**Loops Classwork**

**While loop:**

1. Type in the following code, and get it to compile. This assignment will help you learn how to make a loop, so that you can repeat a section of code over and over again!

import java.util.Scanner;

public class EnterPIN

{

public static void main( String[] args )

{

Scanner keyboard = new Scanner(System.in);

int pin = 12345;

System.out.println("WELCOME TO THE BANK OF MITCHELL.");

System.out.print("ENTER YOUR PIN: ");

int entry = keyboard.nextInt();

while ( entry != pin )

{

System.out.println("\nINCORRECT PIN. TRY AGAIN.");

System.out.print("ENTER YOUR PIN: ");

entry = keyboard.nextInt();

}

System.out.println("\nPIN ACCEPTED. YOU NOW HAVE ACCESS TO YOUR ACCOUNT.");

}

}

1. Write a program that plays an incredibly stupid number-guessing game. The user will get one try to guess the secret number. Tell them if they got it right or wrong, and if they got it wrong, display what the secret number was.

You must store the secret number in a variable, and use that variable throughout. The secret number itself must **not** appear in the program at all, except in the one line where you store it into a variable.

b) Modify your number-guessing game to actually pick a random number from 1 to 10 and have the user try to guess that. Tell them if they get it right or wrong, and if they get it wrong, show them what the random number was.

They will still only get one try.

c) Modify the game so that they can guess **until** they get it right. That means it will keep looping as long as the guess is different from the secret number. Use a while loop.

1. Given a string and a non-negative int n, we'll say that the front of the string is the first 3 chars, or whatever is there if the string is less than length 3. Return n copies of the front;

frontTimes("Chocolate", 2) → "ChoCho"  
frontTimes("Chocolate", 3) → "ChoChoCho"  
frontTimes("Abc", 3) → "AbcAbcAbc"

**For Loop**

1. Given a number, return that many “\*” as a string.

printStar(5) → “\*\*\*\*\*”  
printStar(13) → “\*\*\*\*\*\*\*\*\*\*\*\*\*”  
printStar(1) → “\*”

1. Given a string, return true if the first instance of "x" in the string is immediately followed by another "x".

doubleX("axxbb") → true  
doubleX("axaxax") → false  
doubleX("xxxxx") → true

1. Given a string, return a string made of the chars at indexes 0,1, 4,5, 8,9 ... so "kittens" yields "kien".

altPairs("kitten") → "kien"  
altPairs("Chocolate") → "Chole"  
altPairs("CodingHorror") → "Congrr"

1. Return the number of times that the string "hi" appears anywhere in the given string.

countHi("abc hi ho") → 1  
countHi("ABChi hi") → 2  
countHi("hihi") → 2

1. Return true if the string "cat" and "dog" appear the same number of times in the given string.

catDog("catdog") → true  
catDog("catcat") → false  
catDog("1cat1cadodog") → true

**Nested Loops**

1. Based on question 4, given a number, return a pyramid of stars with that many “\*” as its base.

printPyramid(5) →  
\*  
\*\*  
\*\*\*  
\*\*\*\*  
\*\*\*\*\*  
printPyramid(10) →  
\*  
\*\*  
\*\*\*  
\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*\*  
\*\*\*\*\*\*\*  
\*\*\*\*\*\*\*\*  
\*\*\*\*\*\*\*\*\*  
\*\*\*\*\*\*\*\*\*\*  
printPyramid(3) →  
\*  
\*\*  
\*\*\*

1. Given a string, return the sum of the digits 0-9 that appear in the string, ignoring all other characters. Return 0 if there are no digits in the string.

sumDigits("aa1bc2d3") → 6  
sumDigits("aa11b33") → 8  
sumDigits("Chocolate") → 0

1. Given two strings, **base** and **remove**, return a version of the base string where all instances of the remove string have been removed (not case sensitive). You may assume that the remove string is length 1 or more. Remove only non-overlapping instances, so with "xxx" removing "xx" leaves "x".

withoutString("Hello there", "llo") → "He there"  
withoutString("Hello there", "e") → "Hllo thr"  
withoutString("Hello there", "x") → "Hello there"