KEMENTRIAN PENDIDIKAN, KEBUDAYAAN, RISET, DAN TEKNOLOGI
DIREKTORAT JENDERAL PENDIDIKAN TINGGI, RISET, DAN TEKNOLOGI
DIREKTORAT PEMBELAJARAN DAN KEMAHASISWAAN

Kampus Merdeka
INDONESIA JAYA

# ALGORITMA DAN PEMROGRAMAN

[Komang Aryasa| [Pertemuan 27 dan 28]

# Outline


- Komposisi File Teks
- Membaca File Text
- Menulis File Baru
- 
- 

# Konsep Pengolahan Teks

- Teks terdiri atas deretan karakter yang dikenal oleh komputer. Karakter yang dikenal oleh komputer pada umumnya terdiri atas :

- Abjad atau alphabet, yaitu :{ A, B, C, …, Z, a, b, c, …, z }
- Karakter angka : { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 }
- Operator dan karakter khusus : { +, - , *, /, >, =, <, |, spasi, !, “, #, $, %, &, ^, @, (, ), …}
- Karakter kontrol seperti : DEL (delete), STX (start of text), ETX (end of text), LF (line feed), CR (carriage return / Enter), EOLN (end of line), BOF(begin of file), EOF(end of file),  VT (vertical tab), HT (horizontal tab/TAB), dsb.

- **Definisi :** ***Teks (text)*** adalah deretan karakter yang bisa direkam ke dalam suatu ***file*** */ berkas / arsip*.
- Suatu teks bisa terdiri atas beberapa *kata (words).*
- Satu kata terdiri atas beberapa karakter.
- Setiap kata terpisah dari kata lainnya, dipisahkan oleh paling sedikit satu *spasi.*
- Suatu teks dapat terdiri atas beberapa *baris (lines)* yang dibentuk oleh beberapa kata.
- Setiap baris diakhiri oleh marka *end-of-line (EOLN).*
- Suatu teks diawali oleh marka *begin-of-file (BOF)* dan diakhiri dengan marka *end-of-file (EOF).*

- Berbagai operasi yang bisa dilakukan terhadap teks, antara lain:
  1. menghitung berapa karakter yang ada dalam suatu teks.
  2. menghitung berapa kata dalam teks.
  3. menghitung berapa baris dalam teks.
  4. menghitung berapa kalimat dalam teks.
  5. menggabungkan dua teks.
  6. mencari kata tertentu dalam teks.
  7. menyalin teks

  dsb

# Operasi File Teks

- Beberapa instruksi yang berkaitan dengan (file) teks adalah sbb:

  andaikan F adalah variable dari file :

  - mengembalikan penunjuk teks ke posisi awal (BOF) dari file: **reset ( F )**
  - membuka suatu file teks yang ada di storage (disk) :
  - **assign ( F,** nama_file **)** atau **open(**nama-file)
  - membaca satu karakter dari file teks : **read ( F,** char **)**
  - merekam satu karakter ke file teks : **write ( F,** char **)**
  - menyiapkan file untuk merekam teks : **rewrite(F)**
  - menutup file teks : **close ( F )**

# Contoh: Hitung Karakter

**Algoritma Hit_karakter**

**{** menghitung jumlah karakter yang ada dalam suatu file teks **}**

**Definisi Variabel**

**text** F;
**char** k;
**integer** jkar;

**Rincian Langkah**

**assign (** F, "mytext.txt" **);**
**{** kembalikan ke awal file, baca BOF **}**
**read (** F, k **);**
**if (** k != **BOF** )
**then reset (** F **);**
**read (** F, k **);**
**endif.**
**{** baca karakter hingga akhir file **}**
**read (** F, k **);**
jkar ← 0;
**while (** k != **EOF ) do**
{ hindari tanda EOLN }
**if (** k != **EOLN )**
**then** jkar ← jkar + 1;
**endif.**
**read (** F, k **);**
**endwhile.**
**write (** "jumlah karakter (termasuk spasi ) = ", jkar **);**

