System.***out***.println("CountX (Recursive)\n======================\nGiven a string, this progam will compute recursively (no loops)\nthe number of lowercase 'x' chars in the string.");

System.***out***.println("\nEnter the string to be checked for 'x':");

userInput = sc.nextLine();

System.***out***.println("\nThe number of lowercase 'x' chars in " + userInput + " is: " + *countX*(userInput));

}//end main

**public** **static** **int** countX(String str)//another way to do it with less complication is to have the base case as "" then, find out if the char at 0 is 'x', then recursively call with the parameter substring(1);

{

**if**(str.length() == 0)

{

**return** 0;

}//end if

**else** **if**(str.charAt(str.length() - 1) != 'x')

{

**return**(*countX*(str.substring(0, str.length() - 1)));

}//end else if

**else**

{

**return**(1 + *countX*(str.substring(0, str.length() - 1)));

}//end else

}//end method countX

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 9

\* Given a string, compute recursively (no loops) the number of times lowercase "hi" appears in the string.

\*/

**public** **class** CountHi

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

String userInput;

System.***out***.println("CountX (Recursive)\n======================\nGiven a string, this program will compute recursively (no loops)\nthe number of times lowercase \"hi\" appears in the string.");

System.***out***.println("\nEnter the string to be checked for \"hi\":");

userInput = sc.nextLine();

System.***out***.println("\nThe number of lowercase 'x' chars in " + userInput + " is: " + *countHi*(userInput));

}//end main

**public** **static** **int** countHi(String str)

{

**if**(str.length() < 2)

{

**return** 0;

}//end if

**else** **if**(str.substring(0, 2).equals("hi"))

{

**return**(1 + *countHi*(str.substring(1)));

}//end else if

**else**

{

**return**(*countHi*(str.substring(1)));

}//end else

}//end method countHi

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 10

\* Given a string, compute recursively (no loops) a new string where all the lowercase 'x' chars have been changed to 'y' chars.

\*/

**public** **class** ChangeXY

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

String userInput;

System.***out***.println("ChangeXY (Recursive)\n======================\nGiven a string, compute recursively (no loops) a new string\nwhere all the lowercase 'x' chars have been changed to 'y' chars.");

System.***out***.println("\nEnter the string to be changed for 'x' is 'y':");

userInput = sc.nextLine();

System.***out***.println("\nThe new string with all of the x's switched to y's in " + userInput + " is: " + *changeXY*(userInput));

}//end main

**public** **static** String changeXY(String str)

{

**if**(str.length() == 0)

{

**return** "";

}//end if

**else** **if**(str.charAt(0) == 'x')

{

**return** "y" + *changeXY*(str.substring(1));

}//end else if

**else**

{

**return** str.charAt(0) + *changeXY*(str.substring(1));

}//end else

}//end method changeXY

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 11

\* Given a string, compute recursively (no loops) a new string where all appearances of "pi" have been replaced by "3.14".

\*/

**public** **class** ChangePi

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

String userInput;

System.***out***.println("ChangeXY (Recursive)\n======================\nGiven a string, compute recursively (no loops) a new string\nwhere all appearances of \"pi\" have been replaced by \"3.14\".");

System.***out***.println("\nEnter the string to be changed for \"pi\" is 3.14:");

userInput = sc.nextLine();

System.***out***.println("\nThe new string with all of the \"pi\"s switched to 3.14's in\n" + userInput + " is: " + *changePi*(userInput));

}//end main

**public** **static** String changePi(String str)

{

**if**(str.equals("") || str.length() < 2)

{

**return** str;

}//end if

**else** **if**(str.charAt(0) == 'p' && str.charAt(1) == 'i')

{

**return** "3.14" + *changePi*(str.substring(2));

}//end else if

**else**

{

**return** str.charAt(0) + *changePi*(str.substring(1));

}//end else

}//end method changePi

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 12

\* Given a string, compute recursively a new string where all the 'x' chars have been removed.

