



NAMA : Alexander Agung Raya  
NIM : 2341720040  
NO ABSEN : 03  
KELAS : 1F  
MATERI : SEARCHING

## LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

\*FILE NAME =03\_Alexander Agung Raya \_1F\_P7

\* Pertemuan 7

### SEARCHING

Percobaan 1 :

Buku03 :

```
using P1;

// Definisi kelas Buku03
class Buku03 {
public:
    // Konstruktor
    Buku03() {}
    Buku03(int id, string judul, int tahun, int stok) {
        idBuku = id;
        judulBuku = judul;
        tahunBuku = tahun;
        stokBuku = stok;
    }
    // Getter dan Setter
    int getIdBuku() const { return idBuku; }
    void setIdBuku(int id) { idBuku = id; }
    string getJudulBuku() const { return judulBuku; }
    void setJudulBuku(string judul) { judulBuku = judul; }
    int getTahunBuku() const { return tahunBuku; }
    void setTahunBuku(int tahun) { tahunBuku = tahun; }
    int getStokBuku() const { return stokBuku; }
    void setStokBuku(int stok) { stokBuku = stok; }
};

// Definisi array untuk menyimpan data buku
Buku03 buku[10];

// Fungsi untuk mencari buku berdasarkan ID
int cariBuku(int id) {
    for (int i = 0; i < 10; i++) {
        if (buku[i].getIdBuku() == id) {
            return i;
        }
    }
    return -1;
}
```

Bukumain03 :

```
using P1;
using P2;
using P3;
using P4;
using P5;
using P6;
using P7;
using P8;
using P9;
using P10;
using P11;
using P12;
using P13;
using P14;
using P15;
using P16;
using P17;
using P18;
using P19;
using P20;
using P21;
using P22;
using P23;
using P24;
using P25;
using P26;
using P27;
using P28;
using P29;
using P30;
using P31;
using P32;
using P33;
using P34;
using P35;
using P36;
using P37;
using P38;
using P39;
using P40;
using P41;
using P42;
using P43;
using P44;
using P45;
using P46;
using P47;
using P48;
using P49;
using P50;
using P51;
using P52;
using P53;
using P54;
using P55;
using P56;
using P57;
using P58;
using P59;
using P60;
using P61;
using P62;
using P63;
using P64;
using P65;
using P66;
using P67;
using P68;
using P69;
using P70;
using P71;
using P72;
using P73;
using P74;
using P75;
using P76;
using P77;
using P78;
using P79;
using P80;
using P81;
using P82;
using P83;
using P84;
using P85;
using P86;
using P87;
using P88;
using P89;
using P90;
using P91;
using P92;
using P93;
using P94;
using P95;
using P96;
using P97;
using P98;
using P99;
using P100;
```

PencarianBuku03 :

```
using P1;
using P2;
using P3;
using P4;
using P5;
using P6;
using P7;
using P8;
using P9;
using P10;
using P11;
using P12;
using P13;
using P14;
using P15;
using P16;
using P17;
using P18;
using P19;
using P20;
using P21;
using P22;
using P23;
using P24;
using P25;
using P26;
using P27;
using P28;
using P29;
using P30;
using P31;
using P32;
using P33;
using P34;
using P35;
using P36;
using P37;
using P38;
using P39;
using P40;
using P41;
using P42;
using P43;
using P44;
using P45;
using P46;
using P47;
using P48;
using P49;
using P50;
using P51;
using P52;
using P53;
using P54;
using P55;
using P56;
using P57;
using P58;
using P59;
using P60;
using P61;
using P62;
using P63;
using P64;
using P65;
using P66;
using P67;
using P68;
using P69;
using P70;
using P71;
using P72;
using P73;
using P74;
using P75;
using P76;
using P77;
using P78;
using P79;
using P80;
using P81;
using P82;
using P83;
using P84;
using P85;
using P86;
using P87;
using P88;
using P89;
using P90;
using P91;
using P92;
using P93;
using P94;
using P95;
using P96;
using P97;
using P98;
using P99;
using P100;
```



NAMA : Alexander Agung Raya  
NIM : 2341720040  
NO ABSEN : 03  
KELAS : 1F  
MATERI : SEARCHING

Hasil :

```
kode Buku : 444
Judul Buku : home sweet home
Tahun Terbit : 2023
Pengarang : alex
Stock Buku : 4
=====
kode Buku : 555
Judul Buku : baju
Tahun Terbit : 2024
Pengarang : alex
Stock Buku : 5
=====
pencarian Data :
Masukan kode Buku yg di cari :
Kode Buku :
111
Menggunakan Sequential Seacring
Data : 111Tidak di temukan
=====
kode Buku : 111
Judul Buku : alex
Tahun Terbit : 2020
Pengarang : alex
Stock Buku : 1
ASUS 11. Pratik 03 11. Imaster a 11-2 11. 11
```

Pertayaan :

1. Jelaskan fungsi break yang ada pada method FindSeqSearch!  
Jawab : Break di gunakan untuk menghentikan suatu perulangan jika tidak di break akan terus di ulang
2. Jika Data Kode Buku yang dimasukkan tidak terurut dari kecil ke besar. Apakah program masih dapat berjalan? Apakah hasil yang dikeluarkan benar? Tunjukkan hasil screenshoot untuk bukti dengan kode Buku yang acak. Jelaskan Mengapa hal tersebut bisa terjadi?

Jawab :

Bisa Hasil juga akan tetap di jalankan

```
Pengarang : alex
Stock Buku : 111
=====
kode Buku : 111
Judul Buku : MC
Tahun Terbit : 2014
Pengarang : alex
Stock Buku : 1
=====
pencarian Data :
Masukan kode Buku yg di cari :
Kode Buku :
555
Menggunakan Sequential Seacring
Data : 555Tidak di temukan
=====
kode Buku : 555
Judul Buku : alex
Tahun Terbit : 1811
Pengarang : alex
Stock Buku : 5
```

3. Buat method baru dengan nama FindBuku menggunakan konsep sequential search dengan tipe method dari FindBuku adalah BukuNoAbsen. Sehingga Anda bisa memanggil method tersebut pada class BukuMain seperti gambar berikut :

```
Buku dataBuku = data.FindBuku(cari);
dataBuku.tampilDataBuku();
```

Jawab :



NAMA : Alexander Agung Raya  
NIM : 2341720040  
NO ABSEN : 03  
KELAS : 1F  
MATERI : SEARCHING

```
=====
Pencarian : alex
Stock Buku : 111
=====
kode Buku : 111
Judul Buku : MC
Tahun Terbit : 2014
Pencarian : alex
Stock Buku : 1
=====

pencarian Data :
Masukan kode Buku yg di cari :
Kode Buku :
555
Menggunakan Sequential Seacring
Data : 555 tidak di temukan
=====

kode Buku : 555
Judul Buku : alex
Tahun Terbit : 1011
Pencarian : alex
Stock Buku : 5
```