# Contoh: Jumlah Kata

**Algoritma Hit_kata**
**{** menghitung jumlah kata dalam suatu file teks **}**

**Definisi Variabel**

**text** F;
**char** k1, k2;
**integer** jkata;

**Rincian Langkah**

**assign (** F, "mytext.txt" **);**
**{** kembalikan ke awal file, baca BOF **}**
**read (** F, k1 **);**
**if (** k1 != **BOF** )
**then reset (** F **);**
**read (** F, k1 **);**
**endif.**

**}**
**{** baca dua karakter ber-urutan hingga akhir file

**read (** F, k1 **);**
**read (** F, k2 **);**
jkata ← 0;
**while (** k1 != **EOF ) || (** k2 != **EOF )**
**do**
**if (** k1 != ' ' **) && (** k2 = ' ' **)**
**then** jkata ← jkata + 1;
**endif.**
k1 ← k2;
**read (** F, k2 **);**
**endwhile.**
**if (** k1 != ' ' **) && (** k2 = **EOF )**
**then** jkata ← jkata + 1;
**endif.**
**write (** "jumlah kata = ", jkata **);**

# File dalam C++

- C++ memakai istilah "stream" untuk menyatakan aliran karakter (teks), apabila aliran ini menuju program maka disebut "input stream" dan bila meninggalkan program disebut "output stream".
- C++ menyediakan 2 macam header file untuk menangani stream, yaitu:
  - **<iostream>** untuk keyboard (cin - input) dan monitor (cout - output)
  - **<fstream>** untuk file, dengan tipe **ifstream** (input) dan **ofstream** (output)

# Membuka dan menutup File

- **#include <fstream>**

  **ifstream in_file;  // definisi input file**

  **ofstream out_file;  // definisi output file**

  **in_file.open("mytext.dat"); // in-stream**

  **out_file.open("outfile.dat"); //out-stream**

- **Tutup file dengan:**

  **in_file.close();**

  **out_file.close();**

# Modus buka file

bentuk fungsi open file:
void open(char **filename,* int *mode,* int access);
dimana: mode adalah:

- ios::app - utk menambah rekaman (append)
- ios::ate - buka file diakhir file (at the end)
- ios::in - buka sebagai in-stream (default untuk in-stream)
- ios::nocreate – jangan ciptakan file baru
- ios::noreplace – jangan ganti isinya
- ios::out - buka sebagai out-stream (default utk out-stream)
- ios::trunc – buka dengan overwrite isi-nya

# atribut access

- kode yang menunjukkan bagaimana akses terhadap file:

| | |
|---|---|
| 0 | buka dengan akses normal (default) |
| 1 | buka dengan akses read-only |
| 2 | buka sebagai hidden file |
| 4 | buka sebagai system file |
| 8 | file archive bit set |

- contoh: ofstream out;

out.open(“text.dat”, ios::out, 0);

sebenarnya sama dengan: out.open(“text.dat”);
karena modus dan access-nya semua default.

# Membaca dan Merekam

- Membaca dari in-stream: mirip dengan pemakaian ***cin >> variable***
- Membaca dari file in_stream ke suatu variabel adalah sbb: ***in_file >> variabel;***
- Merekam ke out-stream: mirip dengan pemakaian **cout << variable**
- Merekam nilai variabel ke file out-stream adalah sbb: ***out_file << variabel;***

# get dan put pada stream

- pada *ifstream* terdapat satu pointer ke elemen berikut-nya yang akan dibaca, yaitu melalui: **get,** sehingga pembacaan dapat dilakukan dengan perintah: *in_stream.get(var)*
- pada *ofstream* terdapat pointer yang menunjuk lokasi dimana elemen akan ditulis, yaitu melalui: **put,** sehingga tampilan bisa ditulis atau direkam dengan perintah: *out_stream.put(var)*

# tellg() dan tellp()

- fungsi tanpa parameter **tellg()** memberikan nilai integer dari posisi dimana elemen akan dibaca dengan perintah in_stream.get(var)
- fungsi tanpa parameter **tellp()** memberikan nilai integer dari posisi dimana elemen akan direkam/ditulis

