APLIKASI PENCARIAN KATA DALAM BLOK HURUF ACAK

Gigih Istiawan, Irving Vitra Paputungan, Teduh Dirgahayu, Taufiq Hidayat
Laboratorium Komputasi dan Sistem Cerdas, Jurusan Teknik Informatika, Fakultas Teknologi Industri
Universitas Islam Indonesia, Yogyakarta
E-mail: ipink@engineer.com

ABSTRAKSI

Makalah ini akan menjelaskan bagaimana aplikasi pencarian kata dalam blok acak. Kata-kata yang dicari terletak di dalam blok huruf yang merupakan matrik yang terdiri dari kumpulan huruf acak. Dalam pencariannya program menggunakan teknik pencarian berurut (sequential searching) untuk mencari kata yang dicari. Hasil pencarian berupa koordinat tiap huruf pembentuk kata yang dicari di dalam blok huruf.

***Kata kunci**: matrik, pencarian, pencarian berurut*

1. PENDAHULUAN

Program pencarian kata ini dibuat untuk menyelesaikan masalah yang diberikan dalam Kontes Pemrograman Informatika (KPI) 2006 yang diselenggarakan oleh Jurusan Teknik Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia.

Masalah yang diberikan yaitu bagaimana mencari kata dalam sebuah blok huruf acak dengan menentukan posisi dari masing-masing huruf pembentuk kata. Kata yang dicari dapat berposisi vertikal, horizontal, diagonal dan juga dapat di baca secara terbalik atau dari kanan ke kiri. Dalam prosesnya program dibantu dengan fileteks untuk menyimpan kata-kata yang dicari dan hasil dari pencarian kata. Program yang dibuat harus dapat menangani blok huruf sampai ukuran 50 x 50.

Hasil yang diinginkan dari program ini adalah program dapat menentukan koordinat huruf-huruf dari kata yang dicari di dalam blok huruf, yang ditampilkan dan disimpan di dalam fileteks. Dalam pengembangannya sistem menggunakan pemrograman terstruktur untuk menyelesaikan masalah yang dihadapi.

2. LANDASAN TEORI

Array merupakan kumpulan variabel yang bertipe sama [2]. Sebagai contoh untuk mendeklarasikan lima variabel bertipe integer maka akan dilakukan dengan cara berikut:

```
Var x1, x2, x3, x4, x5: integer;
```

Dengan menggunakan array hal ini dapat dipersingkat dengan cara sebagai berikut:

```
Var x: array[1..5] of integer;
```

Dengan deklarasi array seperti di atas maka kita telah memiliki variabel x[1], x[2], x[3], x[4] dan x[5]. Array dapat digunakan untuk membentuk matrik ukuran m x n dengan menggunakan array multidimensi. Untuk membentuk matrik ukuran 3 x 3 dapat dideklarasikan array dua dimensi sebagai berikut:

```
Var X: array [1..3, 1..3] of integer;
```

Variabel di atas akan tampak seperti gambar di bawah ini:

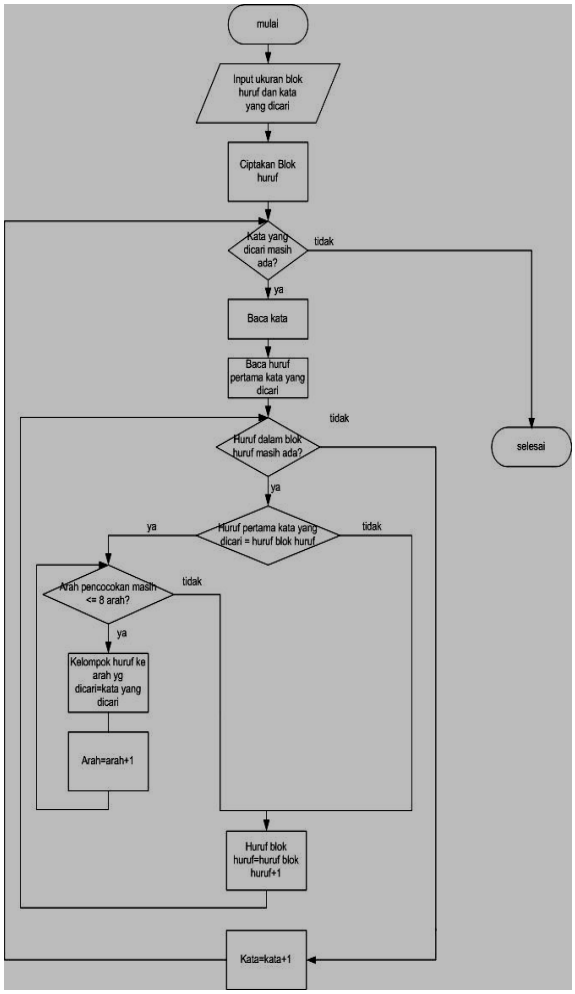
X[1,1]	X[1,2]	X[1,3]
X[2,1]	X[2,2]	X[2,3]
X[3,1]	X[3,2]	X[3,3]