Bukumain03 :

```
1 package P3;
2 import java.util.Scanner;
3
4 public class Bukumain03 {
5     public static void main(String[] args) {
6         Scanner s = new Scanner(System.in);
7         Scanner sk = new Scanner(System.in);
8
9         PencarianData() data = new PencarianData();
10
11         int jmlData = 5;
12
13         System.out.println("Masukan Data Buku secara acak dari kode buku yg tersedia : ");
14         for (int i = 0; i < jmlData; i++) {
15             System.out.println("Masukan Data Buku : ");
16             int kodeBuku = data.kodeBuku(i);
17             String judulBuku = data.judulBuku(i);
18             int tahunTerbit = data.tahunTerbit(i);
19             int tahunTerbit = data.tahunTerbit(i);
20             String pengarang = data.pengarang(i);
21             System.out.println("Kode Buku : " + i);
22             int kodeBuku = data.kodeBuku(i);
23
24             buku[i] = new buku(kodeBuku, judulBuku, tahunTerbit, pengarang, kode);
25         }
26
27         System.out.println("=====");
28         System.out.println("Data Buku yang Tersedia : ");
29         System.out.println("Data Buku yang Tersedia : ");
30         data.tampil();
31
32         System.out.println("=====");
33         System.out.println("Masukan Data Buku : ");
34         System.out.println("Masukan Data Buku yg di cari : ");
35         int cari = data.kodeBuku();
36         int cari = data.kodeBuku();
37         System.out.println("Masukan Data Buku yg di cari : ");
38         int cari = data.kodeBuku();
39         data.tampil();
40
41         Buku[] buku = data.buku;
42         data.tampil();
43     }
44 }
```

pencariannncaribuku03 :

```
1 package P3;
2
3 public class PencarianBuku {
4     public static void main(String[] args) {
5         Scanner s = new Scanner(System.in);
6
7         void tampil() {
8             for (int i = 0; i < buku.length; i++) {
9                 System.out.println("Data Buku : ");
10                 int kodeBuku = data.kodeBuku(i);
11                 String judulBuku = data.judulBuku(i);
12                 int tahunTerbit = data.tahunTerbit(i);
13                 String pengarang = data.pengarang(i);
14                 System.out.println("Kode Buku : " + i);
15                 int kodeBuku = data.kodeBuku(i);
16
17                 buku[i] = new buku(kodeBuku, judulBuku, tahunTerbit, pengarang, kode);
18             }
19         }
20
21         void tampil() {
22             for (int i = 0; i < buku.length; i++) {
23                 System.out.println("Data Buku : ");
24                 int kodeBuku = data.kodeBuku(i);
25                 String judulBuku = data.judulBuku(i);
26                 int tahunTerbit = data.tahunTerbit(i);
27                 String pengarang = data.pengarang(i);
28                 System.out.println("Kode Buku : " + i);
29                 int kodeBuku = data.kodeBuku(i);
30
31                 buku[i] = new buku(kodeBuku, judulBuku, tahunTerbit, pengarang, kode);
32             }
33         }
34
35         void tampil() {
36             for (int i = 0; i < buku.length; i++) {
37                 System.out.println("Data Buku : ");
38                 int kodeBuku = data.kodeBuku(i);
39                 String judulBuku = data.judulBuku(i);
40                 int tahunTerbit = data.tahunTerbit(i);
41                 String pengarang = data.pengarang(i);
42                 System.out.println("Kode Buku : " + i);
43                 int kodeBuku = data.kodeBuku(i);
44
45                 buku[i] = new buku(kodeBuku, judulBuku, tahunTerbit, pengarang, kode);
46             }
47         }
48     }
49 }
```



NAMA : Alexander Agung Raya  
NIM : 2341720040  
NO ABSEN : 03  
KELAS : 1F  
MATERI : SEARCHING

Hasil :

```
kode Buku : 444
judul Buku : Room sweet home
tahun Terbit : 2023
pengarang : alex
stock Buku : 4

=====
kode Buku : 505
judul Buku : buku
tahun Terbit : 2024
pengarang : alex
stock Buku : 5

=====
pencarian Data :
Masukan kode buku yg di cari :
kode Buku : 111
Menggunakan Sequential Searching
Data : 111 tidak di temukan

kode Buku : 111
judul Buku : alex
tahun Terbit : 2020
pengarang : alex
stock Buku : 1
```

Percobaan 2 :

```
1 // Header file
2 #include <iostream>
3 using namespace std;
4 // Definisi variabel global
5 int n; // Jumlah data
6 int arr[100]; // Array data
7 // Fungsi untuk memasukkan data
8 void inputData() {
9     for (int i = 0; i < n; i++) {
10         cout << "Masukkan data ke-" << i+1 << ": ";
11         cin >> arr[i];
12     }
13 }
14 // Fungsi untuk mencari data
15 int cariData(int x) {
16     for (int i = 0; i < n; i++) {
17         if (arr[i] == x) {
18             return i; // Return index jika ditemukan
19         }
20     }
21     return -1; // Return -1 jika tidak ditemukan
22 }
23 // Fungsi untuk menampilkan data
24 void tampilkanData() {
25     for (int i = 0; i < n; i++) {
26         cout << arr[i] << " ";
27     }
28     cout << endl;
29 }
30 // Fungsi untuk menampilkan status pencarian
31 void tampilkanStatus(int index) {
32     if (index != -1) {
33         cout << "Data ditemukan di index " << index << endl;
34     } else {
35         cout << "Data tidak ditemukan" << endl;
36     }
37 }
38 // Fungsi utama
39 int main() {
40     cout << "Masukkan jumlah data: ";
41     cin >> n;
42     inputData();
43     tampilkanData();
44     int x;
45     cout << "Masukkan data yang dicari: ";
46     cin >> x;
47     int index = cariData(x);
48     tampilkanStatus(index);
49     return 0;
50 }
```

```
1 // Header file
2 #include <iostream>
3 using namespace std;
4 // Definisi variabel global
5 int n; // Jumlah data
6 int arr[100]; // Array data
7 // Fungsi untuk memasukkan data
8 void inputData() {
9     for (int i = 0; i < n; i++) {
10         cout << "Masukkan data ke-" << i+1 << ": ";
11         cin >> arr[i];
12     }
13 }
14 // Fungsi untuk mencari data
15 int cariData(int x) {
16     for (int i = 0; i < n; i++) {
17         if (arr[i] == x) {
18             return i; // Return index jika ditemukan
19         }
20     }
21     return -1; // Return -1 jika tidak ditemukan
22 }
23 // Fungsi untuk menampilkan data
24 void tampilkanData() {
25     for (int i = 0; i < n; i++) {
26         cout << arr[i] << " ";
27     }
28     cout << endl;
29 }
30 // Fungsi untuk menampilkan status pencarian
31 void tampilkanStatus(int index) {
32     if (index != -1) {
33         cout << "Data ditemukan di index " << index << endl;
34     } else {
35         cout << "Data tidak ditemukan" << endl;
36     }
37 }
38 // Fungsi utama
39 int main() {
40     cout << "Masukkan jumlah data: ";
41     cin >> n;
42     inputData();
43     tampilkanData();
44     int x;
45     cout << "Masukkan data yang dicari: ";
46     cin >> x;
47     int index = cariData(x);
48     tampilkanStatus(index);
49     return 0;
50 }
```



