MODUL MATA KULIAH

ANALISIS DAN DESAIN ALGORITMA

PG167 – 3 SKS

FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BUDI LUHUR

JAKARTA
SEPTEMBER 2019

TIM PENYUSUN

Dr. Achmad Solichin, S.Kom., M.T.I
Atik Ariesta, S.Kom., M.Kom
Ita Novita, S.Kom., M.T.I

UNIVERSITAS BUDI LUHUR
FAKULTAS TEKNOLOGI INFORMASI

MODUL PERKULIAHAN #15
MANIPULASI ARRAY KARAKTER (STRING)

Capaian Pembelajaran	:	Mahasiswa dapat memahami konsep manipulasi array satu dimensi, dan melakukan penggabungan dua atau lebih array menjadi array satu dimensi.
Sub Pokok Bahasan	:	15.1. Konsep manipulasi array karakter (string). 15.2. Manipulasi array karakter: teknik penelusuran, penggabungan (merge) dan pemecahan (split) array satu dimensi yang bertipe karakter.
Daftar Pustaka	:	1. Gaddis, nd.2011. Starting Out with C++ from Control Structures through Objects .8th. Boston: Addison-Wesley. 2. Institue of Distance & Open Learning, n.d. UNIT l Algorithms, Flowcharts & Program Design in: INTRODUCTION TO C++. p. 205 3. Sjukani,Moh .2014. Algoritma (Algoritma & Struktur Data 1) Dengan C, C++, dan Java Edisi 9", Mitra Wacana Media.

MANIPULASI ARRAY KARAKTER (STRING)

15.1. KONSEP MANIPULASI ARRAY KARAKTER (STRING)

Karakter dalam pemrograman merupakan representasi dari satu buah huruf, angka atau karakter tertentu. Setiap pemrograman memiliki tipe data khusus untuk menyimpan karakter, misalnya tipe data **char** di Bahasa C/C++ dan Java. Sementara itu, kumpulan dari karakter sering disebut sebagai **string**. Umumnya, bahasa pemrograman tidak menyediakan tipe data khusus untuk menyimpan string (kumpulan karakter). Namun ada beberapa yang menyediakan tipe data bentukan untuk string, seperti di Java menfasilitasi tipe data string dengan membentuk class String.

Di bahasa pemrograman C/C++ sendiri, tidak memiliki tipe data khusus untuk menyimpan string. Namun demikian, penanganan string dilakukan melalui struktur data array satu dimensi. Dengan kata lain, untuk menyimpan string atau teks, kita perlu membentuk array bertipe karakter (char) sejumlah karakter dari string yang akan disimpan.

Sebagai contoh, jika kita ingin mendeklarasikan suatu variabel C yang diisi dengan string "UBL", maka dapat dilakukan dengan salah satu dari potongan perintah berikut ini:

```
1. char C[3] = "UBL";
2. char C[3] = {'U','B','L'};
3. char C[3]; strcpy(C,"UBL"); //cara ini membutuhkan header
   fungsi string.h
```

15.2. MANIPULASI ARRAY KARAKTER (STRING)

Pada bagian ini dijelaskan mengenai beberapa contoh manipulasi array karakter (string). Manipulasi array karakter dapat berupa penelusuran array, pencarian isi array, pengurutan, penggabungan (merge) atau pemecahan (split). Contoh 15.1 di bawah ini merupakan contoh soal yang terkait dengan penelusuran dan pencarian isi array.

CONTOH SOAL 15.1 – PENELUSURAN ARRAY

Sudah ada dua buah array satu dimensi **A** dan **B** yang dibuat dengan **char A[6]** dan **char B[8],** sudah ada isinya dengan huruf-huruf kapital tanpa spasi. Ilustrasinya sebagai berikut:

	0	1	2	3	4	5
A	B	O	G	O	**R**	**\0**

	0	1	2	3	4	5	6	7
B	J	A	K	A	**R**	T	A	\0

Susun program untuk memeriksa apakah diantara isi array A ada yang sama dengan isi array B. Bila ada, cetak perkataan "**ADA**", bila tidak ada cetak perkataan "**TIDAK ADA**". Pada contoh di atas, maka akan tercetak perkataan "ADA" karena terdapat huruf yang sama diantara dua array, yaitu huruf R.

Analisis dan Penyelesaian Soal

Untuk menyelesaikan soal 15.1 di atas, kita harus melakukan penelusuran setiap karakter yang tersimpan pada array A[], lalu dilakukan pencarian apakah karakter tersebut ada di array B[] atau tidak. Berikut ini algoritme atau kerangka pikir untuk menyelesaikan soal 15.1 di atas:

1. Inisialisasi array A[] dan isinya
2. Inisialisasi array B[] dan isinya
3. Inisialisasi variabel flag = 0 untuk menyimpan status pencarian (0 = tidak ditemukan, 1 = ditemukan).
4. Lakukan penelusuran isi array A[], lalu lakukan penelusuran dan pemeriksaan isi dari B[]. Jika terdapat karakter dari array A[] yang sama dengan karakter dari array B[], maka ubah flag menjadi 1 dan hentikan penelusuran. Karena soal hanya meminta status "ADA" atau "TIDAK ADA", maka saat ditemukan, penelusuran tidak perlu dilanjutkan.
5. Jika isi flag = 0 maka cetak "TIDAK ADA", namun jika flag = 1 maka cetak "ADA".

