



File Eksternal (dalam Bahasa C++)

Tim Penyusun Materi
KU1072/Pengenalan Teknologi Informasi B
Kurikulum 2013



KU1072/Pengenalan Teknologi Informasi B
Tahap Tahun Pertama Bersama
Institut Teknologi Bandung





Tujuan

- Mahasiswa memahami kegunaan file sebagai sarana penyimpanan data eksternal
- Mahasiswa memahami primitif-primitif dasar dalam pemrosesan file dalam Bahasa C++
- Mahasiswa memahami skema-skema dasar pemrosesan sekuensial untuk pembacaan dan penulisan file



Pendahuluan

- Stream Input/Output:
 - Flow character antara program dengan I/O device :
 - Contoh input device: keyboard
 - Contoh output device: monitor/layar
 - Input stream: flow dari input device ke program
 - Output stream: flow dari output device ke program
- Stream I/O bersifat sementara: begitu program mati maka data/nilai akan hilang
- Variable dalam program dapat menyimpan data/nilai, tapi bersifat sementara → begitu program mati maka nilai yang tersimpan akan hilang

Pendahuluan

```
//Program TulisNama
//Membaca nama dari keyboard dan menuliskan ke layar
#include <iostream>
using namespace std;

int main () {
    //KAMUS
    string nama;

    //ALGORITMA
    cout << "Tuliskan namamu: " << endl;

    cin >> nama ;

    cout << "Namamu adalah : " << nama << endl;

    return 0;
}
```

Contoh **variable**

Contoh **Stream
Input**

Contoh **Stream
Output**



Pendahuluan

- Pada banyak kasus dibutuhkan agar nilai input/output atau nilai variable disimpan sehingga masih dapat dipakai walaupun program selesai
 - Untuk itu digunakan **file [eksternal]**
- File:
 - Bentuk penyimpanan **eksternal** dalam suatu media penyimpanan, misalnya ***harddisk***
 - bentuk penyimpanan sementara (untuk data variable dan stream I/O) adalah **memory**



File Teks dan File Biner

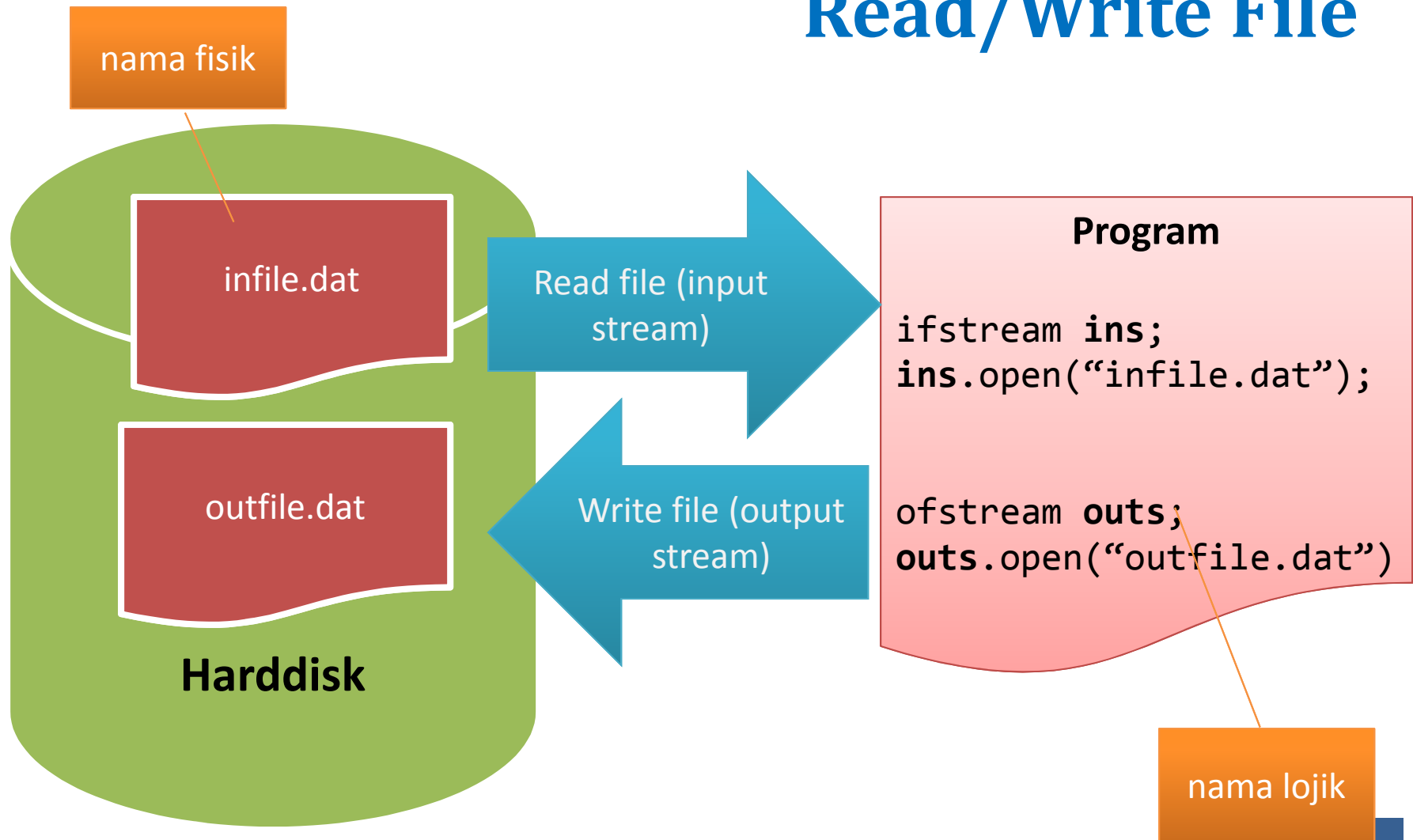
- File Teks:
 - File yang isinya bisa dibaca dan dibuat langsung dengan menggunakan editor teks biasa
 - Contoh editor teks biasa: notepad, wordpad
- File biner (*binary file*):
 - File yang memiliki format khusus yang hanya bisa dibaca dengan program khusus
 - Contoh: Coba buka file *.pdf dengan editor teks biasa. Apa yang terlihat?
- Yang akan digunakan pada kuliah ini hanya **file teks**