NAMA : Alexander Agung Raya  
NIM : 2341720040  
NO ABSEN : 03  
KELAS : 1F  
MATERI : SEARCHING

Hasil :

```
=====
pencarian Data :
Masukan kode Buku yg di cari :
Kode Buku :
111
Menggunakan Sequential Seacring
Data : 111Tidak di temukan
=====
kode Buku : 111
Judul Buku : alex
Tahun Terbit : 2010
Pengarang : alex
Stock Buku : 1
Data : 111Tidak di temukan
=====
Menggunakan Binary Seacring
Data : 111Ditemukan Pada index : 0
=====
kode Buku : 111
Judul Buku : alex
Tahun Terbit : 2010
Pengarang : alex
Stock Buku : 1
PS C:\Users\ASUS\Documents\File untuk kuliah\Semester-2\Pr
```

Pertayaan :

1. Tunjukkan pada kode program yang mana proses divide dijalankan!

Jawab :

int mid;

```
if (right >= left) {
    mid = (right) / 2;
```

2. Tunjukkan pada kode program yang mana proses conquer dijalankan!

Jawab :

```
else if (listBK[mid].KodeBuku > cari) {
    return FindBinarySearch(cari, left, mid - 1);
} else {
    return FindBinarySearch(cari, mid + 1, right);
}
```

3. Jika data Kode Buku yang dimasukkan tidak urut. Apakah program masih dapat berjalan?

Mengapa demikian! Tunjukkan hasil screenshoot untuk bukti dengan kode Buku yang acak.  
Jelaskan Mengapa hal tersebut bisa terjadi?

Jawab :

Jika data Kode Buku yang dimasukkan tidak urut, program masih dapat berjalan, namun hasil pencarian dengan metode binary search mungkin tidak akurat



NAMA : Alexander Agung Raya  
NIM : 2341720040  
NO ABSEN : 03  
KELAS : 1F  
MATERI : SEARCHING

```
=====
kode Buku : 555
Judul Buku : alex
Tahun Terbit : 2011
Pengarang : alex
Stock Buku : 5
=====
kode Buku : 777
Judul Buku : alex
Tahun Terbit : 2012
Pengarang : 7
Stock Buku : 7
=====
kode Buku : 222
Judul Buku : alex
Tahun Terbit : 2019
Pengarang : alex
Stock Buku : 2
=====
kode Buku : 999
Judul Buku : alex
Tahun Terbit : 2020
Pengarang : alex
Stock Buku : 8
=====
pencarian Data :
Masukan kode Buku yg di cari :
Kode Buku :
777
Menggunakan Sequential Seacring
Data : 777Tidak di temukan
=====
kode Buku : 777
Judul Buku : alex
Tahun Terbit : 2012
Pengarang : 7
Stock Buku : 7
Data : 777Tidak di temukan
```

4. Jika Kode Buku yang dimasukkan dari Kode Buku terbesar ke terkecil (missal : 20215, 20214, 20212, 20211, 20210) dan elemen yang dicari adalah 20210. Bagaimana hasil dari binary search? Apakah sesuai? Jika tidak sesuai maka ubahlah kode program binary seach agar hasilnya sesuai!

Jawab :

Jika kode buku yang dimasukkan dari yang terbesar ke terkecil dan kita mencari elemen dengan kode buku 20210 menggunakan binary search, hasilnya akan sesuai. Hal ini karena binary search berfungsi dengan baik saat data dalam keadaan terurut.

```
Kode Buku :
20210
Menggunakan Sequential Seacring
Data : 20210Tidak di temukan
=====
kode Buku : 20210
Judul Buku : kabel
Tahun Terbit : 2080
Pengarang : alex
Stock Buku : 10
Data : 20210Tidak di temukan
=====
Menggunakan Binary Searcng
Data : 20210Tidak di temukan
=====
kode Buku : 20210
Judul Buku : kabel
Tahun Terbit : 2080
Pengarang : alex
Stock Buku : 10
```



NAMA : Alexander Agung Raya  
NIM : 2341720040  
NO ABSEN : 03  
KELAS : 1F  
MATERI : SEARCHING

Percobaan 3 :

margeSorting :

```
1 package P7.MergeSortTest;
2
3 public class MergeSortings03 {
4     public void MergeSort(int[] data) {
5         if (data == null || data.length <= -1) {
6             return;
7         }
8         Sort(data, 0, data.length - 1);
9     }
10    public void Merge(int data [], int left, int middle, int right) {
11        int[] temp = new int[data.length];
12        for(int i = left; i <= right; i++) {
13            temp[i] = data[i];
14        }
15        int a = left;
16        int b = middle + 1;
17        int c = left;
18
19        while (a <= middle && b <= right) {
20            if (temp[a] <= temp[b]) {
21                data[c] = temp[a];
22                a++;
23            } else {
24                data[c] = temp[b];
25                b++;
26            }
27            c++;
28        }
29        int s = middle - a;
30        for (int i = 0; i <= s; i++) {
31            data[c + i] = temp[a + i];
32        }
33    }
34    public void Sort(int data [], int left, int right){
35        if (left < right) {
36            int middle = (left + right)/2;
37            Sort(data, left, middle);
38            Sort(data, middle + 1, right);
39            Merge(data, left, middle, right);
40        }
41    }
42    public void printArray(int arr[]) {
43        for (int i = 0; i < arr.length; i++) {
44            System.out.println(arr[i] + " ");
45        }
46        System.out.println();
47    }
48 }
49
```

MargeSortingMain :

```
1 package P7.MergeSortTest;
2
3 public class MergeSortMain03 {
4     public static void main(String[] args) {
5         int data [] = {10,40,30,50,70,20,100,90};
6         System.out.println("sorting dan merge Sort ");
7         MergeSortings03 mSort = new MergeSortings03();
8         System.out.println("Data Awal");
9         mSort.printArray(data);
10        mSort.MergeSort(data);
11        System.out.println("Setelah Di urutkan :");
12        mSort.printArray(data);
13    }
14 }
15
```



NAMA : Alexander Agung Raya  
NIM : 2341720040  
NO ABSEN : 03  
KELAS : 1F  
MATERI : SEARCHING

Hasil :

```
0001
sorting dan merge Sort
Data Awal
10
40
30
50
70
20
100
90

Setelah Di urutkan :
10
20
30
40
50
70
90
100
```

Latihan :

1. Modifikasi percobaan searching diatas dengan ketentuan berikut ini
  - Ubah tipe data dari kode Buku yang awalnya int menjadi String
  - Tambahkan method untuk pencarian kode Buku (bertipe data String) dengan menggunakan sequential search dan binary search.

