```
import java.util.Scanner;
import java.util.ArrayList;
import java.util.List;


public class TKim_1_NumberTransformation_201920 {
    public static void main(String[] args) {
        //Declaration
        Scanner input = new Scanner(System.in);
        List<Character> characterDataSet = new ArrayList<>();
        List<String> stringDataSet = new ArrayList<>();
        List<String> parsedFinalValue = new ArrayList<>();
        StringBuilder stringBuilder = new StringBuilder();
        List<String> reversedStringDataSet = new ArrayList<>();

        //Initiation
        System.out.println("Welcome to ACSL Number Transformation Algorithm!");
        System.out.println(" ");
        System.out.println(" ");
        System.out.println("Enter your N value:");
        System.out.println("ex. 296351");

        String N = input.next();

        System.out.print(" ");
        System.out.println("Enter your P value:");
        System.out.println("ex. 5");

        int P = input.nextInt();


        //Algorithm
        //Parse the data sets into one array list
        char[] NToArray = N.toCharArray();

        for (int i = 0; i < NToArray.length; i++) {
            characterDataSet.add(NToArray[i]);
        }

        //Convert Characters into string
        for (int x = 0; x < characterDataSet.size(); x++) {
            stringDataSet.add(characterDataSet.get(x).toString());
        }

        //Create a reversed data set
        for(int y = (stringDataSet.size()-1); y >= 0; y--) {
            reversedStringDataSet.add(stringDataSet.get(y));
        }

        System.out.println(reversedStringDataSet);
```

```
        //Find Pivotal Point
        String pivotalStringPoint = reversedStringDataSet.get(P-1);
        int pivotalPoint = Integer.parseInt(pivotalStringPoint);

        System.out.println("Pivotal Point: " + pivotalPoint);

        //Run the left side of the digit

        for (int leftSideDigit = 0; leftSideDigit < stringDataSet.size() - P;
leftSideDigit++) {
            String computedValueL =
leftSideOfNumber(stringDataSet.get(leftSideDigit), pivotalPoint);
            parsedFinalValue.add(computedValueL);
            System.out.println(computedValueL);
        }
        parsedFinalValue.add(pivotalStringPoint);

        //Run the right side of the digit
        for (int rightSideDigit = parsedFinalValue.size(); rightSideDigit <
stringDataSet.size(); rightSideDigit ++) {
            String computedValueR =
rightSideOfNumber(stringDataSet.get(rightSideDigit), pivotalPoint);
            parsedFinalValue.add(computedValueR);
        }


        //Combine the numbers to form one number
        for (int singleString = 0; singleString < parsedFinalValue.size();
singleString++) {
            stringBuilder.append(parsedFinalValue.get(singleString));
        }

        //Final Answer
        String finalOutput = stringBuilder.toString();

        System.out.println(finalOutput);

    }


    //Operations for left side of the value
    public static String leftSideOfNumber(String given, int constantAtPthDigit) {
        //Do preliminary conversions
        int givenInteger = Integer.parseInt(given);

        //apply actual math
        int finalTempValue = givenInteger + constantAtPthDigit;

        // test if sum is greater than 10
        if (finalTempValue >= 10) {
```

```
            int intFinalTempValue = finalTempValue - 10;
            String finalValue = Integer.toString(intFinalTempValue);
            return finalValue;

        } else {
            String finalValue = Integer.toString(finalTempValue);
            return finalValue;
        }

    }

    //Operations for right side of the value
    public static String rightSideOfNumber(String given, int constantAtPthDigit) {
        //Do preliminary conversions
        int givenInteger = Integer.parseInt(given);
        int finalIntegerValue = Math.abs(givenInteger - constantAtPthDigit);
        String finalValue = Integer.toString(finalIntegerValue);
        return finalValue;
    }
}
```