Nama Fisik vs Nama Logik

- Dalam program setiap file mempunyai 2 nama:
 - **Nama fisik** : nama file dalam media penyimpanan
Contoh: myfile.txt, mydata.dat
 - **Nama Logik** : nama variabel yang digunakan untuk menggantikan nama fisik file dalam program
 - **ifstream** : type variable untuk membaca input stream dari file
 - **ofstream**: type variable untuk menuliskan output stream ke file

Read/Write File



Contoh

```
#include <fstream>
using namespace std;
```

```
int main () {
```

```
    //KAMUS
```

```
    ifstream in_stream;
```

```
    ofstream out_stream;
```

```
    //ALGORITMA
```

```
    in_stream.open("infile.dat");
```

```
    out_stream.open("outfile.dat");
```

```
    ...
```

```
    in_stream.close();
```

```
    out_stream.close();
```

```
}
```

Contoh Variable untuk membaca input stream dari file

Contoh Variable untuk menuliskan output stream ke file

Membuka file untuk membaca input stream dari file **infile.dat**

Membuka file untuk menuliskan output stream ke file **outfile.dat**

Menutup file



Pemrosesan File

- Membuka file
 - Membuka file untuk membaca isinya (read only)
 - Membuka file untuk menulis isinya (rewrite)
- Membaca isi file
- Menulis isi file
- Menutup file
- End of File (EOF)

Membuka File untuk Membaca Isinya

- Mempersiapkan file untuk dibaca (*read-only*)
- Input stream dari file ke program

```
// KAMUS
string FILE_NAME = "infile.txt";
ifstream fin;

// ALGORITMA
fin.open(FILE_NAME); //buka file dengan modus read-only
//sama dengan fin.open("infile.txt")
...

```

Cara pemanggilan fungsi open adalah cara memanggil fungsi untuk program **berorientasi objek** (Untuk saat ini, hafalkan!)