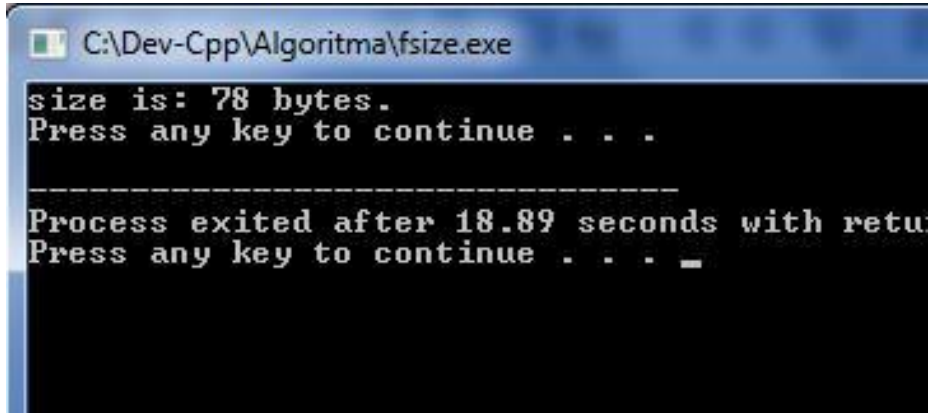
# seekg() dan seekp()

- **seekg(position) :** mencari posisi dalam file dimana pembacaan akan dimulai.
- **seekp(position) :** mencari posisi dalam file dimana rekaman akan dimulai
- **seekg(offset, direction) :** mencari posisi awal pembacaan yang dihitung mulai dari nilai offset hingga nilai direction
- **seekp(offset, direction) :** mencari posisi awal prekaman yang dihitung mulai dari nilai offset hingga nilai direction

- Ada tiga macam nilai direction standard yaitu:
  1. **ios::beg** – offset dihitung dari awal file (BOF)
  2. **ios::cur** – offset dihitung dari posisi pointer saat ini
  3. **ios::end** – offset dihitung dari akhir file (EOF)

# Contoh menghitung file-size

```
// fsize.cpp - obtaining file size
#include <iostream>
#include <fstream>
using namespace std;
int main () {
        long begin,end;
        ifstream myfile ("mytext.txt");
        begin = myfile.tellg();
        myfile.seekg (0, ios::end);
        end = myfile.tellg();
        myfile.close();
        cout << "size is: " << (end-begin) << " bytes.\n";
        system("PAUSE");
        return 0;
}
```



# status file

- beberapa fungsi dapat dipakai untuk memeriksa status file:
- **bad()** – status file buruk tidak bisa dibuka baik sebagai in-stream atau sebagai out-stream
- **fail()** – mirip dengan bad(), tetapi juga apabila dilakukan pembacaan dengan tipe data yang beda (misal huruf dibaca sebagai angka)
- **eof()** – pointer baca sudah sampai ke ujung file (end-of-file)
- **good()** – bernilai true bila kondisi file bagus, dan false bila buruk (bad atau fail)

# Memeriksa kondisi file

- Memeriksa apakah file bisa dibuka

```
ifstream in_file;
in_file.open("mytext.dat");
if (in_file.fail() ) {
    cout << "File ini tidak bisa dibuka.\n";
    exit(1);
}
```

- Apakah file sudah mencapai akhir?

```
if (!in_file.eof() ) {
        // belum mencapai akhir
}
```

