Praktikum 04 - 26 September 2025: *Single Array*

Segera submit pekerjaan anda setiap satu soal selesai dan segera minta diperiksa asdos setiap satu nomor selesai. Setiap program perlu disertai dengan unit test. Baik disediakan soal, atau bila tidak disediakan soal, maka peserta harus membuat-nya sendiri

1. Buat lah method yang bersignature:

   ```
   public static int jumlahArrayGanjil(int[] data);
   ```

   Method tersebut menerima sebuah array, menjumlahkan elemen ganjil pada array tersebut dan kemudian me-return hasilnya. Misalnya

   ```
   jumlahArrayGanjil(new int[]{1,2,3,4,5});
   ```

   akan memberikan output: ***9***.

   Buat juga unit test-nya.

2. Dalam sistem informasi yang baik, perubahan data sebaiknya selalu tercatat. Kelak rekan-rekan akan mempelajari konsep lebih luasnya dalam istilah yang disebut ***audit trails***. Kali ini kita buat bagian kecilnya dan versi sederhana-nya. Buat lah method yang bersignature:

   ```
   public static int[] editArray(int[] data, int index, int datum);
   ```

   Method tersebut menerima sebuah array dengan parameter ***data***, index array yang mau diubah dengan parameter ***index***, dan nilai baru pada array di index yang diminta dengan parameter datum. Jadi method ini akan mengembalikan array baru dengan element ***arraybaru[index]*** adalah bernilai ***datum***; Pastikan array lama tidak berubah.

   Berikut ini contoh *unit test*-nya, silahkan tambahkan:

```
import static org.junit.Assert.*;
import org.junit.Test;

public class EditArrayTest{

    @Test
    public void editTest01(){
        String[] data = {"Amir","Budi","Ceci","Dudi"};
        String[] out1 = {"Amir","Baba","Ceci","Dudi"};

        // output sesuai:
        assertArrayEquals(out1,
                          EditArray.editArray(data, 1,  "Baba"));

        // Data awal tidak berubah setelah pemanggilan method:
        assertArrayEquals(new String[]{"Amir","Budi","Ceci","Dudi"},data);
    }
}
```

3. Buat lah method yang bersignature:

```
public static int modusRentang4(int[] data);
```

Method ini membaca array bilangan bulat yang elemen-nya hanya bilangan bulat pada rentang 0-4, kemudian mengembalikan elemen yang kemunculannya paling banyak. Bila ada beberapa elemen yang jumlah kemunculannya sama, maka kembalikan semuanya. Namun bila elemen dari array data tersebut tidak memenuhi kriteria pada rentang 4, maka method harus mengembalikan array empty, array kosong. Peserta belum diperkenankan menggunakan ArrayList.

Contoh unit test:

```
@Test
public void modus01(){
    int[] data = {0,1,2,2,2,3,3,3,4,4};
    assertArrayEquals(new int[]{2,3}, EditArray.modusRentang4(data));
}

@Test
public void modus02(){
    int[] data = {0,1,1,1,1,1,2,2,2,3,3,3,4,4};
    assertArrayEquals(new int[]{1}, EditArray.modusRentang4(data));
}

@Test
public void modus03(){
    int[] data = {0,1,1,1,1,1,-1,2,2,3,3,3,4,4};
    assertArrayEquals(new int[]{}, EditArray.modusRentang4(data));
}

@Test
public void modus04(){
    int[] data = {0,1,1,1,1,1,5,2,2,3,3,3,4,4};
    assertArrayEquals(new int[]{}, EditArray.modusRentang4(data));
}

@Test
public void modus05(){
    int[] data = {3,3,0,1,2,1,4,1,1,2,1,3,1,3,4,3,3,4};
    assertArrayEquals(new int[]{1,3}, EditArray.modusRentang4(data));
}
```

4. Implementasikan algoritma counting sort, untuk input dalam rentang 0-4. Perhatikan bahwa method tersebut juga harus memeriksa terlebih dahulu apakah input data yang diberikan memang benar valid dalam rentang 0-4. Bila input data tidak valid, kembalikan array kosong.

```
public static int[] countingSort(int[] data);
```

Lengkapi dengan unit test.

Rubrik Penilaian dalam skala 4 berlaku untuk semua soal: (Syarat bisa lanjut praktikum selanjutnya adalah nilai 2 dari 4)

- ☐ Nilai 4: Bila mengerjakan soal dengan sesuai, benar dan teruji oleh *unit test*
- ☐ Nilai 3: Mengerjakan tapi ada yang salah, ada *unit test* yang *failed*
- ☐ Nilai 2: Mengerjakan tapi tidak bisa diuji dengan *unit test*, untuk *unit test failed* semua
- ☐ Nilai 1: Program sudah dibuat, tapi masih tidak bisa di-*compile* atau belum lengkap