Berdasarkan algoritme (kerangka pikir) di atas, selanjutnya dapat diimplementasikan ke dalam bentuk program seperti pada Program 1 berikut ini.

Program 1. Penelusuran dan pencarian karakter pada array satu dimensi.

```
1  #include <stdio.h>
2  int main()
3  {
4      char A[6] = "BOGOR";
5      char B[8] = "JAKARTA";
6      int I, J, flag=0;
7
8      //cetak isi array A
9      printf("Isi Array A : ");
10     for (I=0; I<5; I++) {
11         printf("%3c", A[I]);
12     }
13     //cetak isi array B
14     printf("\nIsi Array B : ");
15     for (I=0; I<7; I++) {
16         printf("%3c", B[I]);
17     }
18
19     for (J=0; J<5; J++) {
20         for (I=0; I<7; I++) {
21             if (A[J] == B[I]) {
22                 flag = 1; break;
23             }
24         }
25     }
26
27     if(flag==1) {
28         printf("\nADA");
29     } else {
30         printf("\nTIDAK ADA");
31     }
32
33     return 0;
34 }
```

Penjelasan Program 1:

- Pada baris 4-6 dilakukan deklarasi array A[] dan B[] lengkap dengan isinya, serta beberapa variabel yang digunakan dalam program.
- Pada baris 8-17, program melakukan pencetakan isi dari array A[] dan B[] agar saat dijalankan kita dapat mengetahui isi array aslinya.

- Baris 19-25 merupakan proses penelusuran isi dari array A[]. Untuk setiap isi dari array A[], lakukan penelusuran ke array B[]. Jika terdapat karakter / isi dari array A[] yang sama dengan isi dari array B[] maka, ubah nilai variabel **flag** menjadi 1 (artinya ditemukan karakter yang sama). Dan selanjutnya keluar dari perulangan / penelusuran dengan perintah **break**. *Mengapa harus ditambahkan perintah break? Apakah jika tidak menggunakan perintah break, program tetap benar?*
- Bagian akhir program, yaitu baris 27-31, dilakukan pencetakan perkataan "ADA" jika isi flag = 1, dan "TIDAK ADA" jika isi flag = 0.

Selain penelusuran dan pencarian karakter di dalam array satu dimensi, proses pemecahan atau pemisahan string juga sering diperlukan dalam berbagai kasus. Salah satunya adalah proses tokenizing. Proses tokenizing adalah proses pemotongan string masukan berdasarkan tiap kata yang menyusunnya. Pada prinsipnya proses ini adalah memisahkan setiap kata yang menyusun suatu dokumen. Tokenizing umumnya dipakai dalam berbagai algoritma information retrieval, text mining, dan sebagainya.

Perhatikan contoh soal 15.2 berikut ini!

CONTOH SOAL 15.2 – TOKENIZING

Sudah ada sebuah array satu dimensi **A** yang dibuat dengan **char A[1000]** belum ada isinya. Susun program untuk menginput satu kalimat dan menyimpannya di array A[]. Selanjutnya cetak setiap kata dari kalimat yang diinputkan pada baris-baris yang berbeda.

Contoh inputan: `algoritma itu mudah`

Luaran:
```
algoritma
itu
mudah
```

Analisis dan Penyelesaian Soal

Untuk menyelesaikan soal 15.2, pertama-tama kita simpan kalimat yang diinputkan ke array A[]. Selanjutnya lakukan penelusuran setiap karakter yang tersimpan pada array A[], lakukan pencetakan dan jika menemukan karakter spasi, ganti dengan karakter pindah baris (\n). Karakter spasi merupakan karakter nomor 32. Berikut ini algoritme atau kerangka pikir untuk menyelesaikan soal 15.2 di atas:

Program 2. Tokenizing.

```
1  #include "stdio.h"
2  int main()
3  {
4      char A[1000];
5      int i;
6      //input
7      printf("Input kalimat: ");
8      gets(A);
9      i=0;
10     while(true) {
11         if (A[i] == '\0') {
12             break;
13         }
14         if (A[i] != ' ') {
15             printf("%c", A[i]);
16         } else {
17             printf("\n");
18         }
19         i++;
20     }
21     return 0;
22 }
```

SOAL LATIHAN

Soal 1.

Sudah ada dua buah array satu dimensi **A** dan **B** yang dibuat dengan **char A[6]** dan **char B[8].** Array A dan B belum ada isinya.

	0	1	2	3	4	5
A						

	0	1	2	3	4	5	6	7
B								

Susun program menginput isi array A dan B dengan huruf-huruf kapital tanpa spasi. Selanjutnya periksa apakah diantara isi array A ada yang sama dengan isi array B. Bila ada, cetak perkataan "**ADA**" dan cetak huruf yang sama tersebut, bila tidak ada cetak perkataan "**TIDAK ADA**".

