EECS 10: Computational Methods in Electrical and Computer Engineering

Lecture 23

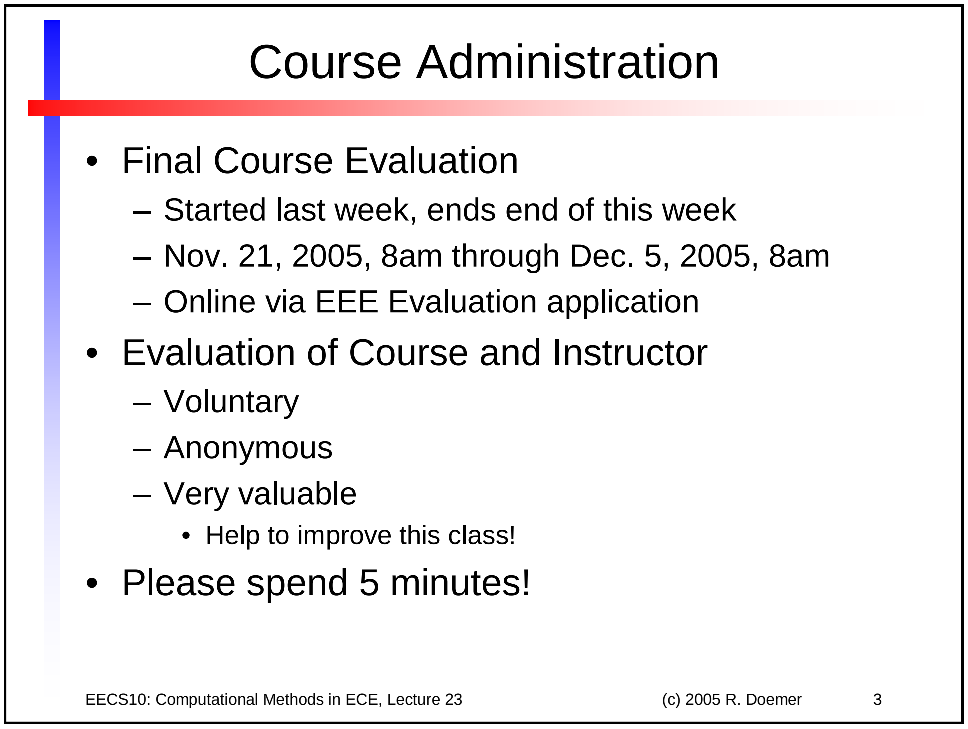
Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Lecture 23: Overview

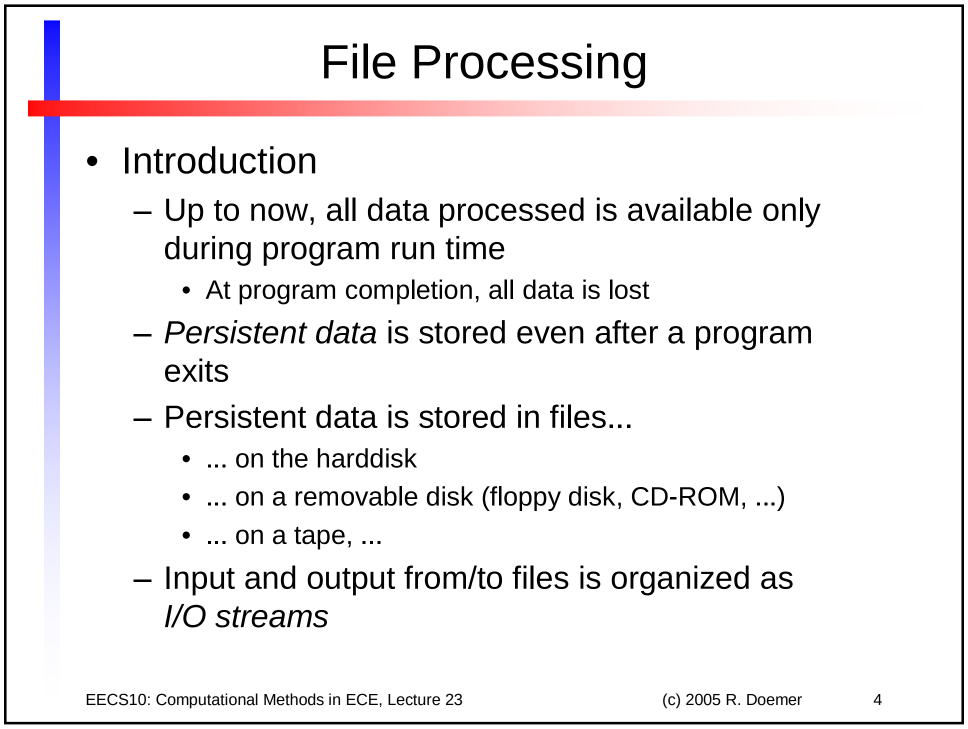
- Course Administration
  - Reminder: Final course evaluation
- File Processing
  - Introduction
  - Standard input and output streams
  - File streams, I/O
  - Standard library functions in **`stdio.h`**
  - Program example **`PhotoLab.c`**