# Memeriksa end-of-file

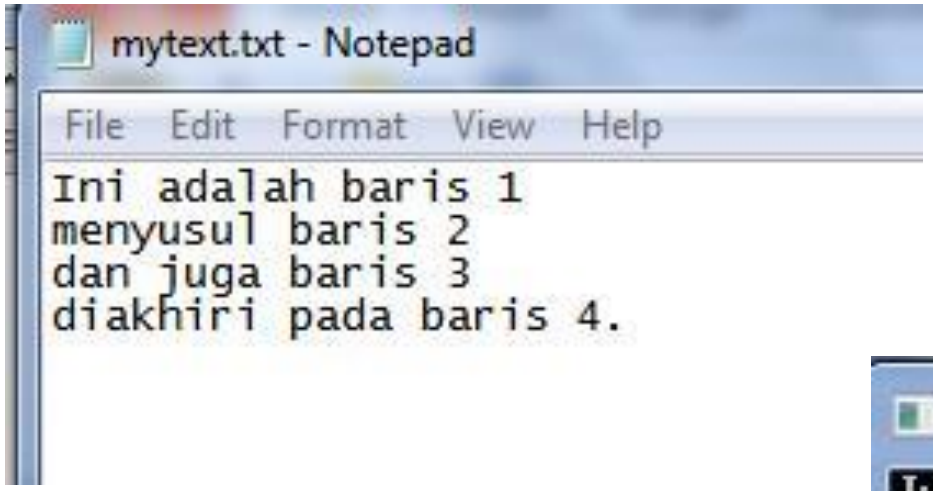
```
char x;
ifstream in_file;
in_file.open("mytext.dat");
if (in_file.fail() ) {
        cout << "File tidak bisa dibuka\n";
        exit(1);
}
in_file >> x;
if (!in_file.eof() ) {
        cout << x;
        infile >> x;
}
else  {
        cout << "File sudah selesai dibaca\n";
        in_file.close();
}
```

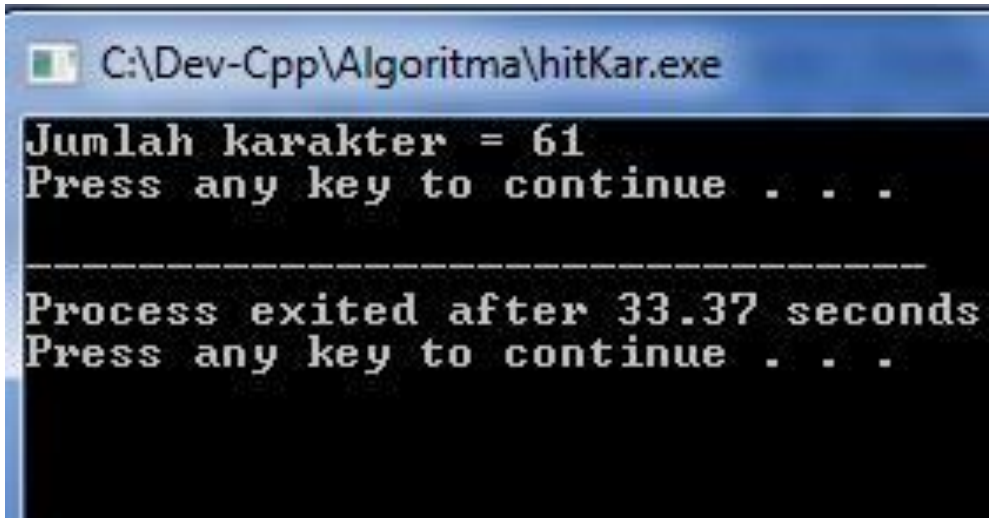
# Contoh: hitung karakter

```
//hitKar.cpp
//mneghitung jumlah karakter
#include <cstdio>
#include <cstdlib>
#include <iostream>
#include <fstream>
using namespace std;

int main() {
    ifstream F;
    char k;
    int jkar;

    F.open("mytext.txt");
    //ketika file dibuka maka posisi-
nya BOF
    jkar = 0;
    F >> k;
    while (!F.eof()) {
        jkar += 1;
        F >> k;
    }
    cout << "Jumlah karakter =
" << jkar << endl;
    system("PAUSE");
    return 0;
}
```


mytext.txt - Notepad
File Edit Format View Help
Ini adalah baris 1
menyusul baris 2
dan juga baris 3
diakhiri pada baris 4.