Soal 2.

Sudah ada sebuah array satu dimensi **A** yang dibuat dengan **char A[8].** Array A sudah ada isinya, berupa huruf kapital tanpa spasi. Ilustrasinya sebagai berikut:

	0	1	2	3	4	5	6	7
A	J	A	K	A	R	T	A	\0

Susun program untuk memeriksa isi array A, apakah **ada huruf yang sama.** Bila ada, maka cetak perkataan "**ADA**" dan cetak huruf yang sama tersebut, bila tidak ada cetak perkataan "**TIDAK ADA**".

Soal 3.

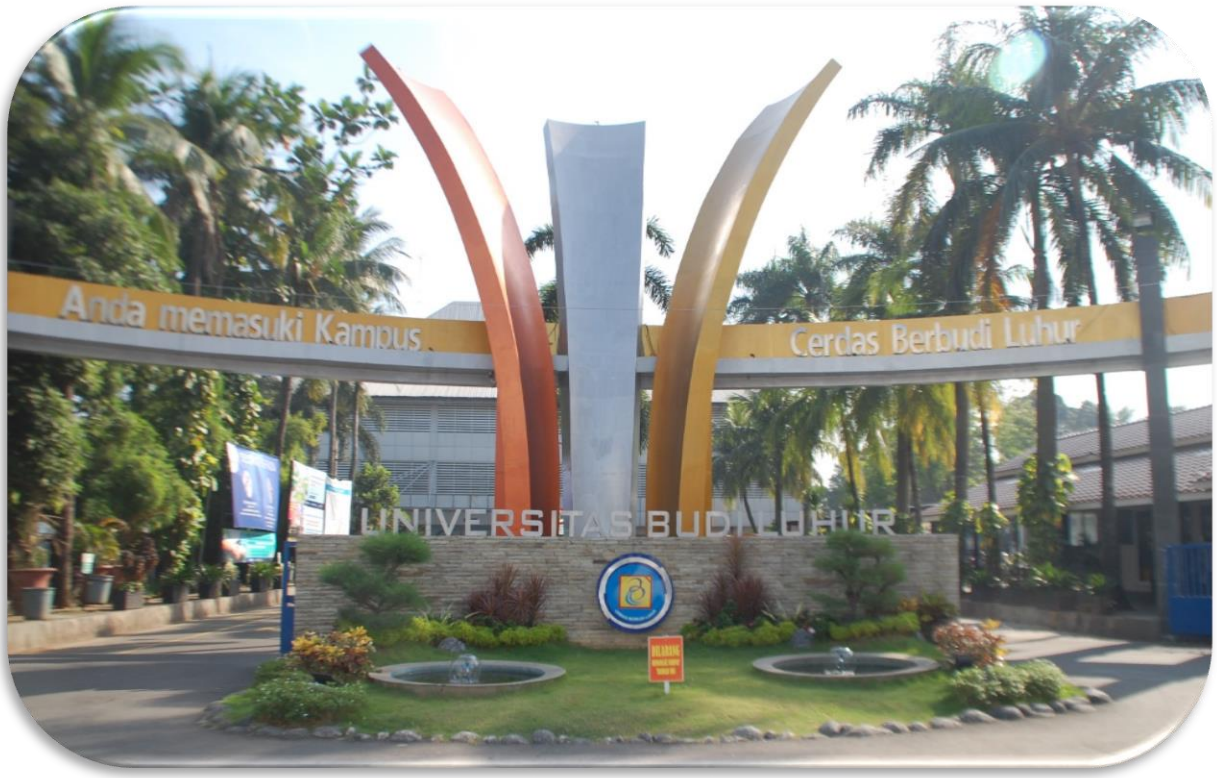
Sudah ada sebuah array satu dimensi **A** yang dibuat dengan **char A[11].** Array A sudah ada isinya, berupa huruf kapital tanpa spasi sebanyak 10 karakter. Ilustrasinya sebagai berikut:

	0	1	2	3	4	5	6	7	8	9	10
A	J	O	G	J	A	K	A	R	T	A	\0

Susun program untuk memeriksa isi array A dan mencetak jumlah huruf yang terbanyak dalam array A tersebut! Sertakan juga jumlah huruf yang terbanyak tersebut. Pada contoh ilustrasi di atas maka akan tercetak: **A 3**

KESIMPULAN

Penanganan atau manipulasi array karakter (string) merupakan salah satu kemampuan mendasar yang harus dikuasai oleh seorang programmer. Pada dasarnya penanganan array karakter (string) tidaklah berbeda dengan penanganan array yang berisi data lainnya. Hanya terdapat beberapa perbedaan dalam menangani data karakter / string seperti dalam hal pengisian variabel.



FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS BUDI LUHUR

Jl. Raya Ciledug, Petukangan Utara, Pesanggrahan

Jakarta Selatan, 12260

Telp: 021-5853753 Fax : 021-5853752

http://fti.budiluhur.ac.id