Membuka File untuk Menulis Isi File

- Output stream dari program ke file
- Mempersiapkan file untuk siap ditulis (*rewrite*)
 - Jika file fisik belum ada, file di-*create*
 - Jika file tidak kosong, maka isi yang lama dihapus dan akan ditimpa dengan isi yang baru

```
// KAMUS
string FILE_NAME = "outfile.txt";
ofstream fout;

// ALGORITMA
fout.open(FILE_NAME); //buka file dengan modus rewrite
...
```

Membaca Isi File

- Membaca data dalam file dan menampung isinya ke suatu variable
 - Hati-hati dengan deklarasi variable → type harus sesuai dengan isi file

```
// KAMUS
string FILE_NAME = "infile.txt";
ifstream fin;
string s1;
int i1;

// ALGORITMA
fin.open(FILE_NAME); //buka file dengan modus read-only
fin >> s1;
fin >> i1;
// bisa disingkat : fin >> s1 >> i1;
...
```

infile.txt

Hello
123

Menulis File

- Menulis nilai-nilai ke dalam file

```
// KAMUS
string FILE_NAME = "outfile.txt";
ofstream fout;

// ALGORITMA
fout.open(FILE_NAME); //buka file dengan modus rewrite
fout << "Hello" << endl;
fout << 123;
// bisa disingkat: fout << "Hello" << endl; << 123;
...
```

outfile.txt

```
Hello
123
```

Menutup File

- Menutup file: file tidak dapat dibaca/ditulis lagi
- Jika **membuka** harus **menutup!!**

```
// KAMUS
ifstream in_stream;
ofstream out_stream;

//ALGORITMA
in_stream.open("infile.dat");
out_stream.open("outfile.dat");

...

in_stream.close();
out_stream.close();
```

Biasakan selalu menulis **close** segera setelah menulis **open!!**
Kode lain sisipkan di antaranya

End of File (EOF)

- Sebuah fungsi yang digunakan untuk menyatakan bahwa pembacaan isi file sudah mencapai akhir file
→ lihat kegunaannya pada pembahasan berikutnya

```
// KAMUS
ifstream in_stream;

//ALGORITMA
in_stream.open("infile.dat");

if (in_stream.eof()) {
    cout << "File kosong" << endl;
} else ...

in_stream.close();
```

`in_stream.eof()` berarti berada di akhir file (artinya sudah tidak ada yang bisa dibaca dari file)



PEMROSESAN FILE SECARA SEKUENSIAL





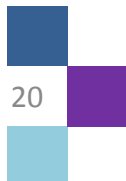
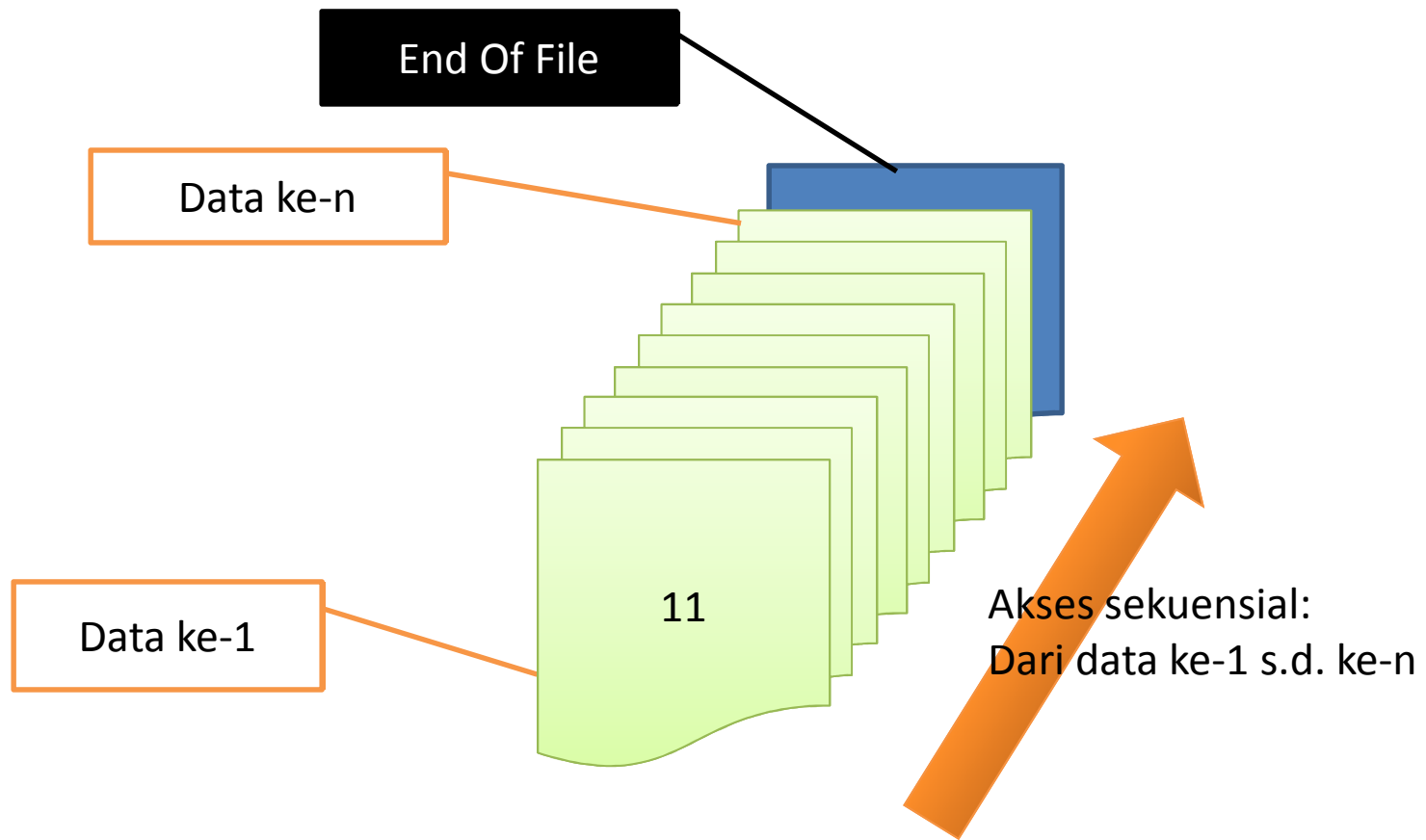
File Sekuensial

