DDP2
Latihan Rekursif – Part I

Membaca Program Rekursif

```
public int misteri (int n) {
   if (n == 1) {
      return 1;
   } else {
      return misteri(n - 1) + n;
   }
}
```

Evaluasi dan lakukan tracing untuk **`misteri (5)`** !

Membuat Program Rekursif

1. Misalkan ada kelinci-kelinci yang sedang berbaris, dan diberi nomor secara berurutan dimulai dari 1, 2, 3, … ada informasi berikut:
   - Kelinci yang berada pada posisi ganjil (1, 3, …) mempunyai 2 buah telinga
   - Kelinci yang berada pada posisi genap (2, 4, …) mempunyai 3 buah telinga

Lengkapi fungsi **`int bunnyEars (int N)`** yang mengembalikan banyaknya total telinga kelinci dimulai dari kelinci pada posisi **1** hingga kelinci pada posisi **N** ! (N ≥ 0)

```
public int bunnyEars(int N) {

   if (_____________) { //basis, kondisi masalah paling simple
     return 0;
   } else {
     //rekurens



   }
}
```

2. Lengkapilah static method **reverse** berikut. Method **reverse** digunakan untuk "membalikkan" urutan karakter pada sebuah string masukan (returned value). Gunakan cara berpikir **rekursif**.

```
public static String reverse (String str) {

    if (str.length() == 0) {   //basis, kondisi masalah paling simple

       return "";

    } else {

       //Head
       char   head = _____________________________;
       String tail = _____________________________;

       return ___________________________________;    //rekurens
    }
```

```
}
```

3. Buatlah sebuah method **`int jumlahA(String str)`** yang dapat menghitung jumlah karakter **'a'** pada parameter input **`str`**. Gunakan cara berpikir rekursif! Asumsi: **`str`** adalah string *lowercase*.
Contoh: **`jumlahA("abcdaanda")`** mengembalikan **4**.

```
public int jumlahA(String str) {



}
```

4. Buatlah sebuah method **`int sumDigits(int number)`**. Fungsi ini akan mengembalikan **jumlah semua digit** yang ada pada **number** secara rekursif.

sumDigits(126) mengembalikan 9, dan sumDigits(2344) mengembalikan 13.

Hint: mod (%) dengan 10 akan menghasilkan digit paling kanan (3234 % 10 = 4), dan divide (/) dengan 10 akan menghapus digit paling kanan (3234 / 10 = 323).

5. Buatlah sebuah method **`int jumlahGenap(int N)`** yang mengembalikan banyaknya **digit genap** dari bilangan bulat **N**, asumsi **N > 0**.
Contoh:
`jumlahGenap(1234)` akan mengembalikan 2
`jumlahGenap(20454)` akan mengembalikan 4

6. Implementasikan method **`int jumlah (int[] arr)`** yang menghitung **jumlah elemen** pada array masukan secara rekursif.

```
int [] arr = {2,4,5,6};
```
`jumlah (arr)` akan mengembalikan **17**.

Hint: Anda boleh mengimplementasikan **Recursive Helper Method**:

```
int jumlahRecc (int start, int[] arr)
```

yang melakukan proses penjumlahan elemen secara rekursif dimulai dari indeks **start** hingga akhir array. Fungsi utama hanya perlu memanggil **Recursive Helper Method** tersebut dengan:

```
int jumlah (int[] arr) {
      return jumlahRecc(0, arr);
}
```

7. Implementasikan method **`void printArray (int[] arr)`** yang mencetak ke layar seluruh elemen dari sebuah array of integer dari **kiri ke kanan** secara rekursif. Gunakan cara yang serupa dengan soal nomor 6.

```
int [] arr = {2,4,5,6};
```
`printArray (arr)` akan mencetak:
2
4
5
6

8. Implementasikan method **`int numGenap (int[] arr)`** yang menghitung **banyaknya elemen genap** pada array masukan secara rekursif. Gunakan cara yang serupa dengan soal nomor 6.

```
int [] arr = {2,4,5,6};
```
`numGenap (arr)` akan mengembalikan **3**.

9. Implementasikan method **`void printArray (int[] arr)`** yang mencetak ke layar seluruh elemen dari sebuah array of integer dari **kanan ke kiri** secara rekursif. Anda boleh menggunakan **Recursive Helper Method**.

```
int [] arr = {2,4,5,6};
```
`printArray (arr)` akan mencetak:
6
5
4
2

10. Implementasikan method **`String changeXY(String str)`** yang mengembalikan sebuah string baru, dimana karakter "x" pada string masukan diganti dengan karakter "y". Gunakan paradigma rekursif.

`changeXY("decodex")` mengembalikan **"decodey"**
`changeXY("xyxy")` mengembalikan **"yyyy"**

11. Implementasikan method **`int findMax (int[] arr)`** yang mengembalikan elemen bernilai paling maksimum dari sebuah array of integer secara rekursif. Anda boleh menggunakan **Recursive Helper Method**.

```
int [] arr = {2,4,5,6,7,3,4};
```
`findMax (arr)` akan mengembalikan 7

12. **PALINDROM**
Palindrom adalah sebuah kumpulan string yang dapat dibaca dengan sama, baik dimulai dari depan, maupun dimulai dari belakang.

Contoh:

KATAK
KASURNABABANRUSAK

Implementasikan method **`boolean isPalindrom (String str)`** yang memeriksa apakah string masukan palindrom atau bukan. Jika palindrom, maka method akan mengembalikan **true**. Jika bukan palindrom, maka method akan mengembalikan **false**. Pada soal kali ini, implementasikan **DENGAN RECURSIVE CALL**.
Asumsi: String masukan hanya terdiri dari karakter huruf

`isPalindrom("KATAK")` akan mengembalikan **true**

```
public boolean isPalindrom (String str) {
     return isPalindromRecc (0, str.length()-1, str);
}

//recursive helper method
public boolean isPalindromRecc (int first, int last, String str) {

   if (first >= last) {   //hanya satu karakter, atau string kosong
      return true;        //pasti palindrom
   }

   char firstChar = str.charAt(first);
   char lastChar  = str.charAt(last);




   // lanjutkan ...



}
```

13. Implementasikan kembali method yang ada pada nomor 12, tetapi kali ini String masukan **bisa saja mengandung karakter non-huruf** (mis, angka, tanda baca, dll). Seandainya String masukan mengandung karakter non-huruf, karakter non-huruf tersebut akan diabaikan.

Contoh:
`isPalindrom("KA236TA4K5")` akan mengembalikan **true**