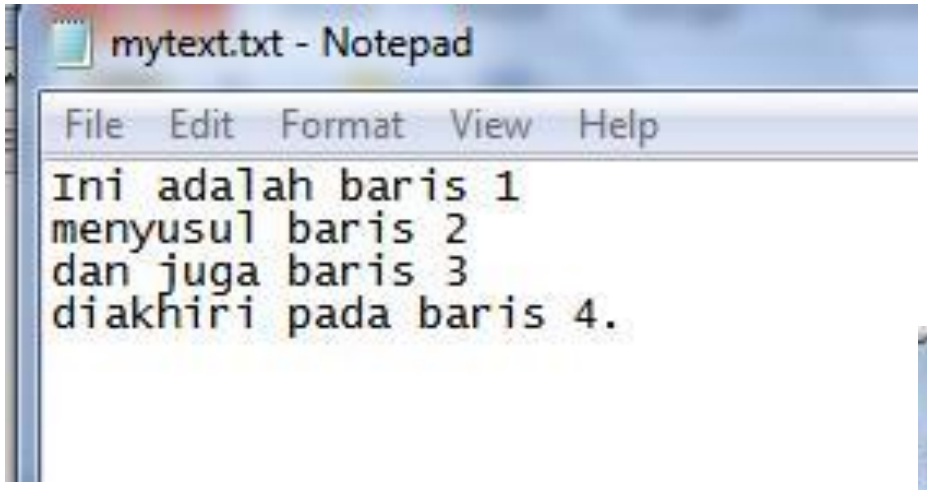

C:\Dev-Cpp\Algoritma\hitKar.exe
Jumlah karakter = 61
Press any key to continue . . .
Process exited after 33.37 seconds
Press any key to continue . . .

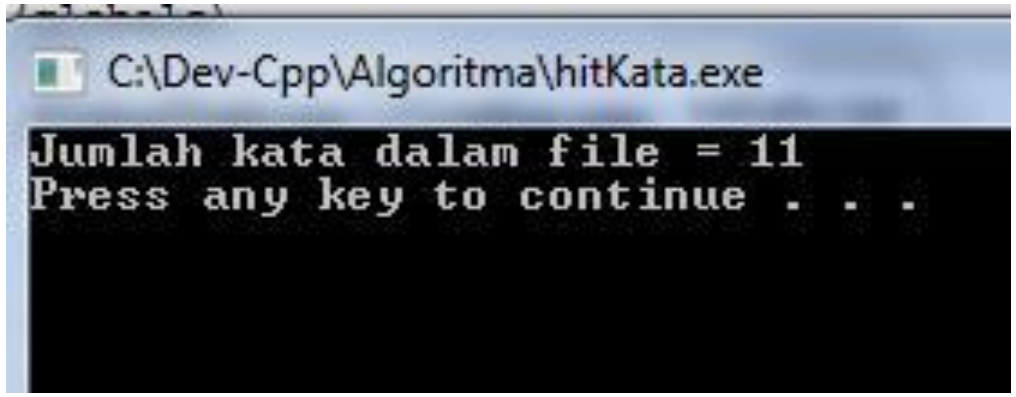
# Contoh: hitung kata

```
//hitKata.cpp
//mneghitung jumlah kata
#include <cstdio>
#include <cstdlib>
#include <iostream>
#include <fstream>
using namespace std;

int main()  {
     ifstream F;
     char k1,k2;
     int jkata;

      F.open("mytext.txt");
//ketika file dibuka maka posisi-nya BOF
     F >> k1; F >> k2;
     jkata = 0;
      while (!F.eof()) {
        //cout << k1 << k2 << endl;
           if ((k1 != ' ') && (k2==' '))
                jkata += 1;
           k1 = k2;
           F.get(k2);
       }
     if ((k1 != ' ') && (k2==' '))
           jkata += 1;
     cout << "Jumlah kata dalam file = " << jkata << endl;
     system("PAUSE");
     return 0;
   }
```


mytext.txt - Notepad
File Edit Format View Help
Ini adalah baris 1
menyusul baris 2
dan juga baris 3
diakhiri pada baris 4.


C:\Dev-Cpp\Algoritma\hitKata.exe
Jumlah kata dalam file = 11
Press any key to continue . . .

