Mettl 2B

1. Generate series and find Nth element

```
public class GenerateSeriesFindN {
   public int generateSeriesFindN(int input1,int input2,int input3,int input4){
      int gap1 = (input2 - input1);
      int gap2 = (input3 - input2);
      int output = input1;
      for (int i = 1; i < input4; i++) {
         if (i % 2 == 1)
            output += gap1;
         else
            output += gap2;
      }
      return output;
   }
}
```

---

2. Find result after alternate add_sub on N

```
public class AlternateAddSub {
   public int AddSub(int input1,int input2){
      int result = input1;
      int check = 0;
      if (input2 == 1) check = 1;
      for (int i = input1 - 1, j = 0; i >= 1; i--, j++) {
         if (j % 2 == check) result += i;
```

```
            else result -= i;
        }
        return result;
    }
}
```

---

3. Find Password (stable unstable)

```
import java.io.*;
import  java.util.*;
public class FindPasswordStableUnstable {
    public int findPassword(int input1,int input2,int input3,int input4,int input5){
        int[] nums = {input1, input2, input3, input4, input5};
        int stable = 0, unstable = 0;
        for (int num: nums){
            if (isStable(num)) stable += num;
            else unstable += num;
        }
        return stable - unstable;
    }
    public static boolean isStable(int num) {
        boolean isStable = true;
        int[] freq = new int[10];
        String numStr = String.valueOf(num);
        for (int i = 0; i < numStr.length(); i++) {
            freq[Integer.parseInt(String.valueOf(numStr.charAt(i)))]++;
        }
```

```
        int primFreq = 0;
        for (int i = 0; i < 10; i++) {
            if (freq[i] > 0) {
                primFreq = freq[i];
                break;
            }
        }
        for (int i = 0; i < 10; i++) {
            if (freq[i] != 0 && freq[i] != primFreq) {
                isStable = false;
                break;
            }
        }
        return isStable;
    }
}
```

---

4. Calculate sum of non-prime index values

```
public class SumOfNonPrimeIndexes {
    public int sumOfNonPrimeIndexes(int[] input1, int input2) {
        int sum = 0;
        for (int i = 0; i <= Math.sqrt(input2); i++)
            if (!isPrime(i))
                sum += input1[i];
        return sum;
```

```
    }
    public static boolean isPrime(int input1) {
        if (1 == input1 || 0 == input1)
            return false;
        for (int i = 2; i < input1; i++) {
            if (i == input1)
                continue;
            if (input1 % i == 0) {
                return false;
            }
        }
        return true;
    }
}
```

---

5. Find the one digit to be removed to form palindrome

```
import java.io.*;
import java.util.*;
public class Remove1DigitForPalindrome {
    public int digitRemove_Palin(int input1){
        StringBuilder num = new StringBuilder(String.valueOf(input1));

        for (int i = 0; i < num.length(); i++) {
            if (isPalindrome(num.toString())) return -1;

            char removed = num.charAt(i);
```

```
            String newNum = num.deleteCharAt(i).toString();


            if (isPalindrome(newNum)) {
                return Integer.parseInt(String.valueOf(removed));
            } else {
                num.insert(i, removed);
            }
        }
        return -1;
    }
    public static boolean isPalindrome(String str) {
        str = str.toLowerCase();
        int len = str.length();
        boolean isPalindrome = true;
        int range = len / 2;

        if (len % 2 == 0) range--;

        for (int i = 0; i <= range; i++) {
            if (str.charAt(i) != str.charAt(len - i - 1)) isPalindrome = false;
        }
        return isPalindrome;
    }
}
```

