

1. Soal 1

Buatlah 1 set data random bulat positif kurang dari 10.000.000 sebanyak 1.000.000. Selanjutnya simpan data tersebut ke dalam array

```
package praktikum_11_alstruk;  
public class Praktikum_11_alstruk {  
  
    public static void main(String[] args) {  
  
        Array s = new Array(1000000);  
        s.random(10000000);  
  
    }  
}
```

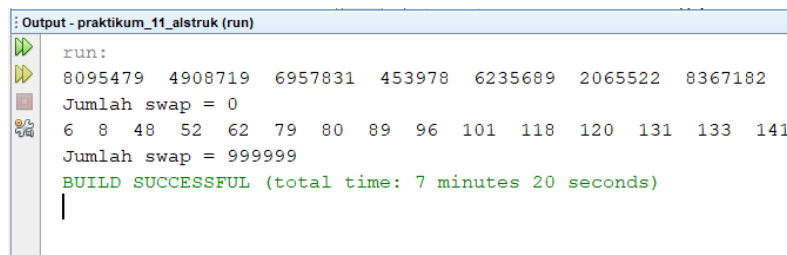
2. Soal 2

Lakukanlah sorting pada set data tersebut dengan menggunakan selection sort, selanjutnya hitunglah berapa kali dilakukan data swaping (menukar data) dan waktu yang diperlukan untuk sorting setiap set data.

```
package praktikum_11_alstruk;  
public class Praktikum_11_alstruk {  
  
    public static void main(String[] args) {  
  
        Array s = new Array(1000000);  
        s.random(10000000);  
        s.print();  
        s.selection();  
        s.print();  
  
    }  
}
```

Output:

Terjadi 999999 kali penukaran dalam 7 menit 20 detik.



```
Output - praktikum_11_alstruk (run)  
run:  
8095479 4908719 6957831 453978 6235689 2065522 8367182  
Jumlah swap = 0  
6 8 48 52 62 79 80 89 96 101 118 120 131 133 141  
Jumlah swap = 999999  
BUILD SUCCESSFUL (total time: 7 minutes 20 seconds)  
|
```

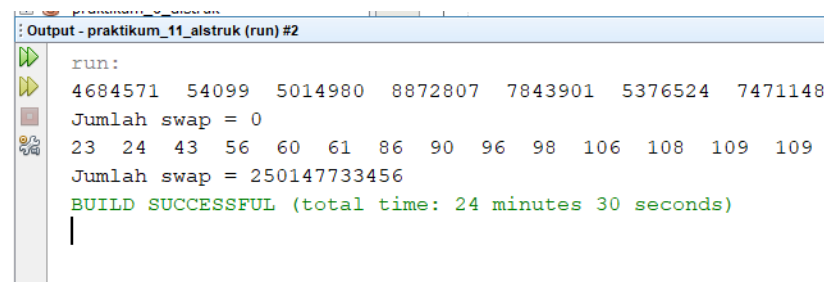
3. Soal 3

Lakukanlah sorting pada set data tersebut dengan menggunakan bubble sort, selanjutnya hitunglah berapa kali dilakukan data swaping (menukar data) dan waktu yang diperlukan untuk sorting setiap set data.

```
package praktikum_11_alstruk;  
public class Praktikum_11_alstruk {  
  
    public static void main(String[] args) {  
  
        Array b = new Array(1000000);  
        b.random(10000000);  
        b.print();  
        b.bubblesort();  
        b.print()  
    }  
}
```

Output:

Terjadi 250147733456 kali penukaran data dalam 24 menit 30 detik



```
run:  
4684571 54099 5014980 8872807 7843901 5376524 7471148  
Jumlah swap = 0  
23 24 43 56 60 61 86 90 96 98 106 108 109 109  
Jumlah swap = 250147733456  
BUILD SUCCESSFUL (total time: 24 minutes 30 seconds)
```

4. Soal 4

Bandingkanlah jumlah swaping dan waktu yang diperlukan antara metode bubble sort dan selection sort