- File yang dibaca secara sekuensial dari awal sampai akhir:
 - Tidak ada akses di tengah file
 - Akses hanya bisa maju, tidak bisa mundur, atau lompat
- Untuk itu file harus diproses juga secara sekuensial
- Data yang tersimpan dalam file memiliki type yang sama:
 - *file text, file of integer, file of float, dll.*



Membaca data secara sekuensial hingga akhir file

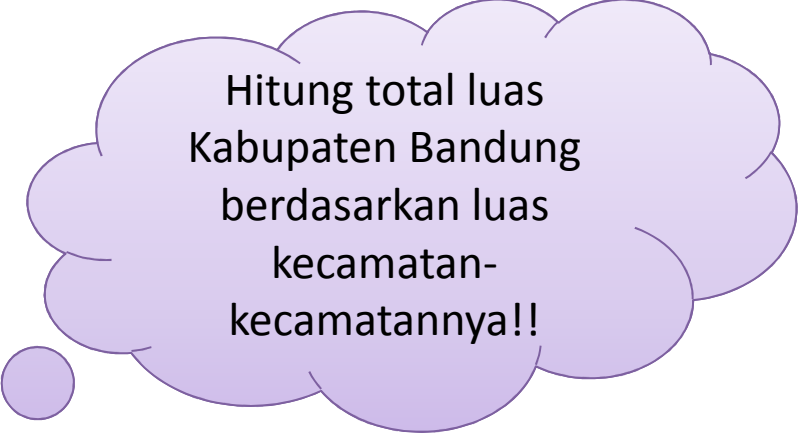
- Pada banyak kasus, program diharapkan membaca data secara sekuensial sampai akhir file, contoh:
 - File berisi nilai mahasiswa satu kelas (skala 0 s.d. 100). harus dihitung rata-rata nilai mahasiswa
 - File berisi luas wilayah setiap kecamatan suatu kabupaten. harus dihitung total wilayah kecamatan untuk mendapatkan luas kabupaten
 - File berisi data tinggi badan pasien. harus dicari pasien yang tertinggi
- Banyaknya data yang tersimpan di file tidak bisa diketahui:
 - File juga mungkin kosong!!



Contoh: Isi file datakecamatan.dat

```
4846.92
14837.01
23957.65
5500.03
19540.93
15207.37
9193.97
5456.52
5102.91
4013.63
3599.23
4930.30
4524.83
2536.46
2400.66
4617.57
4155.54
6497.79
4291.79
2461.06
1462.32
...
```

```
...
1572.46
2550.68
4730.26
1834.50
1054.33
1102.91
2781.23
3157.51
3011.95
5308.34
```



Hitung total luas
Kabupaten Bandung
berdasarkan luas
kecamatan-
kecamatanannya!!