Course Administration

- Final Course Evaluation
  - Started last week, ends end of this week
  - Nov. 21, 2005, 8am through Dec. 5, 2005, 8am
  - Online via EEE Evaluation application
- Evaluation of Course and Instructor
  - Voluntary
  - Anonymous
  - Very valuable
    - Help to improve this class!
- Please spend 5 minutes!

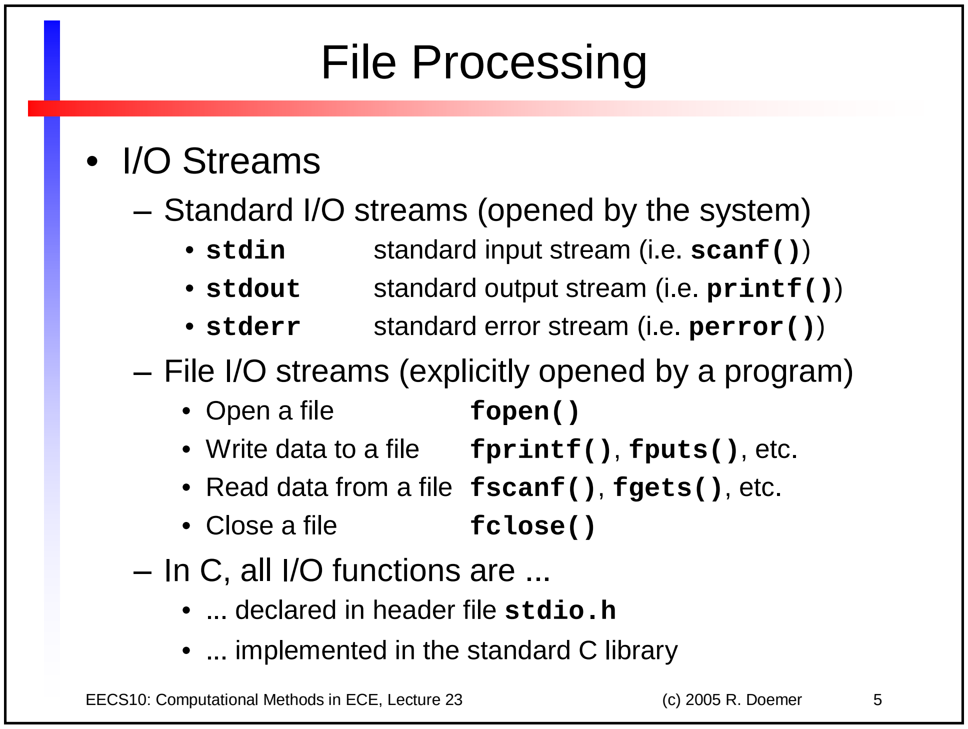
EECS10: Computational Methods in ECE, Lecture 23 (c) 2005 R. Doemer 3