Jawab :

```
1 public class Buku03 {
2     String KodeBuku;
3     String JudulBuku;
4     int TahunTerbit;
5     String Pengarang;
6     int Stock;
```





NAMA : Alexander Agung Raya  
NIM : 2341720040  
NO ABSEN : 03  
KELAS : 1F  
MATERI : SEARCHING

```
1 public BukuB(String KodeBuku,String JudulBuku,int TahunTerbit, String Pengarang, int Stock) {  
2     this.KodeBuku = KodeBuku;  
3     this.JudulBuku = JudulBuku;  
4     this.TahunTerbit = TahunTerbit;  
5     this.Pengarang = Pengarang;  
6     this.Stock = Stock;  
7 }  
  
1 public int FindBinarySearch(String cari, int left, int right) {  
2     if (right >= left) {  
3         int mid = left + (right - left) / 2;  
4         if (cari.equalsIgnoreCase(listBK[mid].KodeBuku)) {  
5             return (mid);  
6         }  
7         else if (Integer.valueOf(cari)>Integer.valueOf(listBK[mid].KodeBuku)) {  
8             return FindBinarySearch(cari, 0, mid - 1);  
9         } else if (Integer.valueOf(cari)<Integer.valueOf(listBK[mid].KodeBuku)){  
10            return FindBinarySearch(cari, mid + 1, right);  
11        }  
12    }  
13    return -1;  
14 }
```

```
1 public int FindSeqSearch(String cari) {  
2     int posisi = -1;  
3     for(int j = 0; j < listBK.length; j++) {  
4         if (listBK[j].KodeBuku == cari) {  
5             posisi = j;  
6             break;  
7         }  
8     }  
9     return posisi;  
10 }
```

2. Modifikasi percobaan searching diatas dengan ketentuan berikut ini
- Tambahkan method pencarian judul buku menggunakan sequential search dan binary search. Sebelum dilakukan searching dengan binary search data harus dilakukan pengurutan dengan menggunakan algoritma Sorting (bebas pilih algoritma sorting apapun)! Sehingga ketika input data acak, maka algoritma searching akan tetap berjalan
  - Buat aturan untuk mendeteksi hasil pencarian judul buku yang lebih dari 1 hasil dalam bentuk kalimat peringatan! Pastikan algoritma yang diterapkan sesuai dengan kasus yang diberikan!

Jawab :

```
1 public class Main {  
2     public static void main(String[] args) {  
3         Scanner input = new Scanner(System.in);  
4         System.out.println("Masukkan jumlah data buku yang akan dicari: ");  
5         int n = input.nextInt();  
6         BukuB[] listBK = new BukuB[n];  
7         for (int i = 0; i < n; i++) {  
8             System.out.println("Masukkan data buku ke-" + (i+1) + ":");  
9             listBK[i] = new BukuB(input.next(), input.next(), input.nextInt(), input.next(), input.nextInt());  
10        }  
11        System.out.println("Data buku yang akan dicari:");  
12        for (int i = 0; i < n; i++) {  
13            System.out.println(listBK[i].KodeBuku + " | " + listBK[i].JudulBuku + " | " + listBK[i].TahunTerbit + " | " + listBK[i].Pengarang + " | " + listBK[i].Stock);  
14        }  
15        System.out.println("Masukkan judul buku yang akan dicari:");  
16        String cari = input.next();  
17        int posisi = -1;  
18        for (int j = 0; j < listBK.length; j++) {  
19            if (listBK[j].KodeBuku == cari) {  
20                posisi = j;  
21                break;  
22            }  
23        }  
24        System.out.println("Posisi buku yang dicari: " + posisi);  
25        if (posisi == -1) {  
26            System.out.println("Tidak ditemukan buku yang dicari!");  
27        } else {  
28            System.out.println("Buku yang dicari ditemukan!");  
29        }  
30    }  
31 }
```



