VOLO

HW03

Problem 01

```
import java.util.Random;
public class Problem1 {
    public static boolean isLeapYear(int year)
    { if (year % 4 == 0)
        if (year % 100 == 0)
            return year % 400 == 0;
        else return true;
      else return false;
    }
public static void main(String[] args){
  Random rand = new Random();
  int count = 0;
  int years;
  for(int i=0; i<3; i++){
    years = 1901 + rand.nextInt(100);
    if (isLeapYear(years)) count++;
    System.out.println(years);
  }
 if(count>=2) System.out.println("Most were leap years.");
 else System.out.println("Most were not leap years");
 }
}
```

Proble 02

```
import java.util.Scanner;

public class Problem2{

    Public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Your Value");
        String input1 = scanner.nextLine();
        int input2 = sc.nextInt();
        sc.close();

        boolean ispalinS = CsUtils.isPalindrome(input1);
        boolean isPalinI = CsUtils.isPalindrome(input2);

        if(isPalindros) System.out.println("Is Palindr");
        else System.out.println("IsNotPalindr");
        If (isPalinI) Syst.out.Println("Is Palin");
        else system.out.println("Is not Palind");
    }
class CsUtils { Pul
    Public static int reverse(int value){
    int reversed = 0, absV = Math.abs(value);
    Boolean negative = value < 0;
    while (absV > 0) {
        int digit = absV % 10;
        reversed = reversed * 10 + digit,
```

```
Public static boolean isPalindrome(int n){
    if (n<0){
      return false;
    }
    return n == reverse(n);
  }
public class Palindrome{
  public static boolean isPalindrome(String s){
    int left =0;
    int right =s.length()-1;
    while(left<right){
      if(s.charAt(left)!=s.charAt(right)) return false;
      left++;
      right--;
    }
    return true;
  }
}
```

Ruben0}

```
import java.util.Scanner;
public class Prob13{
    public static void main(String[] args){
        Scanner scanner = new Scanner(System.in);
        int n = scanner.nextInt();
        scanner.
```

```
String[] firstNames = new String[n];
String[] lastNames = new String[n];
double[] scores = new double[n];

for (int i=0; i<n; i++) {
System.out.println("Student " + (i+1) + " details");
firstNames[i] = sc.next();
lastNames[i] = sc.next();
scores[i] = sc.nextDouble();
}
double max = scores[0]; int index;
for (int i=1; i<n; i++)
if (scores[i] > max){ max = scores[i]; index = i;}
System.out.println("Highest score " + lastNames[index]);
int sum = 0
for (int i=1; i<n; i++)
sum += scores[i];
System.out.println("Total score " + sum);
double min = scores[0];
for(int i=1; i<n; i++)
if(scores[i] < min) min = scores[i];
System.out.println("Lowest score " + min);
}sc.close();
```

Problem 04

```
import java.util.Arrays;
import java.util.Scanner;
public class PrimeFact {

    public static int[] findPrimeFactors(int n, int fac, int[] result)
    {   if (n <= 1) return result;
        if (n % fac == 0) {
           result = appendToArray(result, factor);
           return findPrimeFactors(n/fact, fac, result);
        else return findPrimeFactors(n, fac+1, result);
    }

    public static int[] appToArray(int[] arr, int value)
     {  int[] newArray = Arrays.copyOf(arr, arr.length + 1);
        newArray[newArray.length - 1] = value;
        return newArray;
    }

  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt();
  int[] primeFactors = findPrimeFactors(n, 2, new int[0])
    System.out.println(n + Arrays.toString(primeFactors));
    sc.close();
  }
```

Problem: 05

```
import java.util.Arrays;

public class prob5 {

    public static int [][] snake order (int n)

        int[][] arr = new int[n][n];
        int val = n*n;
        for (int col = 0; col < n; col++) {
            if (col % 2 == 0)
                for (int row = 0; row < n; row++)
                    arr[row][col] = val--;
            else for (int row = n-1; row >= 0; row--)
                arr[row][col] = val--;
        return arr;
    }
    public static void main (String [] args) {
        int n = 3;
        int [][] = snakeorder (n)
        for (int [] row: arr)
            System.out.println(Arrays.toString(row));
        }
    }
}
```

Rust 6.

```
use std::io::{self, Read};

fn main() {
    let mut input = String::new();
    io::stdin().read_to_string(&mut input).unwrap();
    let n: usize = input.trim().parse().unwrap();

    let mut matrix = vec![vec![0; n]; n];
    let mut num = 1;
    let mut top = 0;
    let mut bottom = n - 1;
    let mut left = 0;
    let mut right = n - 1;

    while num <= n * n {
        for col in left..=right {
            matrix[top][col] = num;
            num += 1;
        }
        top += 1;

        for row in top..=bottom {
            matrix[row][right] = num;
            num += 1;
        }
        if right == 0 { break; }
        right -= 1;

        if top <= bottom {
            for col in (left..=right).rev() {
                matrix[bottom][col] = num;
                num += 1;
            }
            if bottom == 0 { break; }
            bottom -= 1;
        }

        if left <= right {
            for row in (top..=bottom).rev() {
                matrix[row][left] = num;
                num += 1;
            }
            left += 1;
        }
    }

    for row in matrix {
        let line: Vec<String> = row.iter().map(|x| x.to_string()).collect();
        println!("{}", line.join(" "));
    }
}
```