File Processing

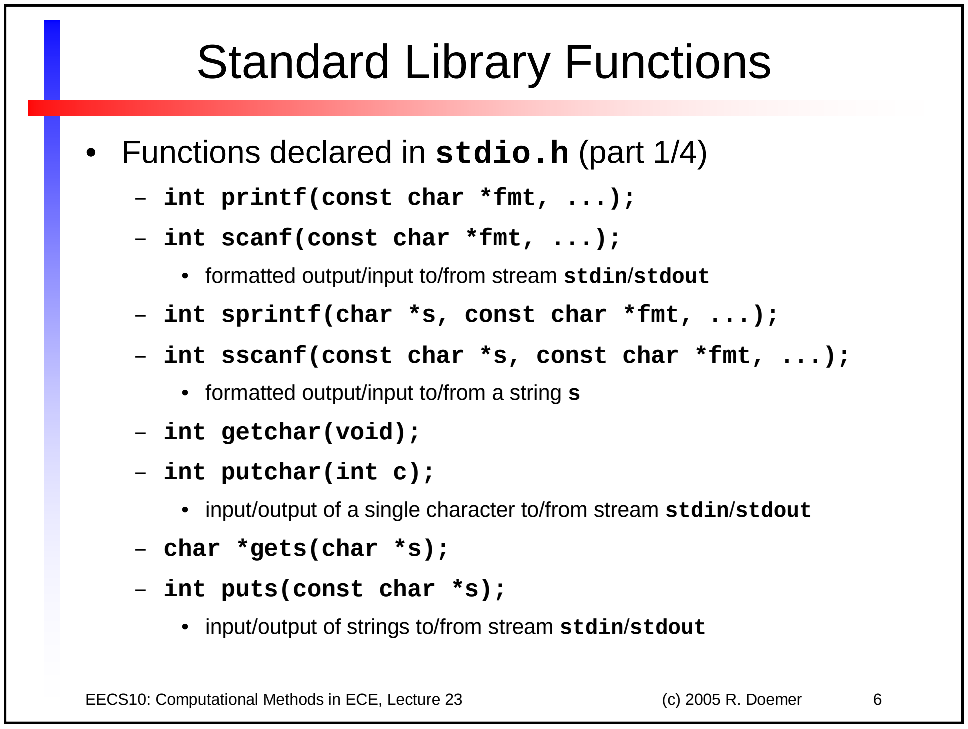
- Introduction
  - Up to now, all data processed is available only during program run time
    - At program completion, all data is lost
  - *Persistent data* is stored even after a program exits
  - Persistent data is stored in files...
    - ... on the harddisk
    - ... on a removable disk (floppy disk, CD-ROM, ...)
    - ... on a tape, ...
  - Input and output from/to files is organized as *I/O streams*

EECS10: Computational Methods in ECE, Lecture 23 (c) 2005 R. Doemer 4



File Processing

- I/O Streams
  - Standard I/O streams (opened by the system)
    - **`stdin`** standard input stream (i.e. **`scanf()`**)
    - **`stdout`** standard output stream (i.e. **`printf()`**)
    - **`stderr`** standard error stream (i.e. **`perror()`**)
  - File I/O streams (explicitly opened by a program)
    - Open a file **`fopen()`**
    - Write data to a file **`fprintf()`**, **`fputs()`**, etc.
    - Read data from a file **`fscanf()`**, **`fgets()`**, etc.
    - Close a file **`fclose()`**
  - In C, all I/O functions are ...
    - ... declared in header file **`stdio.h`**
    - ... implemented in the standard C library



Standard Library Functions

- Functions declared in **`stdio.h`** (part 1/4)
  - **`int printf(const char *fmt, ...);`**
  - **`int scanf(const char *fmt, ...);`**
    - formatted output/input to/from stream **`stdin`**/**`stdout`**
  - **`int sprintf(char *s, const char *fmt, ...);`**
  - **`int sscanf(const char *s, const char *fmt, ...);`**
    - formatted output/input to/from a string **`s`**
  - **`int getchar(void);`**
  - **`int putchar(int c);`**
    - input/output of a single character to/from stream **`stdin`**/**`stdout`**
  - **`char *gets(char *s);`**
  - **`int puts(const char *s);`**
    - input/output of strings to/from stream **`stdin`**/**`stdout`**

Standard Library Functions

- Functions declared in **`stdio.h`** (part 2/4)
  - **`typedef __FILE FILE;`**
    - opaque type for a file handle
  - **`FILE *fopen(const char *n, const char *m);`**
    - open file named **`n`** for input (**`"r"`**), output (**`"w"`**), or append (**`"a"`**)
    - returns a file handle, or **`NULL`** in case of an error
  - **`int fclose(FILE *f);`**
    - closes an open file handle
  - **`int fflush(FILE *f);`**
    - flushes any unwritten data from a buffer into the file
  - **`int fprintf(FILE *f, const char *fmt, ...);`**
  - **`int fscanf(FILE *f, const char *fmt, ...);`**
  - **`int fgetc(FILE *f);`**
  - **`char *fgets(char *s, int n, FILE *f);`**
  - **`int fputc(int c, FILE *f);`**
  - **`int fputs(const char *s, FILE *f);`**
    - input/output functions from/to stream **`f`**

Standard Library Functions

- Functions declared in **`stdio.h`** (part 3/4)
  - **`typedef unsigned int size_t;`**
    - type for size of a piece of memory
  - **`size_t fread(void *p, size_t s, size_t n, FILE *f);`**
    - binary input to memory location **`p`** for **`n`** times **`s`** bytes from file **`f`**
  - **`size_t fwrite(const void *p, size_t s, size_t n, FILE *f);`**
    - binary output from memory location **`p`** for **`n`** times **`s`** bytes to file **`f`**
  - **`int fseek(FILE *f, long pos, int w);`**
    - move to position **`pos`** in file **`f`** (from beginning/current pos/end)
  - **`long ftell(FILE *f);`**
    - return the current position in file **`f`** (from beginning)
  - **`void rewind(FILE *f);`**
    - move to beginning of file **`f`**
  - **`int feof(FILE *f);`**
    - check if end of file **`f`** is reached