NAMA : Alexander Agung Raya  
NIM : 2341720040  
NO ABSEN : 03  
KELAS : 1F  
MATERI : SEARCHING

```
1 // mencari ke belakang (2)
2
3 public static boolean cari (
4     String[] data, int cari) {
5     for (int i = data.length - 1; i >= 0; i--) {
6         if (data[i].equals(cari)) {
7             return true;
8         }
9     }
10    return false;
11 }
12
13 // mencari ke depan (3)
14
15 public static boolean cari (
16     String[] data, int cari) {
17     for (int i = 0; i < data.length; i++) {
18         if (data[i].equals(cari)) {
19             return true;
20         }
21     }
22    return false;
23 }
24
25 // mencari ke belakang (4)
26
27 public static boolean cari (
28     String[] data, int cari) {
29     for (int i = data.length - 1; i >= 0; i--) {
30         if (data[i].equals(cari)) {
31             return true;
32         }
33     }
34    return false;
35 }
36
37 // mencari ke depan (5)
38
39 public static boolean cari (
40     String[] data, int cari) {
41     for (int i = 0; i < data.length; i++) {
42         if (data[i].equals(cari)) {
43             return true;
44         }
45     }
46    return false;
47 }
48
49 // mencari ke belakang (6)
50
51 public static boolean cari (
52     String[] data, int cari) {
53     for (int i = data.length - 1; i >= 0; i--) {
54         if (data[i].equals(cari)) {
55             return true;
56         }
57     }
58    return false;
59 }
60
61 // mencari ke depan (7)
62
63 public static boolean cari (
64     String[] data, int cari) {
65     for (int i = 0; i < data.length; i++) {
66         if (data[i].equals(cari)) {
67             return true;
68         }
69     }
70    return false;
71 }
72
73 // mencari ke belakang (8)
74
75 public static boolean cari (
76     String[] data, int cari) {
77     for (int i = data.length - 1; i >= 0; i--) {
78         if (data[i].equals(cari)) {
79             return true;
80         }
81     }
82    return false;
83 }
84
85 // mencari ke depan (9)
86
87 public static boolean cari (
88     String[] data, int cari) {
89     for (int i = 0; i < data.length; i++) {
90         if (data[i].equals(cari)) {
91             return true;
92         }
93     }
94    return false;
95 }
96
97 // mencari ke belakang (10)
98
99 public static boolean cari (
100    String[] data, int cari) {
101    for (int i = data.length - 1; i >= 0; i--) {
102        if (data[i].equals(cari)) {
103            return true;
104        }
105    }
106    return false;
107 }
108
109 // mencari ke depan (11)
110
111 public static boolean cari (
112    String[] data, int cari) {
113    for (int i = 0; i < data.length; i++) {
114        if (data[i].equals(cari)) {
115            return true;
116        }
117    }
118    return false;
119 }
120
121 // mencari ke belakang (12)
122
123 public static boolean cari (
124    String[] data, int cari) {
125    for (int i = data.length - 1; i >= 0; i--) {
126        if (data[i].equals(cari)) {
127            return true;
128        }
129    }
130    return false;
131 }
132
133 // mencari ke depan (13)
134
135 public static boolean cari (
136    String[] data, int cari) {
137    for (int i = 0; i < data.length; i++) {
138        if (data[i].equals(cari)) {
139            return true;
140        }
141    }
142    return false;
143 }
144
145 // mencari ke belakang (14)
146
147 public static boolean cari (
148    String[] data, int cari) {
149    for (int i = data.length - 1; i >= 0; i--) {
150        if (data[i].equals(cari)) {
151            return true;
152        }
153    }
154    return false;
155 }
156
157 // mencari ke depan (15)
158
159 public static boolean cari (
160    String[] data, int cari) {
161    for (int i = 0; i < data.length; i++) {
162        if (data[i].equals(cari)) {
163            return true;
164        }
165    }
166    return false;
167 }
168
169 // mencari ke belakang (16)
170
171 public static boolean cari (
172    String[] data, int cari) {
173    for (int i = data.length - 1; i >= 0; i--) {
174        if (data[i].equals(cari)) {
175            return true;
176        }
177    }
178    return false;
179 }
180
181 // mencari ke depan (17)
182
183 public static boolean cari (
184    String[] data, int cari) {
185    for (int i = 0; i < data.length; i++) {
186        if (data[i].equals(cari)) {
187            return true;
188        }
189    }
190    return false;
191 }
192
193 // mencari ke belakang (18)
194
195 public static boolean cari (
196    String[] data, int cari) {
197    for (int i = data.length - 1; i >= 0; i--) {
198        if (data[i].equals(cari)) {
199            return true;
200        }
201    }
202    return false;
203 }
204
205 // mencari ke depan (19)
206
207 public static boolean cari (
208    String[] data, int cari) {
209    for (int i = 0; i < data.length; i++) {
210        if (data[i].equals(cari)) {
211            return true;
212        }
213    }
214    return false;
215 }
216
217 // mencari ke belakang (20)
218
219 public static boolean cari (
220    String[] data, int cari) {
221    for (int i = data.length - 1; i >= 0; i--) {
222        if (data[i].equals(cari)) {
223            return true;
224        }
225    }
226    return false;
227 }
228
229 // mencari ke depan (21)
230
231 public static boolean cari (
232    String[] data, int cari) {
233    for (int i = 0; i < data.length; i++) {
234        if (data[i].equals(cari)) {
235            return true;
236        }
237    }
238    return false;
239 }
240
241 // mencari ke belakang (22)
242
243 public static boolean cari (
244    String[] data, int cari) {
245    for (int i = data.length - 1; i >= 0; i--) {
246        if (data[i].equals(cari)) {
247            return true;
248        }
249    }
250    return false;
251 }
252
253 // mencari ke depan (23)
254
255 public static boolean cari (
256    String[] data, int cari) {
257    for (int i = 0; i < data.length; i++) {
258        if (data[i].equals(cari)) {
259            return true;
260        }
261    }
262    return false;
263 }
264
265 // mencari ke belakang (24)
266
267 public static boolean cari (
268    String[] data, int cari) {
269    for (int i = data.length - 1; i >= 0; i--) {
270        if (data[i].equals(cari)) {
271            return true;
272        }
273    }
274    return false;
275 }
276
277 // mencari ke depan (25)
278
279 public static boolean cari (
280    String[] data, int cari) {
281    for (int i = 0; i < data.length; i++) {
282        if (data[i].equals(cari)) {
283            return true;
284        }
285    }
286    return false;
287 }
288
289 // mencari ke belakang (26)
290
291 public static boolean cari (
292    String[] data, int cari) {
293    for (int i = data.length - 1; i >= 0; i--) {
294        if (data[i].equals(cari)) {
295            return true;
296        }
297    }
298    return false;
299 }
300
301 // mencari ke depan (27)
302
303 public static boolean cari (
304    String[] data, int cari) {
305    for (int i = 0; i < data.length; i++) {
306        if (data[i].equals(cari)) {
307            return true;
308        }
309    }
310    return false;
311 }
312
313 // mencari ke belakang (28)
314
315 public static boolean cari (
316    String[] data, int cari) {
