

# Binary File (cont)

Inst. Nguyễn Minh Huy

# Contents



- Binary file.
- BMP exercise.

# Contents



- **Binary file.**
- **BMP exercise.**



## ■ Read/write struct:

- Read/write one-byte-one bytes of whole struct to file.

➔ More effective than read/write struct members.

```
struct Fraction {  
    int num;  
    int denom;  
};
```

```
void readFraction(FILE *f, Fraction &p) { // Read 8 bytes from file into p.  
    fread( &p, sizeof(Fraction), 1, f ); // First 4 bytes into p.num.  
} // Second 4 bytes into p.denom.
```

```
void writeFraction(FILE *f, Fraction p) { // Write 8 bytes p to file.  
    fwrite( &p, sizeof(Fraction), 1, f ); // p.num write to first 4 bytes.  
} // p.denom write to second 4 bytes.
```

# Binary file



## ■ #include <stdint.h>:

### ■ What size of int in C?

➔ Depends on platform.

### ■ Binary file read/write needs fix-sized integer.

➔ Use #include <stdint.h>

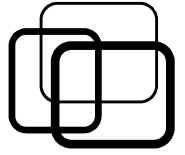
### ■ Fix-sized integer:

- 1 byte: int8\_t, uint8\_t.
- 2 bytes: int16\_t, uint16\_t.
- 4 bytes: int32\_t, uint32\_t.
- 8 bytes: int64\_t, uint64\_t.

```
#include <stdint.h>
```

```
struct Fraction  
{  
    int32_t num;  
    int32_t denom;  
};
```

# Contents



- Binary file.
- **BMP exercise.**



## ■ Practice 5.1 (\*):

Write C/C++ program to cut Bitmap file into equal parts in command-line. Each part is saved in a new Bitmap file.

Command-line syntax:

`<program> <file Bmp> [-h <parts in height>] [-w <parts in width>]`

Example: program cutbmp.exe

- Cut 3 parts in height (save in 3 new Bitmap files):

`cutbmp.exe d:/images/img1.bmp -h 3`

- Cut 2 parts in height, 4 parts in width (save in 8 new Bitmap files)

`cutbmp.exe d:/images/img1.bmp -h 2 -w 4`