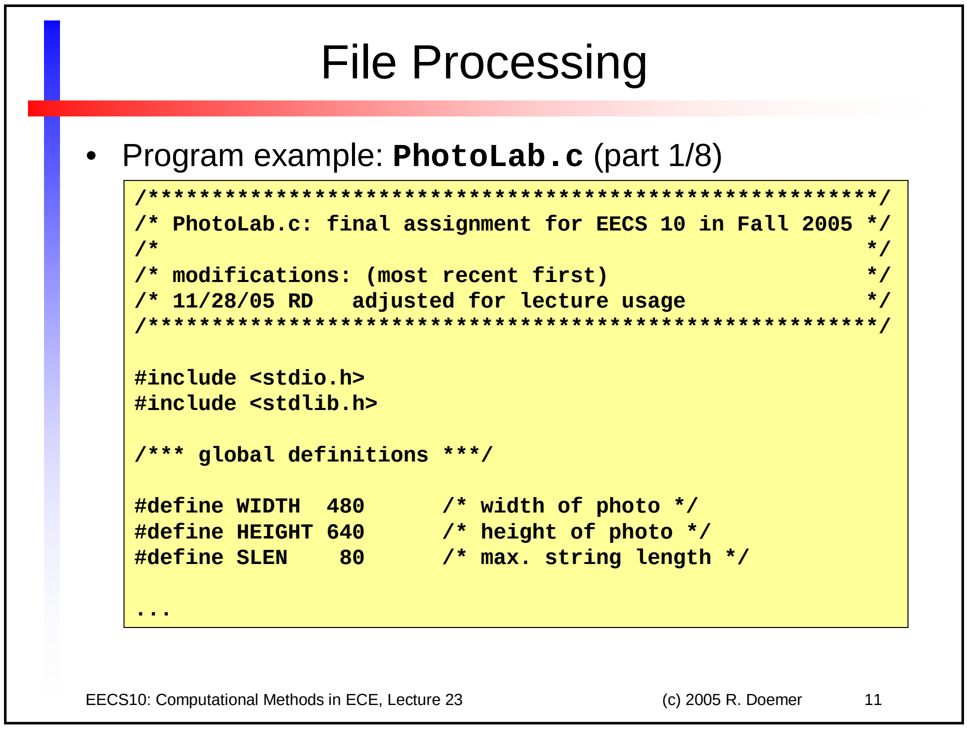
Standard Library Functions

- Functions declared in **`stdio.h`** (part 4/4)
  - **`int ferror(FILE *f);`**
    - returns the current error status for file **`f`**
  - **`void perror(const char *prg);`**
    - print current error for program **`prg`** to stream **`stderr`**
  - **`int remove(const char *filename);`**
    - delete file **`filename`**
  - **`int rename(const char *old, const char *new);`**
    - rename file **`old`** to new name **`new`**

File Processing

- Program example: **`PhotoLab`**
  - Digital image manipulation
    - Read an image from a file
    - Manipulate the image in memory
    - Write the modified image to file
  - Portable Pixel Map (PPM) file format
    - simple uncompressed file format for color images
    - Header section (including picture width, height)
    - Data section (pixel values in Red/Green/Blue format)

```
P6
480 640
255
RGBRGBRGB…
```