317    for (int i = data.length - 1; i >= 0; i--) {
318        if (data[i].equals(cari)) {
319            return true;
320        }
321    }
322    return false;
323 }
324
325 // mencari ke depan (29)
326
327 public static boolean cari (
328    String[] data, int cari) {
329    for (int i = 0; i < data.length; i++) {
330        if (data[i].equals(cari)) {
331            return true;
332        }
333    }
334    return false;
335 }
336
337 // mencari ke belakang (30)
338
339 public static boolean cari (
340    String[] data, int cari) {
341    for (int i = data.length - 1; i >= 0; i--) {
342        if (data[i].equals(cari)) {
343            return true;
344        }
345    }
346    return false;
347 }
348
349 // mencari ke depan (31)
350
351 public static boolean cari (
352    String[] data, int cari) {
353    for (int i = 0; i < data.length; i++) {
354        if (data[i].equals(cari)) {
355            return true;
356        }
357    }
358    return false;
359 }
360
361 // mencari ke belakang (32)
362
363 public static boolean cari (
364    String[] data, int cari) {
365    for (int i = data.length - 1; i >= 0; i--) {
366        if (data[i].equals(cari)) {
367            return true;
368        }
369    }
370    return false;
371 }
372
373 // mencari ke depan (33)
374
375 public static boolean cari (
376    String[] data, int cari) {
377    for (int i = 0; i < data.length; i++) {
378        if (data[i].equals(cari)) {
379            return true;
380        }
381    }
382    return false;
383 }
384
385 // mencari ke belakang (34)
386
387 public static boolean cari (
388    String[] data, int cari) {
389    for (int i = data.length - 1; i >= 0; i--) {
390        if (data[i].equals(cari)) {
391            return true;
392        }
393    }
394    return false;
395 }
396
397 // mencari ke depan (35)
398
399 public static boolean cari (
400    String[] data, int cari) {
401    for (int i = 0; i < data.length; i++) {
402        if (data[i].equals(cari)) {
403            return true;
404        }
405    }
406    return false;
407 }
408
409 // mencari ke belakang (36)
410
411 public static boolean cari (
412    String[] data, int cari) {
413    for (int i = data.length - 1; i >= 0; i--) {
414        if (data[i].equals(cari)) {
415            return true;
416        }
417    }
418    return false;
419 }
420
421 // mencari ke depan (37)
422
423 public static boolean cari (
424    String[] data, int cari) {
425    for (int i = 0; i < data.length; i++) {
426        if (data[i].equals(cari)) {
427            return true;
428        }
429    }
430    return false;
431 }
432
433 // mencari ke belakang (38)
434
435 public static boolean cari (
436    String[] data, int cari) {
437    for (int i = data.length - 1; i >= 0; i--) {
438        if (data[i].equals(cari)) {
439            return true;
440        }
441    }
442    return false;
443 }
444
445 // mencari ke depan (39)
446
447 public static boolean cari (
448    String[] data, int cari) {
449    for (int i = 0; i < data.length; i++) {
450        if (data[i].equals(cari)) {
451            return true;
452        }
453    }
454    return false;
455 }
456
457 // mencari ke belakang (40)
458
459 public static boolean cari (
460    String[] data, int cari) {
461    for (int i = data.length - 1; i >= 0; i--) {
462        if (data[i].equals(cari)) {
463            return true;
464        }
465    }
466    return false;
467 }
468
469 // mencari ke depan (41)
470
471 public static boolean cari (
472    String[] data, int cari) {
473    for (int i = 0; i < data.length; i++) {
474        if (data[i].equals(cari)) {
475            return true;
476        }
477    }
478    return false;
479 }
480
481 // mencari ke belakang (42)
482
483 public static boolean cari (
484    String[] data, int cari) {
485    for (int i = data.length - 1; i >= 0; i--) {
486        if (data[i].equals(cari)) {
487            return true;
488        }
489    }
490    return false;
491 }
492
493 // mencari ke depan (43)
494
495 public static boolean cari (
496    String[] data, int cari) {
497    for (int i = 0; i < data.length; i++) {
498        if (data[i].equals(cari)) {
499            return true;
500        }
501    }
502    return false;
503 }
504
505 // mencari ke belakang (44)
506
507 public static boolean cari (
508    String[] data, int cari) {
509    for (int i = data.length - 1; i >= 0; i--) {
510        if (data[i].equals(cari)) {
511            return true;
512        }
513    }
514    return false;
515 }
516
517 // mencari ke depan (45)
518
519 public static boolean cari (
520    String[] data, int cari) {
521    for (int i = 0; i < data.length; i++) {
522        if (data[i].equals(cari)) {
523            return true;
524        }
525    }
526    return false;
527 }
528
529 // mencari ke belakang (46)
530
531 public static boolean cari (
532    String[] data, int cari) {
533    for (int i = data.length - 1; i >= 0; i--) {
534        if (data[i].equals(cari)) {
535            return true;
536        }
537    }
538    return false;
539 }
540
541 // mencari ke depan (47)
542
543 public static boolean cari (
544    String[] data, int cari) {
545    for (int i = 0; i < data.length; i++) {
546        if (data[i].equals(cari)) {
547            return true;
548        }
549    }
550    return false;
551 }
552
553 // mencari ke belakang (48)
554
555 public static boolean cari (
556    String[] data, int cari) {
557    for (int i = data.length - 1; i >= 0; i--) {
558        if (data[i].equals(cari)) {
559            return true;
560        }
561    }
562    return false;
563 }
564
565 // mencari ke depan (49)
566
567 public static boolean cari (
568    String[] data, int cari) {
569    for (int i = 0; i < data.length; i++) {
570        if (data[i].equals(cari)) {
571            return true;
572        }
573    }
574    return false;
575 }
576
577 // mencari ke belakang (50)
578
579 public static boolean cari (
580    String[] data, int cari) {
581    for (int i = data.length - 1; i >= 0; i--) {
582        if (data[i].equals(cari)) {
583            return true;
584        }
585    }
586    return false;
587 }
588
589 // mencari ke depan (51)
590
591 public static boolean cari (
592    String[] data, int cari) {
593    for (int i = 0; i < data.length; i++) {
594        if (data[i].equals(cari)) {
595            return true;
596        }
597    }
598    return false;
599 }
600
601 // mencari ke belakang (52)
602
603 public static boolean cari (
604    String[] data, int cari) {
605    for (int i = data.length - 1; i >= 0; i--) {
606        if (data[i].equals(cari)) {
607            return true;
608        }
609    }
610    return false;
611 }
612
613 // mencari ke depan (53)
614
615 public static boolean cari (
616    String[] data, int cari) {
617    for (int i = 0; i < data.length; i++) {
618        if (data[i].equals(cari)) {
619            return true;
620        }
621    }
622    return false;
623 }
624
625 // mencari ke belakang (54)
626
627 public static boolean cari (
628    String[] data, int cari) {
629    for (int i = data.length - 1; i >= 0; i--) {
630        if (data[i].equals(cari)) {
631            return true;
632        }
633    }
634    return false;
635 }