6. The “Nambiar Number” Generator

```
import java.io.*;
import java.util.*;
public class NambiarNumberGenerator {
   public int nnGenerator(String input1){
      String mobileNo = input1;
      StringBuilder numbiarNo = new StringBuilder();
      for (int i = 0; i < mobileNo.length(); i++) {
         int digit = Integer.parseInt(String.valueOf(mobileNo.charAt(i)));
         int evenOdd = digit % 2 == 0 ? 0 : 1;
         int sum = digit;
         int j = i + 1;
         if (j == mobileNo.length()) {
            numbiarNo.append(digit);
            break;
         }
         while (true) {
            sum += Integer.parseInt(String.valueOf(mobileNo.charAt(j++)));
            if (sum % 2 != evenOdd || j >= mobileNo.length()) {
               numbiarNo.append(sum);
               i = j - 1;
               break;
            }
         }
      }
      return Integer.parseInt(numbiarNo.toString());
   }
```

```
}
```

7. User ID Generation

```
import java.io.*;
import java.util.*;
public class UserIDGeneration {
   public String userIdGeneration(String input1,String input2,int input3,int input4){
      String firstName = input1,lastName = input2, longerName, smallerName;
      int pin = input3,N = input4;
      StringBuilder userId = new StringBuilder();
      if (firstName.length() > lastName.length()) {
         longerName = firstName;
         smallerName = lastName;
      } else if (firstName.length() < lastName.length())  {
         longerName = lastName;
         smallerName = firstName;
      } else {
         if (firstName.compareTo(lastName) < 1 ) {
            longerName = lastName;
            smallerName = firstName;
         } else {
            longerName = firstName;
            smallerName = lastName;
         }
      }
```

```
        userId.append(smallerName.charAt(smallerName.length() - 1));
        userId.append(longerName);
        for (int i = 0; i < userId.length(); i++) {
            if (Character.isUpperCase(userId.charAt(i)))
                userId.setCharAt(i, Character.toLowerCase(userId.charAt(i)));
            else
                userId.setCharAt(i, Character.toUpperCase(userId.charAt(i)));
        }
        userId.append(String.valueOf(pin).charAt(N - 1));
        userId.append(String.valueOf(pin).charAt(String.valueOf(pin).length() - N));

        return userId.toString();
    }
}
```

---

8. Message controlled Robot movement

```
import java.io.*;
import java.util.*;
public class MsgControlledRobot {
    public String moveRobot(int input1,int input2,String input3,String input4){
        int X = input1, Y = input2;
        String currentPos = input3, msg = input4;

        int currX = Integer.parseInt(currentPos.split("-")[0]);
        int currY = Integer.parseInt(currentPos.split("-")[1]);
```

```
String currD = currentPos.split("-")[2];
String[] instructions = msg.split(" ");
StringBuilder output = new StringBuilder();

for (int i = 0; i < instructions.length; i++) {
   if (instructions[i].equals("M")) {
      if (currD.equals("E") && (currX + 1 > X )) {
         output.append("-ER");
         break;
      }
      if (currD.equals("W") && (currX - 1 < 0 )) {
         output.append("-ER");
         break;
      }
      if (currD.equals("N") && (currY + 1 > Y )) {
         output.append("-ER");
         break;
      }
      if (currD.equals("S") && (currY - 1 < 0 )) {
         output.append("-ER");
         break;
      }

      if (currD.equals("E")) currX++;
      else if (currD.equals("W")) currX--;
      else if (currD.equals("N")) currY++;
      else if (currD.equals("S")) currY--;
```

```
        } else {
            if (currD.equals("E") && instructions[i].equals("L"))
                currD = "N";
            else if (currD.equals("E") && instructions[i].equals("R"))
                currD = "S";
            else if (currD.equals("W") && instructions[i].equals("L"))
                currD = "S";
            else if (currD.equals("W") && instructions[i].equals("R"))
                currD = "N";
            else if (currD.equals("N") && instructions[i].equals("L"))
                currD = "W";
            else if (currD.equals("N") && instructions[i].equals("R"))
                currD = "E";
            else if (currD.equals("S") && instructions[i].equals("L"))
                currD = "E";
            else if (currD.equals("S") && instructions[i].equals("R"))
                currD = "W";
        }

        output.delete(0, output.length());
        output.append(currX + "-" + currY + "-" + currD);
      }
      return output.toString();
   }
}
```