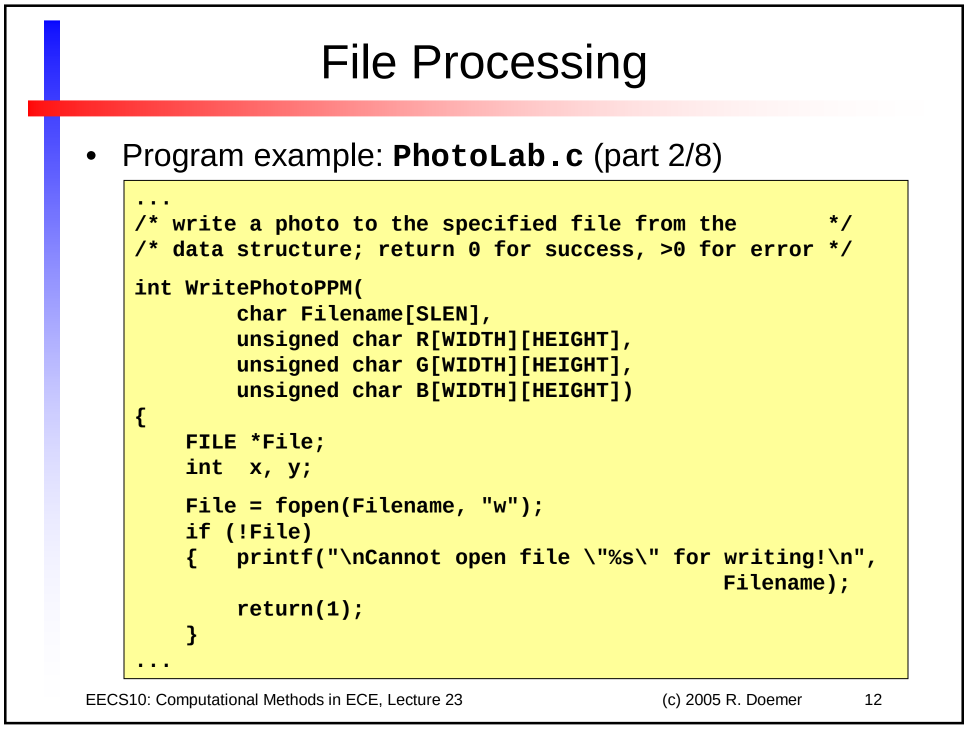


File Processing

- Program example: **`PhotoLab.c`** (part 1/8)

```
/*********************************************************/
/* PhotoLab.c: final assignment for EECS 10 in Fall 2005 */
/*                                                       */
/* modifications: (most recent first)                    */
/* 11/28/05 RD   adjusted for lecture usage              */
/*********************************************************/

#include <stdio.h>
#include <stdlib.h>

/*** global definitions ***/

#define WIDTH  480      /* width of photo */
#define HEIGHT 640      /* height of photo */
#define SLEN    80      /* max. string length */

...
```

EECS10: Computational Methods in ECE, Lecture 23 (c) 2005 R. Doemer 11

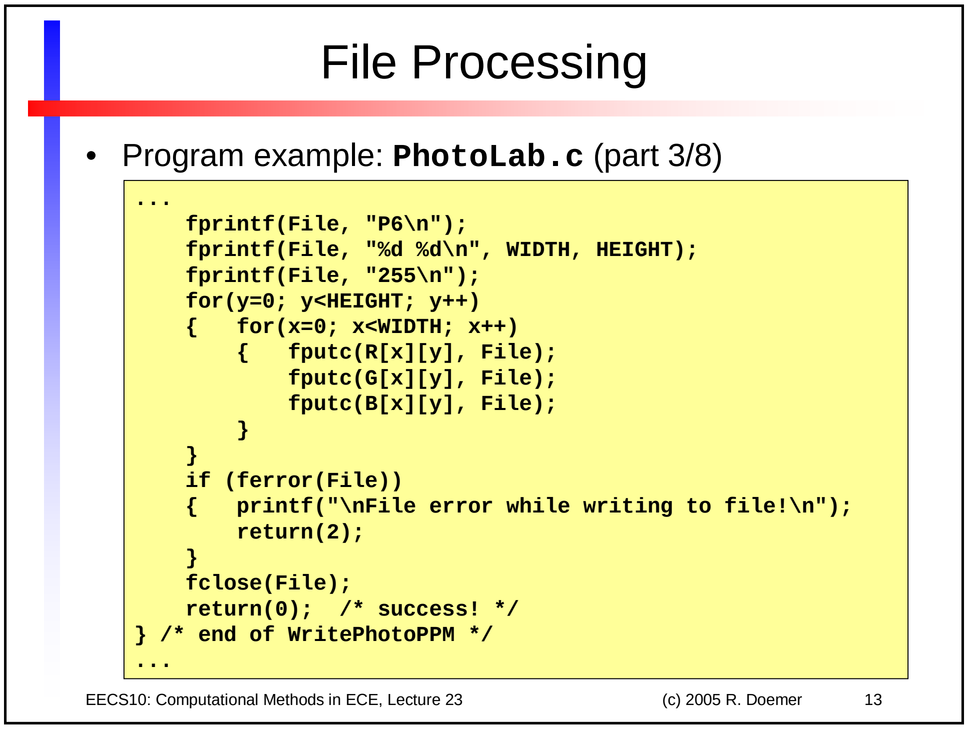


File Processing

- Program example: **`PhotoLab.c`** (part 2/8)

```
...
/* write a photo to the specified file from the       */
/* data structure; return 0 for success, >0 for error */
int WritePhotoPPM(
        char Filename[SLEN],
        unsigned char R[WIDTH][HEIGHT],
        unsigned char G[WIDTH][HEIGHT],
        unsigned char B[WIDTH][HEIGHT])
{
    FILE *File;
    int  x, y;
    File = fopen(Filename, "w");
    if (!File)
    {   printf("\nCannot open file \"%s\" for writing!\n",
                                            Filename);
        return(1);
    }
...
```