636
637 // mencari ke depan (55)
638
639 public static boolean cari (
640    String[] data, int cari) {
641    for (int i = 0; i < data.length; i++) {
642        if (data[i].equals(cari)) {
643            return true;
644        }
645    }
646    return false;
647 }
648
649 // mencari ke belakang (56)
650
651 public static boolean cari (
652    String[] data, int cari) {
653    for (int i = data.length - 1; i >= 0; i--) {
654        if (data[i].equals(cari)) {
655            return true;
656        }
657    }
658    return false;
659 }
660
661 // mencari ke depan (57)
662
663 public static boolean cari (
664    String[] data, int cari) {
665    for (int i = 0; i < data.length; i++) {
666        if (data[i].equals(cari)) {
667            return true;
668        }
669    }
670    return false;
671 }
672
673 // mencari ke belakang (58)
674
675 public static boolean cari (
676    String[] data, int cari) {
677    for (int i = data.length - 1; i >= 0; i--) {
678        if (data[i].equals(cari)) {
679            return true;
680        }
681    }
682    return false;
683 }
684
685 // mencari ke depan (59)
686
687 public static boolean cari (
688    String[] data, int cari) {
689    for (int i = 0; i < data.length; i++) {
690        if (data[i].equals(cari)) {
691            return true;
692        }
693    }
694    return false;
695 }
696
697 // mencari ke belakang (60)
698
699 public static boolean cari (
700    String[] data, int cari) {
701    for (int i = data.length - 1; i >= 0; i--) {
702        if (data[i].equals(cari)) {
703            return true;
704        }
705    }
706    return false;
707 }
708
709 // mencari ke depan (61)
710
711 public static boolean cari (
712    String[] data, int cari) {
713    for (int i = 0; i < data.length; i++) {
714        if (data[i].equals(cari)) {
715            return true;
716        }
717    }
718    return false;
719 }
720
721 // mencari ke belakang (62)
722
723 public static boolean cari (
724    String[] data, int cari) {
725    for (int i = data.length - 1; i >= 0; i--) {
726        if (data[i].equals(cari)) {
727            return true;
728        }
729    }
730    return false;
731 }
732
733 // mencari ke depan (63)
734
735 public static boolean cari (
736    String[] data, int cari) {
737    for (int i = 0; i < data.length; i++) {
738        if (data[i].equals(cari)) {
739            return true;
740        }
741    }
742    return false;
743 }
744
745 // mencari ke belakang (64)
746
747 public static boolean cari (
748    String[] data, int cari) {
749    for (int i = data.length - 1; i >= 0; i--) {
750        if (data[i].equals(cari)) {
751            return true;
752        }
753    }
754    return false;
755 }
756
757 // mencari ke depan (65)
758
759 public static boolean cari (
760    String[] data, int cari) {
761    for (int i = 0; i < data.length; i++) {
762        if (data[i].equals(cari)) {
763            return true;
764        }
765    }
766    return false;
767 }
768
769 // mencari ke belakang (66)
770
771 public static boolean cari (
772    String[] data, int cari) {
773    for (int i = data.length - 1; i >= 0; i--) {
774        if (data[i].equals(cari)) {
775            return true;
776        }
777    }
778    return false;
779 }
780
781 // mencari ke depan (67)
782
783 public static boolean cari (
784    String[] data, int cari) {
785    for (int i = 0; i < data.length; i++) {
786        if (data[i].equals(cari)) {
787            return true;
788        }
789    }
790    return false;
791 }
792
793 // mencari ke belakang (68)
794
795 public static boolean cari (
796    String[] data, int cari) {
797    for (int i = data.length - 1; i >= 0; i--) {
798        if (data[i].equals(cari)) {
799            return true;
800        }
801    }
802    return false;
803 }
804
805 // mencari ke depan (69)
806
807 public static boolean cari (
808    String[] data, int cari) {
809    for (int i = 0; i < data.length; i++) {
810        if (data[i].equals(cari)) {
811            return true;
812        }
813    }
814    return false;
815 }
816
817 // mencari ke belakang (70)
818
819 public static boolean cari (
820    String[] data, int cari) {
821    for (int i = data.length - 1; i >= 0; i--) {
822        if (data[i].equals(cari)) {
823            return true;
824        }
825    }
826    return false;
827 }
828
829 // mencari ke depan (71)
830
831 public static boolean cari (
832    String[] data, int cari) {
833    for (int i = 0; i < data.length; i++) {
834        if (data[i].equals(cari)) {
835            return true;
836        }
837    }
838    return false;
839 }
840
841 // mencari ke belakang (72)
842
843 public static boolean cari (
844    String[] data, int cari) {
845    for (int i = data.length - 1; i >= 0; i--) {
846        if (data[i].equals(cari)) {
847            return true;
848        }
849    }
850    return false;
851 }
852
853 // mencari ke depan (73)
854
855 public static boolean cari (
856    String[] data, int cari) {
857    for (int i = 0; i < data.length; i++) {
858        if (data[i].equals(cari)) {
859            return true;
860        }
861    }
862    return false;
863 }
864
865 // mencari ke belakang (74)
866
867 public static boolean cari (
868    String[] data, int cari) {
869    for (int i = data.length - 1; i >= 0; i--) {
870        if (data[i].equals(cari)) {
871            return true;
872        }
873    }
874    return false;
875 }
876
877 // mencari ke depan (75)
878
879 public static boolean cari (
880    String[] data, int cari) {
881    for (int i = 0; i < data.length; i++) {
882        if (data[i].equals(cari)) {
883            return true;
884        }
885    }
886    return false;
887 }
888
889 // mencari ke belakang (76)
890
891 public static boolean cari (
892    String[] data, int cari) {
893    for (int i = data.length - 1; i >= 0; i--) {
894        if (data[i].equals(cari)) {
895            return true;
896        }
897    }
898    return false;
899 }
900
901 // mencari ke depan (77)
902
903 public static boolean cari (
904    String[] data, int cari) {
905    for (int i = 0; i < data.length; i++) {
906        if (data[i].equals(cari)) {
907            return true;
908        }
909    }
910    return false;
911 }
912
913 // mencari ke belakang (78)
914
915 public static boolean cari (
916    String[] data, int cari) {
917    for (int i = data.length - 1; i >= 0; i--) {
918        if (data[i].equals(cari)) {
919            return true;
920        }
921    }
922    return false;
923 }
924
925 // mencari ke depan (79)
926
927 public static boolean cari (
928    String[] data, int cari) {
929    for (int i = 0; i < data.length; i++) {
930        if (data[i].equals(cari)) {
931            return true;
932        }
933    }
934    return false;
935 }
936
937 // mencari ke belakang (80)
938
939 public static boolean cari (
940    String[] data, int cari) {
941    for (int i = data.length - 1; i >= 0; i--) {
942        if (data[i].equals(cari)) {
943            return true;
944        }
945    }
946    return false;
947 }
948
949 // mencari ke depan (81)
950
951 public static boolean cari (
952    String[] data, int cari) {
953    for (int i = 0; i < data.length; i++) {