\*/

**public** **class** NoX

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

String userInput;

System.***out***.println("ChangeXY (Recursive)\n======================\nGiven a string, this program will compute recursively a new string\nwhere all the 'x' chars have been removed.");

System.***out***.println("\nEnter the string to have all 'x's removed:");

userInput = sc.nextLine();

System.***out***.println("\nThe new string with all of the 'x's removed in " + userInput + " is: " + *noX*(userInput));

}//end main

**public** **static** String noX(String str)

{

**if**(str.length() == 0)

{

**return** "";

}//end if

**else** **if**(str.charAt(0) == 'x')

{

**return** "" + *noX*(str.substring(1));

}//end else if

**else**

{

**return** str.charAt(0) + *noX*(str.substring(1));

}//end else

}//end method noX

}//end class

**import** java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 10, 2017

\* Last Edited: November 10, 2017

\* Unit 4 Recursion Practice 13

\* Given an array of ints, compute recursively if the array contains a 6. We'll use the convention of considering only the part of the array that begins at the given index.

\*/

**public** **class** Array6

{

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

{

//Variable Declarations and Initializations

Scanner sc = **new** Scanner(System.***in***);

**int** array[], elements = 0;

System.***out***.println("Array6 (Recursive)\n======================\nGiven an array of ints, this program will compute recursively if the array contains a 6. We'll use the\nconvention of considering only the part of the array that begins at the given index.");

System.***out***.println("\nEnter the number of elements you want in the array:");

elements = sc.nextInt();

array = **new** **int**[elements];

//to populate array

**for**(**int** i = 0; i < elements; i++)

{

System.***out***.println("Enter a number at element " + (i + 1) + " in the array:");

array[i] = sc.nextInt();

}//end loop

System.***out***.println("\nAre there any 6's in the array?: "+ *array6*(array, 0));

}//end main

**public** **static** **boolean** array6(**int**[] nums, **int** index)

{

**if**(index == nums.length)

{

**return** **false**;

}//end if

**else** **if**(nums[index] == 6)

{

**return** **true**;

}//end else if

**else**

{

**return**(*array6*(nums, index + 1));

}//end else

}//end method array6

}//end class

import java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 13, 2017

\* Last Edited: November 13, 2017

\* Unit 4 Recursion Practice 14

\* Given an array of ints, compute recursively the number of times that the value 11 appears in the array.

\*/

public class Array11

{

public static void main(String[] args)

{

//Variable Declarations and Initializations

Scanner sc = new Scanner(System.in);

int array[], elements = 0;

System.out.println("Array11 (Recursive)\n======================\nGiven an array of ints, this program will compute recursively the number of times that the value 11 appears in the array.");

System.out.println("\nEnter the number of elements you want in the array:");

elements = sc.nextInt();

array = new int[elements];

//to populate array

for(int i = 0; i < elements; i++)

{

System.out.println("Enter a number at element " + (i + 1) + " in the array:");

array[i] = sc.nextInt();

}//end loop

System.out.println("\nHow many 11's in the array?: "+ array11(array, 0));

}//end main

public static int array11(int[] nums, int index)

{

if(index == nums.length)

{

return 0;

}//end if

if(nums[index] == 11)

{

return 1 + array11(nums, index + 1);

}//end if

else

{

return array11(nums, index + 1);

}//end else

}//end method array11

}//end class

import java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 13, 2017

\* Last Edited: November 20, 2017

\* Unit 4 Recursion Practice 15

\* Given an array of ints, compute recursively if the array contains somewhere a value followed in the array by that value times 10

\*/

public class Array220

{

public static void main(String[] args)

{

//Variable Declarations and Initializations

Scanner sc = new Scanner(System.in);

int array[], elements = 0;

System.out.println("Array220 (Recursive)\n======================\nGiven an array of ints, this program will compute recursively if the array contains somewhere a value followed in the array by that value times 10");

System.out.println("\nEnter the number of elements you want in the array:");

elements = sc.nextInt();

array = new int[elements];

//to populate array

for(int i = 0; i < elements; i++)

{

System.out.println("Enter a number at element " + (i + 1) + " in the array:");

array[i] = sc.nextInt();

}//end loop

System.out.println("\nAre there any values followed by another value ten times its size?: "+ array220(array, 0));

}//end main

public static boolean array220(int[] nums, int index)

{

if(index >= nums.length - 1)

{

return false;

}//end if

else if(nums[index + 1] == nums[index] \* 10)

{

return true;

}//else if

else

{

return array220(nums, index + 1);

}//end else

}//end method array220

}//end class

import java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 20, 2017

\* Last Edited: November 20, 2017

\* Unit 4 Recursion Practice 16

\* Given a string, compute recursively a new string where all the adjacent chars are now separated by a "\*".