EECS10: Computational Methods in ECE, Lecture 23 (c) 2005 R. Doemer 12

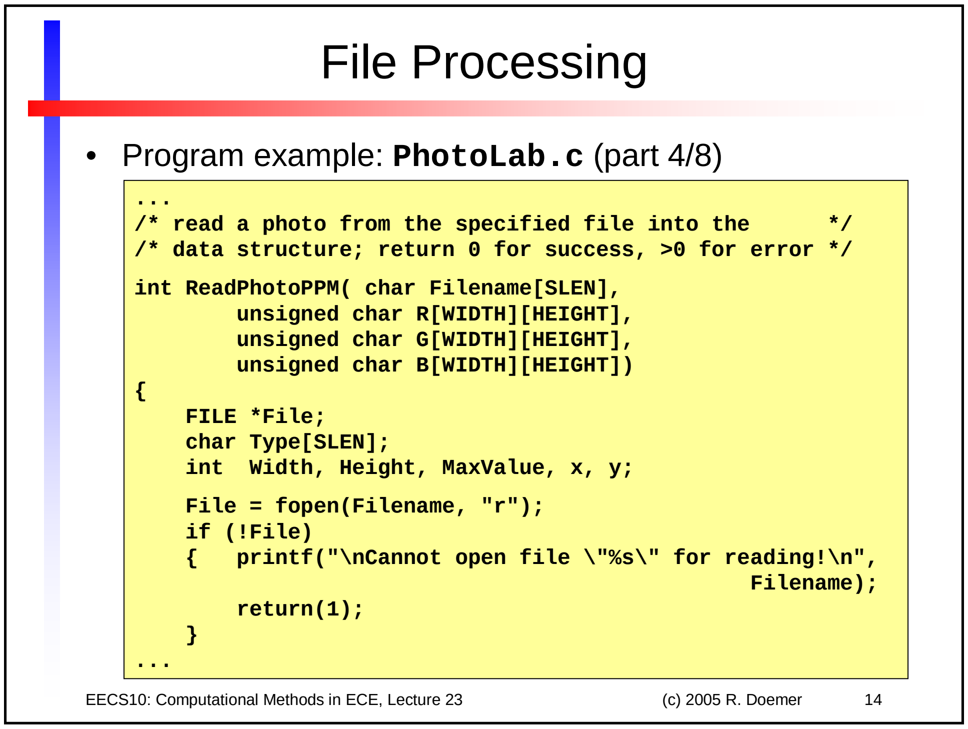


File Processing

- Program example: **`PhotoLab.c`** (part 3/8)

```
...
    fprintf(File, "P6\n");
    fprintf(File, "%d %d\n", WIDTH, HEIGHT);
    fprintf(File, "255\n");
    for(y=0; y<HEIGHT; y++)
    {   for(x=0; x<WIDTH; x++)
        {   fputc(R[x][y], File);
            fputc(G[x][y], File);
            fputc(B[x][y], File);
        }
    }
    if (ferror(File))
    {   printf("\nFile error while writing to file!\n");
        return(2);
    }
    fclose(File);
    return(0);  /* success! */
} /* end of WritePhotoPPM */
...
```

EECS10: Computational Methods in ECE, Lecture 23 (c) 2005 R. Doemer 13



File Processing

- Program example: **`PhotoLab.c`** (part 4/8)

```
...
/* read a photo from the specified file into the      */
/* data structure; return 0 for success, >0 for error */

int ReadPhotoPPM( char Filename[SLEN],
        unsigned char R[WIDTH][HEIGHT],
        unsigned char G[WIDTH][HEIGHT],
        unsigned char B[WIDTH][HEIGHT])
{
    FILE *File;
    char Type[SLEN];
    int  Width, Height, MaxValue, x, y;

    File = fopen(Filename, "r");
    if (!File)
    {   printf("\nCannot open file \"%s\" for reading!\n",
                                                Filename);
        return(1);
    }
...
```