Gambar 1. Visualisasi array dua dimensi

Yang menentukan cepat atau tidaknya program ini dapat berjalan adalah proses pencariannya. Jika data berhubungan dengan data yang sedikit maka proses pencarian tidak akan berpengaruh tetapi jika berhubungan dengan data yang sangat banyak maka proses pencarian ini akan sangat berpengaruh pada cepat lambatnya program. Pada pencarian ada beberapa kelompok yaitu metode pencarian yang dilakukan untuk data yang sudah terurut dan data yang belum terurut. Untuk data yang sudah terurut metode yang dipakai antara lain pencarian berurutan (*sequential searching*), pencarian biner (*binary search*) dan pencarian berurutan berindex (*indexed sequential search*) sedangkan untuk data yang tidak terurut salah satunya adalah *sequential searching.*

3. METODE PERANCANGAN

Blok huruf dalam program ini menggunakan matrik ukuran m x n untuk merepresentasikan kumpulan huruf acak. Dalam pencarian kata dalam blok huruf digunakan metode pencarian berurut (*sequential searching*) yang dilakukan dengan cara membandingkan data satu per satu sampai data tersebut ditemukan atau tidak ditemukan. Pada saat data yang dicari sudah ketemu maka proses pencarian langsung dihentikan. Tetapi jika data yang dicari belum ketemu maka pencarian diteruskan sampai seluruh data dibandingkan [3]. **Gambar 2** berikut ini adalah diagram alur dari program:



Gambar 2. Diagram Alur Perancangan

Pencarian dalam program dimulai dengan mencari huruf pertama dari kata yang dicari dalam blok huruf. Jika telah ditemukan maka program akan memeriksa ke delapan penjuru mata angin untuk mencocokkan kata yang dicari. Dalam mencocokkan ke delapan penjuru mata angin ini terdapat batasan-batasan yaitu jika posisi suatu huruf yang telah ditemukan dalam blok huruf tidak memungkinkan untuk memeriksa ke suatu arah maka tidak akan dilakukan pencocokan ke arah tersebut.

Masing-masing arah akan dilihat apakah jumlah banyaknya huruf ke suatu arah lebih banyak atau sama dengan banyak huruf dalam kata yang dicari jika betul maka akan dilakukan pemeriksaan dan jika tidak maka pencarian akan dilakukan terhadap arah lain.

Masing-masing pemeriksaan ke suatu arah akan diperiksa huruf per huruf jika huruf berikutnya tidak cocok dengan kata yang dicari maka akan dilanjutkan ke arah yang lain. Selanjutnya setelah suatu huruf selesai diperiksa ke delapan arah maka program akan mencari huruf pertama lagi yang cocok dengan huruf pertama kata yang dicari dalam blok huruf dan kemudian dilakukan pemeriksaan seperti sebelumnya. Jika sudah tidak ditemukan lagi huruf yang cocok dengan huruf pertama kata yang dicari maka program akan mulai berpindah mencari kata berikutnya. Berikut ini adalah algoritma pembuatan blok huruf dan pencarian tiap kata dalam program:

```
1.  UkMatR_brs ← b {jml baris matrik
        random}
2.  UkMatR_klm ← k {jml kolom matrik
        random}
3.  I ← 1
4.  Selama i <= UkMatR_brs kerjakan langkah
        5 sampai 10 (membuat matrik
        random}
5.      j ← 1
6.      Selama i <= UkMatR_klm kerjakan
          langkah 7 sampai 9
7.          temp ← random(26)
8.          MatRandom[baris,kolom] <-
          MatAlfabet[temp] {matalfabet
          adalah matrik
             Yang berisi alfabet dari A..Z}
9.          j ← j+1
10.    I ← i+1
11. baris ← 1
12. ada ← false
13. tdklengkap ← true
14. selama baris <= UkMatR_brs kerjakan
        langkah 15 sampai 47
15.   kolom ← 1
16.   selama kolom <= UkMatR_klm kerjakan
        langkah 17 sampai 46
        {Pengecekan karakter pertama yang
        dicari apakah ada dlm matrik
        random}
17.     jika
        MatRandom[baris,kolom]=MatKatCari[
        0] maka kerjakan langka 18 s/d 45
18.       jika y=1 maka kerjakan langkah 19
        sampai 22 {y adalah panjang
        karakter
         kata yang dicari}
19.         ada ← true
20.         tdklengkap ← true
21.         b ← baris
22.         k ← kolom
23.       jika y>1 maka kerjakan langkah 24
        sampai 45
        {jika kata yang dicari panjangnya
        lebih dari satu maka akan
        dilakukan pengecekan ke arah 8
        penjuru mata angin}
24.         jika baris > y maka kerjakan
        langkah 25 sampai 38
25.           bar ← baris
26.           kol ← kolom
27.           cocok ← true
28.           v ← 2
29.           *bar ← bar - 1
30.           selama cocok=true dan v<=y
        kerjakan langkah 31 sampai 34
31.             jika MatRandom[bar,kol] <>
        MatKata[v]
32.               maka cocok ← false
33.             v ← v + 1
34.             *bar ← bar - 1
35.           jika cocok=true kerjakan maka
        langkah 36 sampai 38
36.             b ← baris
37.             k ← kolom
38.             tdklengkap ← false
    {langkah 39 s/d 45 mengerjakan hal yang
    sama seperti langkah 25
    sampai 38 hanya mengganti inisialisasi
    yang diberi tanda " * "}
39.         jika baris >= y dan kolom <=
        UkMatR_klm – y + 1 maka kerjakan
          langkah 25 sampai 38 dengan
              b ← b-1
```