Dari percobaan yang telah dilakukan, dapat disimpulkan bahwa metode selection sorting lebih efektif dibandingkan dengan bubble sorting. Hal tersebut dapat dilihat dari banyaknya jumlah penukaran yang dilakukan dan waktu eksekusi yang memiliki selisih cukup besar pada kedua metode tersebut

1. Soal 1

Buatlah 1 set data random bulat positif kurang dari 10.000.000 sebanyak 200.000. Selanjutnya simpan data tersebut ke dalam array

```
package praktikum_11_alstruk;
public class Praktikum_11_alstruk {

    public static void main(String[] args) {

        Array i = new Array(200000);
        i.random(100000000);

    }
}
```

2. Soal 2

Lakukanlah sorting pada set data tersebut dengan menggunakan insertion sort, selanjutnya hitunglah berapa kali dilakukan data shift (menggeser data) dan waktu yang diperlukan untuk sorting setiap set data.

```
package praktikum_11_alstruk;
public class Praktikum_11_alstruk {

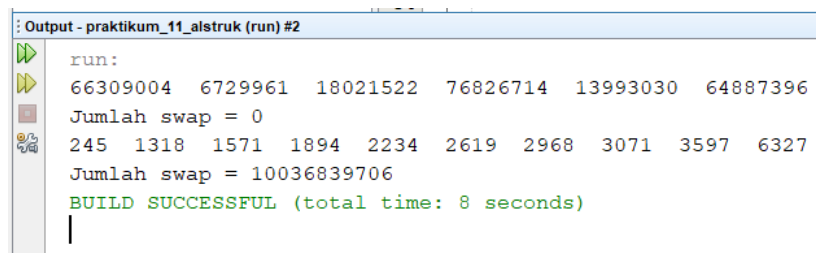
    public static void main(String[] args) {

        Array i = new Array(200000);
        i.random(100000000);
        i.print();
        i.insertion();
        i.print();

    }
}
```

Output:

Terjadi 10036839706 kali penggeseran data dalam 8 detik



```
Output - praktikum_11_alstruk (run) #2
run:
66309004  6729961  18021522  76826714  13993030  64887396
Jumlah swap = 0
245  1318  1571  1894  2234  2619  2968  3071  3597  6327
Jumlah swap = 10036839706
BUILD SUCCESSFUL (total time: 8 seconds)
```

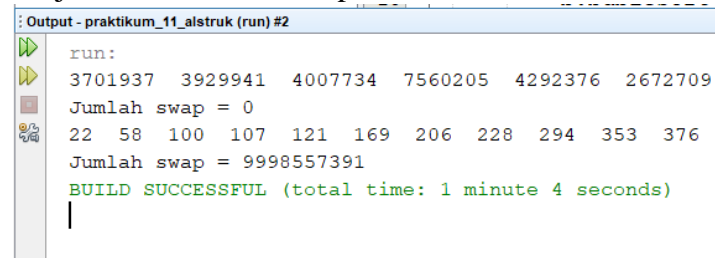
3. Soal 3

Badingkanlah jumlah swaping dan waktu yang diperlukan antara metode bubble sort dan insertion sort

```
package praktikum_11_alstruk;  
public class Praktikum_11_alstruk {  
  
    public static void main(String[] args) {  
  
        Array b = new Array(1000000);  
        b.random(10000000);  
        b.print();  
        b.bubblesort();  
        b.print()  
    }  
}
```

Output:

Terjadi 9998557391 kali penukaran data dalam 1 menit 4 detik.



The screenshot shows the output of a Java program. It starts with a 'run:' prompt, followed by a line of 10 random integers: 3701937 3929941 4007734 7560205 4292376 2672709. Below this, it says 'Jumlah swap = 0'. Then, another line of 10 integers is shown: 22 58 100 107 121 169 206 228 294 353 376. This is followed by 'Jumlah swap = 9998557391'. The final line in green text says 'BUILD SUCCESSFUL (total time: 1 minute 4 seconds)'.

Dari percobaan yang telah dilakukan, dapat disimpulkan bahwa metode insertion sorting lebih efektif dibandingkan dengan bubble sorting. Hal tersebut dapat dilihat dari banyaknya jumlah penukaran yang dilakukan dan waktu eksekusi yang memiliki selisih cukup besar pada kedua metode tersebut