\*/

public class AllStar

{

public static void main(String[] args)

{

//Variable Declarations and Initializations

Scanner sc = new Scanner(System.in);

String userInput;

System.out.println("AllStar (Recursive)\n======================\nGiven a string, compute recursively a new string where all the adjacent chars are now separated by a \"\*\".");

System.out.println("\nEnter the string to be changed for all stars in between:");

userInput = sc.nextLine();

System.out.println("\nThe new string with all of the \*'s added in " + userInput + " is: " + allStar(userInput));

}//end main

public static String allStar(String str)

{

if(str.length() == 0)

{

return str;

}//end if

else if(str.length() >= 2)

{

return str.charAt(0) + "\*" + allStar(str.substring(1));

}//end else if

else

{

return str.charAt(0) + allStar(str.substring(1));

}//end else

}//end method allStar

}//end class

import java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 28, 2017

\* Last Edited: November 28, 2017

\* Unit 4 Recursion Practice 17

\* Given a string, compute recursively a new string where identical chars that are adjacent in the original string are separated from each other by a "\*".

\*/

public class PairStar

{

public static void main(String[] args)

{

//Variable Declarations and Initializations

Scanner sc = new Scanner(System.in);

String userInput;

System.out.println("PairStar (Recursive)\n======================\nGiven a string, this program will compute recursively a new string\nwhere identical chars that are adjacent in the original string are separated from each other by a \"\*\".");

System.out.println("\nEnter the string to be changed for paired stars in between:");

userInput = sc.nextLine();

System.out.println("\nThe new string with all of the \*'s added in\nbetween paired letters is" + userInput + " is: " + pairStar(userInput));

}//end main

public static String pairStar(String str)

{

if(str.length() <= 1)

{

return str;

}//end if

if(str.charAt(0) == str.charAt(1))

{

return str.charAt(0) + "\*" + pairStar(str.substring(1));

}//end if

else

{

return str.charAt(0) + pairStar(str.substring(1));

}//end else

}//end method pairStar

}//end class

import java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 28, 2017

\* Last Edited: November 28, 2017

\* Unit 4 Recursion Practice 18

\* Given a string, compute recursively a new string where all the lowercase 'x' chars have been moved to the end of the string.

\*/

public class EndX

{

public static void main(String[] args)

{

//Variable Declarations and Initializations

Scanner sc = new Scanner(System.in);

String userInput;

System.out.println("EndX (Recursive)\n======================\nGiven a string, this program will compute recursively a new string\nwhere all the lowercase 'x' chars have been moved to the end of the string..");

System.out.println("\nEnter the string to the lowercase x's moved to the end:");

userInput = sc.nextLine();

System.out.println("\nThe new string with all of the x's moved in " + userInput + " is: " + endX(userInput));

}//end main

public static String endX(String str)

{

if(str.length() == 0)

{

return str;

}//end if

if(str.charAt(0) == 'x')

{

return endX(str.substring(1)) + "x";

}//end if

else

{

return str.charAt(0) + endX(str.substring(1));

}//end else

}//end method endX

}//end class

import java.util.Scanner;

/\*

\* Justin Mendes

\* Created: November 30, 2017

\* Last Edited: November 30, 2017

\* Unit 4 Recursion Practice 19

\* We'll say that a "pair" in a string is two instances of a char separated by a char. So "AxA" the A's make a pair.

\* Pair's can overlap, so "AxAxA" contains 3 pairs -- 2 for A and 1 for x. Recursively compute the number of pairs in the given string.

\*/

public class CountPairs

{

public static void main(String[] args)

{

//Variable Declarations and Initializations

Scanner sc = new Scanner(System.in);

String userInput;

System.out.println("CountPair (Recursive)\n======================\nWe'll say that a \"pair\" in a string is two instances of a char separated by a char. So \"AxA\"\n"

+ "the A's make a pair. Pair's can overlap, so \"AxAxA\" contains 3 pairs -- 2 for A and 1 for x.\n This program will recursively compute the number of pairs in the given string.");

System.out.println("\nEnter the string to the lowercase x's moved to the end:");

userInput = sc.nextLine();

System.out.println("\nThe new string with all of the x's moved in " + userInput + " is: " + endX(userInput));

}//end main

public static String endX(String str)

{

if(str.length() == 0)

{

return str;

}//end if

if(str.charAt(0) == 'x')

{

return endX(str.substring(1)) + "x";

}//end if

else

{

return str.charAt(0) + endX(str.substring(1));

}//end else

}//end method endX

}//end class