# Pemrosesan String

- String adalah untaian beberapa karakter
- Cara mendefinisikan dan inisialisasi:

```
char Nama[21];  // nama maximum 20 huruf
                // akhir string di-isi '\0'
char Nama[21] = "Abdul Hakim Nasution";
```

- Assignment string tidak diperbolehkan:

```
Nama = "Abdul Hakim Nasution";  // Salah
```

- Pemberian nilai diluar inisialisasi harus menggunakan fungsi: *strcpy(target, source);*

```
strcpy(Nama, "Abdul Hakim Nasution");
```

- Fungsi-fungsi string ada dalam library cstring:

```
#include <cstring>
```

| Function | Description | Cautions |
|---|---|---|
| `strcpy(`*Target_String_Var*`,` *Src_String*`)` | Copies the C-string value *Src_String* into the C-string variable *Target_String_Var*. | Does not check to make sure *Target_String_Var* is large enough to hold the value *Src_String*. |
| `strncpy(`*Target_String_Var*`,` *Src_String*`,` *Limit*`)` | The same as the two-argument `strcpy` except that at most *Limit* characters are copied. | If *Limit* is chosen carefully, this is safer than the two-argument version of `strcpy`. Not implemented in all versions of C++. |
| `strcat(`*Target_String_Var*`,` *Src_String*`)` | Concatenates the C-string value *Src_String* onto the end of the C string in the C-string variable *Target_String_Var*. | Does not check to see that *Target_String_Var* is large enough to hold the result of the concatenation. |
| `strncat(`*Target_String_Var*`,` *Src_String*`,` *Limit*`)` | The same as the two-argument `strcat` except that at most *Limit* characters are appended. | If *Limit* is chosen carefully, this is safer than the two-argument version of `strcat`. Not implemented in all versions of C++. |

| | | |
|---|---|---|
| `strlen(`*`Src_String`*`)` | Returns an integer equal to the length of *Src_String*. (The null character, `'\0'`, is not counted in the length.) | |
| `strcmp(`*`String_1, String_2`*`)` | Returns `0` if *String_1* and *String_2* are the same. Returns a value < 0 if *String_1* is less than *String_2*. Returns a value > 0 if *String_1* is greater than *String_2* (that is, returns a nonzero value if *String_1* and *String_2* are different). The order is lexicographic. | If *String_1* equals *String_2*, this function returns `0`, which converts to *false*. Note that this is the reverse of what you might expect it to return when the strings are equal. |
| `strncmp(`*`String_1, String_2, Limit`*`)` | The same as the two-argument `strcat` except that at most *Limit* characters are compared. | If *Limit* is chosen carefully, this is safer than the two-argument version of `strcmp`. Not implemented in all versions of C++. |

- Membandingkan dua string tidak boleh memakai tanda == atau != , tetapi memakai fungsi *strcmp(string1, string2).*

```
if (string1 == string2)   //  salah
   cout << "Keduanya sama";
else
   cout << "Keduanya berbeda";

if  strcmp(string1, string2)  // benar
   cout << "Keduanya berbeda";
else
   cout << Keduanya sama";
```

- Bila string1 sama dengan string2 maka hasilnya 0 (false) bila string1 > string2 maka hasilnya 1 (true), dan bila string1 < string2 maka hasilnya -1.

# input string

Contoh:

```
char Nama[22];
cout << "Masukkan nama anda : ";
cin >> Nama;
cout << Nama;
```

Masukkan nama anda : Abdul Hakim Nasution

*hanya menghasilkan :* Abdul

```
char Nama[22];
cout << "Masukkan nama anda : ";
cin.getline(Nama,21);
cout << Nama;
```

Masukkan Nama anda : Abdul Hakim Nasution

*menghasilkan:* Abdul Hakim Nasution.

```
//inputNama.cpp
#include <iostream>
#include <cstring>
using namespace std;

int main() {
		char Nama[22];
		char x[80];

		cout << "Masukkan Nama : ";
		cin >> Nama;
		cout << "Menghasilkan Nama = " << Nama << endl;

		cin.getline(x,80);
		cout << endl << endl;

		cout << "Masukkan Nama : ";
		cin.getline(Nama,21);
		cout << "Menghasilkan Nama = " << Nama << endl;

		system("PAUSE");
		return 0;
}
```