EECS10: Computational Methods in ECE, Lecture 23 (c) 2005 R. Doemer 14

File Processing

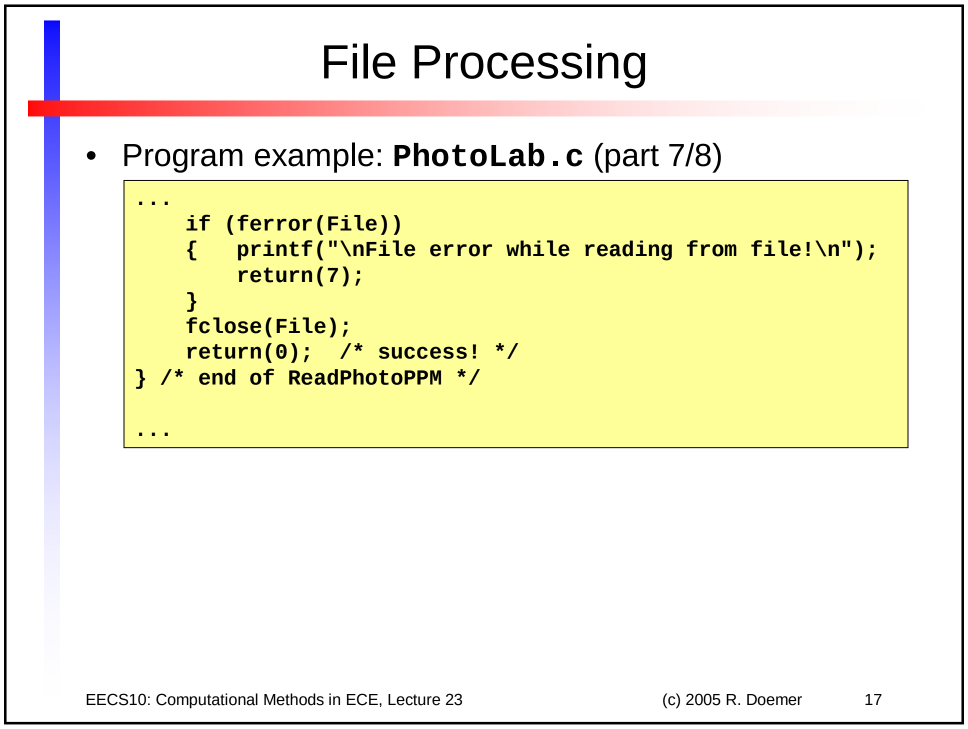
- Program example: **`PhotoLab.c`** (part 5/8)

```
...
    fscanf(File, "%79s", Type);
    if (Type[0] != 'P' || Type[1] != '6' || Type[2] != 0)
    {   printf("\nUnsupported file format!\n");
        return(2);
    }
    fscanf(File, "%d", &Width);
    if (Width != WIDTH)
    {   printf("\nUnsupported image width %d!\n", Width);
        return(3);
    }
    fscanf(File, "%d", &Height);
    if (Height != HEIGHT)
    {   printf("\nUnsupported image height %d!\n", Height);
        return(4);
    }
...
```

File Processing

- Program example: **`PhotoLab.c`** (part 6/8)

```
...
    fscanf(File, "%d", &MaxValue);
    if (MaxValue != 255)
    {   printf("\nUnsupported maximum %d!\n", MaxValue);
        return(5);
    }
    if ('\n' != fgetc(File))
    {   printf("\nCarriage return expected!\n");
        return(6);
    }
    for(y=0; y<HEIGHT; y++)
    {   for(x=0; x<WIDTH; x++)
        {   R[x][y] = fgetc(File);
            G[x][y] = fgetc(File);
            B[x][y] = fgetc(File);
        }
    }
...
```