 ISSN: 1907-5022

```
                k ← k+1
40.           jika kolom <= ukMatR_klm - y + 1
            maka kerjakan langkah 25 s/d 38
                k ← k+1
41.           jika baris <= ukMatR_brs - y + 1
            dan kolom <= ukMatR_klm - y + 1)
              maka kerjakan langkah 25 s/d 38
                b ← b+1
               k ← k+1
42.           jika baris <= ukMatR_brs - y + 1
            maka kerjakan langkah 25 s/d 38
                b ← b+1
43.           jika kolom >= y dan baris <=
            ukMatR_brs - y + 1 maka kerjakan
              langkah 25 s/d 38
                b ← b+1
               k ← k-1
44.           jika kolom >= y maka kerjakan
            langkah 25 s/d 38
                k ← k-1
45.           jika kolom >= y dan baris >= y
            maka kerjakan langkah 25 s/d 38
                b ← b-1
               k ← k-1
46.       kolom ← kolom + 1
47.     baris ← baris + 1
48. jika tdklengkap=true atau ada=false maka
          kerjakan langkah 49 s/d 50
49.     b ← baris
    k ← kolom
```

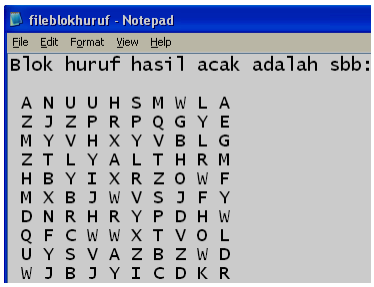
4. IMPLEMENTASI

Program dibuat dengan menggunakan *console* milik Borland Delphi 7. Untuk tampilan program sampai program selesai dijalankan adalah sebagai berikut:

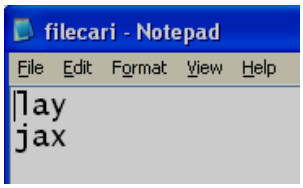
Gambar 3. Tampilan program

Dalam program di atas pertama kali muncul akan menanyakan jumlah baris dan kolom yang akan dipakai untuk membentuk blok huruf acak. Setelah mendapat masukan maka program akan mulai membuat blok huruf dan hasilnya akan ditampilkan dalam program dan disimpan dalam fileblokhuruf.dat.

Setelah itu program akan meminta masukan kata yang dicari. Untuk memasukkan kata yang dicari, masing-masing kata diketik diakhiri dengan enter. Jika akan mengakhiri mendaftar kata yang dicari maka ketik spasi lalu enter setelah kata sebelumnya. Kata-kata yang dicari tersimpan dalam filecari.dat.

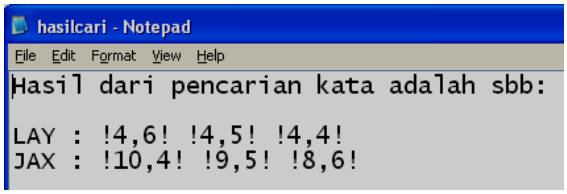


Gambar 4. fileblokhuruf.dat



Gambar 5. filecari.dat

Hasil pencarian akan ditampilkan dalam program dan disimpan di hasilcari.dat dalam bentuk kata yang dicari disertai dengan posisinya.



Gambar 6. Hasilcari.dat

5. KESIMPULAN

Program ini telah dapat menunjukkan posisi kata yang dicari dalam blok huruf yang dicari dengan benar tetapi kekurangannya program ini belum bisa menerima input blok huruf. Program ini merandom sendiri blok hurufnya kemudian kita tentukan kata yang dicari setelah blok huruf program terbentuk. Hal itu sangat menyusahkan karena selain kita harus mencari dulu kata yang ingin dicari dalam blok huruf random, susunan huruf yang membentuk kata sangat jarang ditemukan kecuali kata kata pendek. Akan lebih baik jika sebuah blok huruf yang telah mengandung kata-kata yang ingin dicari diinputkan ke program dalam bentuk fileteks sehingga program lebih nyaman digunakan. Akan lebih baik jika program ini dikembangkan dengan pemrograman dengan menggunakan objek visual sehingga mudah dimengerti.

DAFTAR PUSTAKA

[1] Jogiyanto, H. M. *Pengenalan Komputer*. Yogyakarta: Andi, 2002.
[2] Pranata, A. *Pemrograman Borland Delphi 6*. Yogyakarta: Andi, 2003.
[3] Santosa, P. I. *Struktur Data Menggunakan Turbo Pascal 6.0*. Yogyakarta: Andi, 1997.
[4] Sutedjo, B., dan Michael, A. N. *Algoritma dan Teknik Pemrograman*. Yogyakarta: Andi, 2004.