Data luas kecamatan di Kab. Bandung (2009) dlm. Hektar
(diakses dari http://bapeda.bandungkab.go.id/index2.php?option=com_docman&task=doc_view&gid=79&Itemid=37 pada 29 Mei 2013)

```
#include <iostream>
#include <fstream>
using namespace std;

int main () {
// KAMUS
    ifstream in_stream;
    float luas; // variable utk luas kec yg sdg dibaca
    float luaskab; // variable utk luas kabupaten total
// ALGORITMA
    in_stream.open("datakecamatan.dat");

    luaskab = 0; //inisialisasi
    while (!in_stream.eof()) {
        in_stream >> luas;
        luaskab = luaskab + luas;
    }
    cout << "Luas wilayah kabupaten Bandung tahun 2009 adalah
" << luaskab << " hektar";

    in_stream.close();

    return 0;
}
```

Loop akan berhenti, jika sudah sampai pada akhir file, termasuk jika file kosong



Menulis data ke dalam file

- Contoh:
Menyimpan nilai mahasiswa skala 0 s.d. 100 sampai pengguna mengetik angka -999 (tidak termasuk nilai)

```
#include <iostream>
#include <fstream>
using namespace std;

int main () {
// KAMUS
    ofstream out_stream;
    int nr;

// ALGORITMA
    out_stream.open("datanilai.dat");

    cin >> nr;
    while (nr != -999) {
        out_stream << nr << endl;
        cin >> nr;
    }

    out_stream.close();

    return 0;
}
```

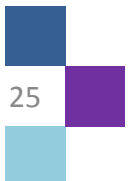
Data dibaca dari keyboard
sampai pengguna
mengetikkan -999

Contoh isi file:

```
100
20
12
54
66
67
78
99
```




CONTOH-CONTOH



Contoh-1

- Apa yang akan tertulis di layar jika kode di samping dieksekusi?
- Isi file **dataku.dat** adalah:

```
1 2 3 4 5
1 2 3
```

```
#include <fstream>
#include <iostream>
using namespace std;

int main () {
//KAMUS
    ifstream ins;
    int sum = 0, num;

//ALGORITMA
    ins.open("dataku.dat");
    while (!ins.eof()) {
        ins >> num;
        sum = sum + num;
    }
    ins.close();
    cout << sum;
    return 0;
}
```



Contoh-2

- File **namakota.txt** berisi sekumpulan nama kota di Indonesia. Lihat contoh di samping.
- Buatlah program untuk membaca file tersebut dan menuliskan ke layar seluruh nama kota yang tertulis dalam file itu.

```
Jakarta  
Bandung  
Semarang  
Jogjakarta  
Medan  
Ambon  
Jayapura  
Palangkaraya  
Manado  
Kendari  
Padang  
Jambi
```



Contoh-3

- Diketahui sebuah file of integer **mymarks.dat** yang berisi daftar nilai mahasiswa di sebuah mata kuliah.
- Buatlah sebuah program yang membaca nilai-nilai tersebut dan menghasilkan nilai rata-rata dari semua mahasiswa
- File mungkin kosong. Jika file kosong, maka keluarkan pesan : “File kosong”.



Contoh-4

- Pak Lurah Ganesha ingin menyimpan data umur penduduk di kelurahannya supaya suatu saat bisa digunakan untuk berbagai kebutuhan
- Bantulah Pak Lurah untuk membuat sebuah program yang menyimpan data masukan umur semua penduduk (umur adalah bilangan bulat) ke dalam suatu file, jika diketahui bahwa jumlah penduduk kelurahan adalah 100 orang.
- Petunjuk: gunakan loop for untuk mengendalikan entri data umur



Contoh-5b

- Diketahui sebuah file **IPK.dat** yang berisi daftar IPK mahasiswa sebuah fakultas/sekolah.
- Buatlah program yang membaca isi file IPK.dat dan kemudian memindahkan isinya ke file **IPK_3.dat** hanya IPK mahasiswa yang bernilai ≥ 3.00 .
- File mungkin kosong. Jika file kosong, maka keluarkan pesan : “File kosong”.