954        if (data[i].equals(cari)) {
955            return true;
956        }
957    }
958    return false;
959 }
960
961 // mencari ke belakang (82)
962
963 public static boolean cari (
964    String[] data, int cari) {
965    for (int i = data.length - 1; i >= 0; i--) {
966        if (data[i].equals(cari)) {
967            return true;
968        }
969    }
970    return false;
971 }
972
973 // mencari ke depan (83)
974
975 public static boolean cari (
976    String[] data, int cari) {
977    for (int i = 0; i < data.length; i++) {
978        if (data[i].equals(cari)) {
979            return true;
980        }
981    }
982    return false;
983 }
984
985 // mencari ke belakang (84)
986
987 public static boolean cari (
988    String[] data, int cari) {
989    for (int i = data.length - 1; i >= 0; i--) {
990        if (data[i].equals(cari)) {
991            return true;
992        }
993    }
994    return false;
995 }
996
997 // mencari ke depan (85)
998
999 public static boolean cari (
1000    String[] data, int cari) {
1001    for (int i = 0; i < data.length; i++) {
1002        if (data[i].equals(cari)) {
1003            return true;
1004        }
1005    }
1006    return false;
1007 }
1008
1009 // mencari ke belakang (86)
1010
1011 public static boolean cari (
1012    String[] data, int cari) {
1013    for (int i = data.length - 1; i >= 0; i--) {
1014        if (data[i].equals(cari)) {
1015            return true;
1016        }
1017    }
1018    return false;
1019 }
1020
1021 // mencari ke depan (87)
1022
1023 public static boolean cari (
1024    String[] data, int cari) {
1025    for (int i = 0; i < data.length; i++) {
1026        if (data[i].equals(cari)) {
1027            return true;
1028        }
1029    }
1030    return false;
1031 }
1032
1033 // mencari ke belakang (88)
1034
1035 public static boolean cari (
1036    String[] data, int cari) {
1037    for (int i = data.length - 1; i >= 0; i--) {
1038        if (data[i].equals(cari)) {
1039            return true;
1040        }
1041    }
1042    return false;
1043 }
1044
1045 // mencari ke depan (89)
1046
1047 public static boolean cari (
1048    String[] data, int cari) {
1049    for (int i = 0; i < data.length; i++) {
1050        if (data[i].equals(cari)) {
1051            return true;
1052        }
1053    }
1054    return false;
1055 }
1056
1057 // mencari ke belakang (90)
1058
1059 public static boolean cari (
1060    String[] data, int cari) {
1061    for (int i = data.length - 1; i >= 0; i--) {
1062        if (data[i].equals(cari)) {
1063            return true;
1064        }
1065    }
1066    return false;
1067 }
1068
1069 // mencari ke depan (91)
1070
1071 public static boolean cari (
1072    String[] data, int cari) {
1073    for (int i = 0; i < data.length; i++) {
1074        if (data[i].equals(cari)) {
1075            return true;
1076        }
1077    }
1078    return false;
1079 }
1080
1081 // mencari ke belakang (92)
1082
1083 public static boolean cari (
1084    String[] data, int cari) {
1085    for (int i = data.length - 1; i >= 0; i--) {
1086        if (data[i].equals(cari)) {
1087            return true;
1088        }
1089    }
1090    return false;
1091 }
1092
1093 // mencari ke depan (93)
1094
1095 public static boolean cari (
1096    String[] data, int cari) {
1097    for (int i = 0; i < data.length; i++) {
1098        if (data[i].equals(cari)) {
1099            return true;
1100        }
1101    }
1102    return false;
1103 }
1104
1105 // mencari ke belakang (94)
1106
1107 public static boolean cari (
1108    String[] data, int cari) {
1109    for (int i = data.length - 1; i >= 0; i--) {
1110        if (data[i].equals(cari)) {
1111            return true;
1112        }
1113    }
1114    return false;
1115 }
1116
1117 // mencari ke depan (95)
1118
1119 public static boolean cari (
1120    String[] data, int cari) {
1121    for (int i = 0; i < data.length; i++) {
1122        if (data[i].equals(cari)) {
1123            return true;
1124        }
1125    }
1126    return false;
1127 }
1128
1129 // mencari ke belakang (96)
1130
1131 public static boolean cari (
1132    String[] data, int cari) {
1133    for (int i = data.length - 1; i >= 0; i--) {
1134        if (data[i].equals(cari)) {
1135            return true;
1136        }
1137    }
1138    return false;
1139 }
1140
1141 // mencari ke depan (97)
1142
1143 public static boolean cari (
1144    String[] data, int cari) {
1145    for (int i = 0; i < data.length; i++) {
1146        if (data[i].equals(cari)) {
1147            return true;
1148        }
1149    }
1150    return false;
1151 }
1152
1153 // mencari ke belakang (98)
1154
1155 public static boolean cari (
1156    String[] data, int cari) {
1157    for (int i = data.length - 1; i >= 0; i--) {
1158        if (data[i].equals(cari)) {
1159            return true;
1160        }
1161    }
1162    return false;
1163 }
1164
1165 // mencari ke depan (99)
1166
1167 public static boolean cari (
1168    String[] data, int cari) {
1169    for (int i = 0; i < data.length; i++) {
1170        if (data[i].equals(cari)) {
1171            return true;
1172        }
1173    }
1174    return false;
1175 }
1176
1177 // mencari ke belakang (100)
1178
1179 public static boolean cari (
1180    String[] data, int cari) {
1181    for (int i = data.length - 1; i >= 0; i--) {
1182        if (data[i].equals(cari)) {
1183            return true;
1184        }
1185    }
1186    return false;
1187 }
1188
1189 // mencari ke depan (101)
1190
1191 public static boolean cari (
1192    String[] data, int cari) {
1193    for (int i = 0; i < data.length; i++) {
1194        if (data[i].equals(cari)) {
1195            return true;
1196        }
1197    }
1198    return false;
1199 }
1200
1201 // mencari ke belakang (102)
1202
1203 public static boolean cari (
1204    String[] data, int cari) {
1205    for (int i = data.length - 1; i >= 0; i--) {
1206        if (data[i].equals(cari)) {
1207            return true;
1208        }
1209    }
1210    return false;
1211 }
1212
1213 // mencari ke depan (103)
1214
1215 public static boolean cari (
1216    String[] data, int cari) {
1217    for (int i = 0; i < data.length; i++) {
1218        if (data[i].equals(cari)) {
1219            return true;
1220        }
1221    }
1222    return false;
1223 }
1224
1225 // mencari ke belakang (104)
1226
1227 public static boolean cari (
1228    String[] data, int cari) {
1229    for (int i = data.length - 1; i >= 0; i--) {
1230        if (data[i].equals(cari)) {
1231            return true;
1232        }
1233    }
1234    return false;
1235 }
1236
1237 // mencari ke depan (105)
1238
1239 public static boolean cari (
1240    String[] data, int cari) {
1241    for (int i = 0; i < data.length; i++) {
1242        if (data[i].equals(cari)) {
1243            return true;
1244        }
1245    }
1246    return false;
1247 }
1248
1249 // mencari ke belakang (106)
1250
1251 public static boolean cari (
1252    String[] data, int cari) {
1253    for (int i = data.length - 1; i >= 0; i--) {
1254        if (data[i].equals(cari)) {
1255            return true;
1256        }
1257    }
1258    return false;
1259 }
1260
1261 // mencari ke depan
```