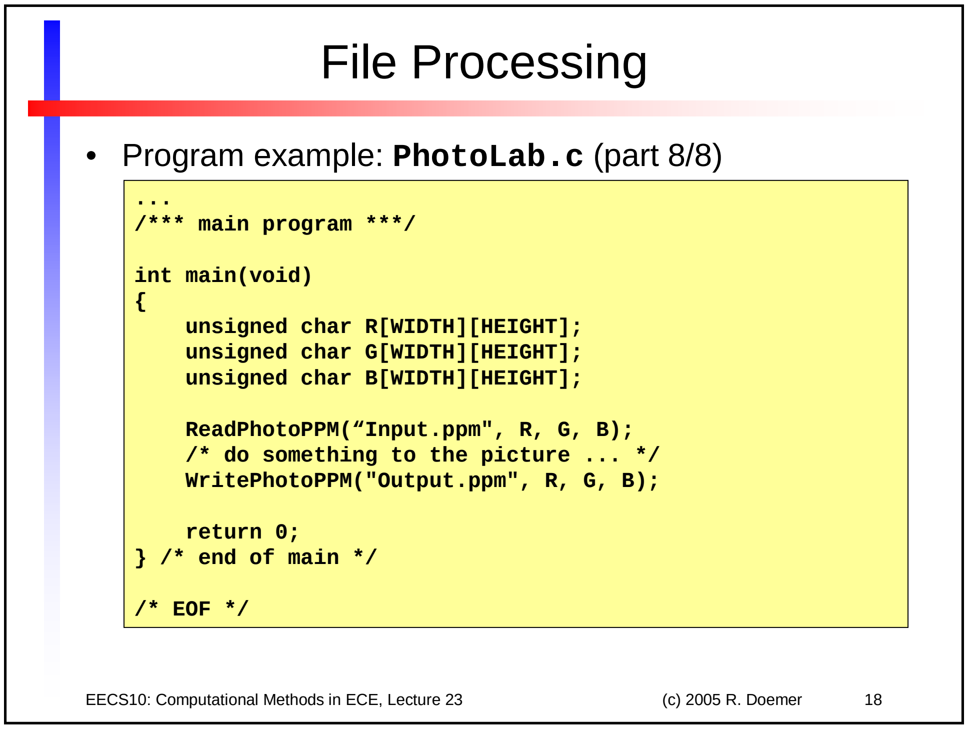


File Processing

- Program example: **`PhotoLab.c`** (part 7/8)

```
...
    if (ferror(File))
    {   printf("\nFile error while reading from file!\n");
        return(7);
    }
    fclose(File);
    return(0);  /* success! */
} /* end of ReadPhotoPPM */

...
```

EECS10: Computational Methods in ECE, Lecture 23 (c) 2005 R. Doemer 17

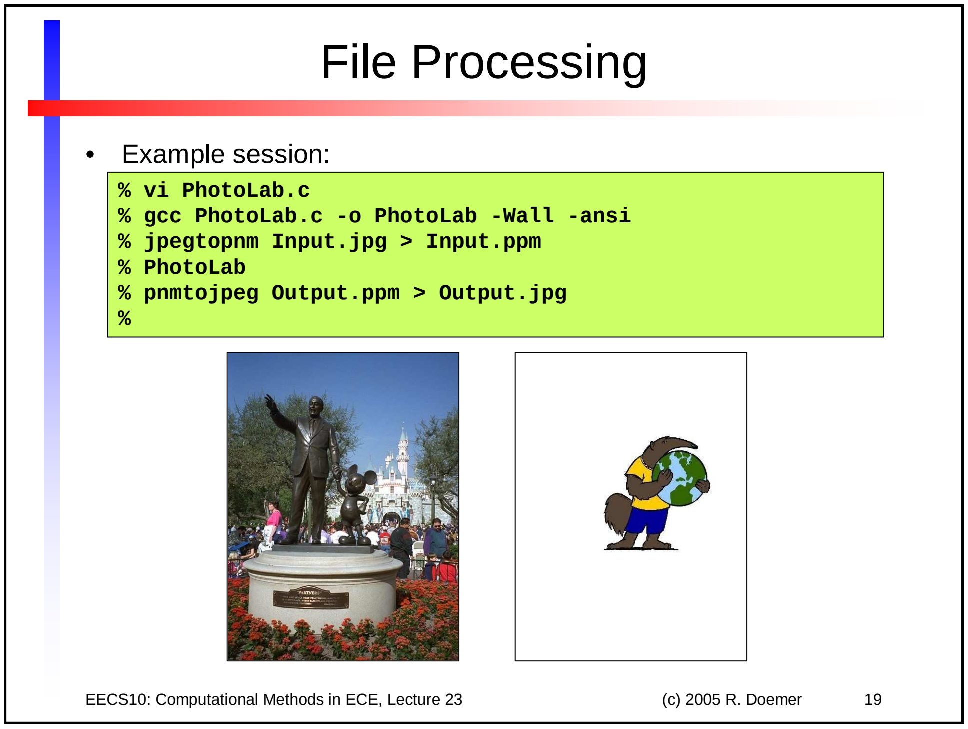


File Processing

- Program example: **`PhotoLab.c`** (part 8/8)

```
...
/*** main program ***/

int main(void)
{
    unsigned char R[WIDTH][HEIGHT];
    unsigned char G[WIDTH][HEIGHT];
    unsigned char B[WIDTH][HEIGHT];

    ReadPhotoPPM(“Input.ppm", R, G, B);
    /* do something to the picture ... */
    WritePhotoPPM("Output.ppm", R, G, B);

    return 0;
} /* end of main */

/* EOF */
```

EECS10: Computational Methods in ECE, Lecture 23 (c) 2005 R. Doemer 18



File Processing

- Example session:

```
% vi PhotoLab.c
% gcc PhotoLab.c -o PhotoLab -Wall -ansi
% jpegtopnm Input.jpg > Input.ppm
% PhotoLab
% pnmtojpeg Output.ppm > Output.jpg
%
```

EECS10: Computational Methods in ECE, Lecture 23 (c) 2005 R. Doemer 19