Contoh-5b

- Bagaimana jika diinginkan data IPK mahasiswa yang bernilai ≥ 3.00 tidak disimpan ke file, melainkan ke sebuah array?
- Asumsikan data IPK yang akan disimpan tidak lebih banyak dari 100 buah
- Dengan demikian, selanjutnya akan dilakukan pemrosesan data pada array → tidak dibahas di sini

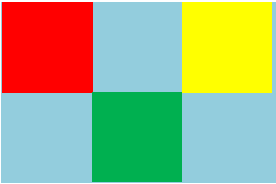
Contoh-2 : Solusi

Contoh-1 : Solusi

- Yang tertulis di layar adalah: **21**

```
//Program namakota
//Menuliskan nama kota dari file
//namakota.txt dan menampilkannya
//ke layar
#include <iostream>
#include <fstream>
using namespace std;

int main () {
//KAMUS
    ifstream ins;
    string kota;
//ALGORITMA
    ins.open("namakota.txt");
    while (!ins.eof()) {
        ins >> kota;
        cout << kota << endl;
    }
    ins.close();
    return 0;
}
```

Contoh-3: Solusi

```
// File: nilairataan.cpp
// Program NilaiRataan
// Membaca nilai-nilai integer
// dari file dan menghasilkan
// rata-rata dari nilai-nilai
// tersebut.
```

```
#include <iostream>
#include <fstream>
using namespace std;
```

```
int main () {
    //KAMUS
    ifstream ins;
    int x;        //bilangan
    int sum;     //jml bilangan
    int N;       //banyak bil
    float rata; //rata-rata
```

```
//ALGORITMA
ins.open("mymarks.dat");
//Membaca isi file sekaligus
//menjumlahkan dan menghitung
//banyaknya bilangan
sum = 0; N = 0;
while (!ins.eof()) {
    ins >> x;
    sum = sum + x;
    N++;
}
//Menampilkan ke layar
if (N > 0) {
    cout << "Nilai rata-rata = " <<
(float)sum/(float)N << endl;
} else { //N = 0, file kosong
    cout << "File kosong" << endl;
}

ins.close();
return 0;
}
```

```

// Program UmurPenduduk
// Membaca data umur sebanyak jumlah
// penduduk dan menyimpan ke file
#include <iostream>
#include <fstream>
using namespace std;

int main () {
// KAMUS
    ofstream fout;
    int i,
        N,    //banyak penduduk
        umur; //masukan umur
// ALGORITMA
    fout.open("dataumur.dat");

    N = 100;
    for (i=1;i<=N;i++) {
        cin >> umur;
        out_stream << umur << endl;
    }

    fout.close();
    return 0;
}

```

Contoh-4: Solusi

Contoh-5a: Solusi

```
// Program DataIPKFile
// Membaca data IPK dari
// sebuah file dan
// menyimpan IPK >= 3.00 ke
// file lain
```

```
#include <iostream>
#include <fstream>
using namespace std;
```

```
int main () {
```

```
    //KAMUS
```

```
    ifstream infile;
    ofstream outfile;
    float ipk;
```

```
//ALGORITMA
```

```
infile.open("IPK.dat");
outfile.open("IPK_3.dat");
if(!infile.eof()) {
```

```
    do {
        infile >> ipk;
        if (ipk >= 3.00) {
            fout << ipk << endl;
        }
    } while (!infile.eof());
```

```
    } else {
        cout << "File kosong" <<endl;
    }
```

```
outfile.close();
infile.close();
```

```
return 0;
```

```
}
```

Contoh-5b: Solusi

```
// Program DataIPKArray
// Membaca data IPK dari
// sebuah file dan
// menyimpan IPK >= 3.00 ke
// sebuah array

#include <iostream>
#include <fstream>
using namespace std;

int main () {
    //KAMUS
    ifstream infile;
    float ipk;
    float arrIPK[100];
    //indeks array dari 0-99
    int i, N;
    //N : banyak data IPK
```

```
//ALGORITMA
infile.open("IPK.dat");
if(!infile.eof()) {
    i = 0;
    do {
        infile >> ipk;
        if (ipk >= 3.00) {
            arrIPK[i] = ipk;
            i++;
        }
    } while (!infile.eof());
    N = i+1;
} else {
    cout << "File kosong" <<endl;
    N = 0;
}

infile.close();
return 0;
}
```