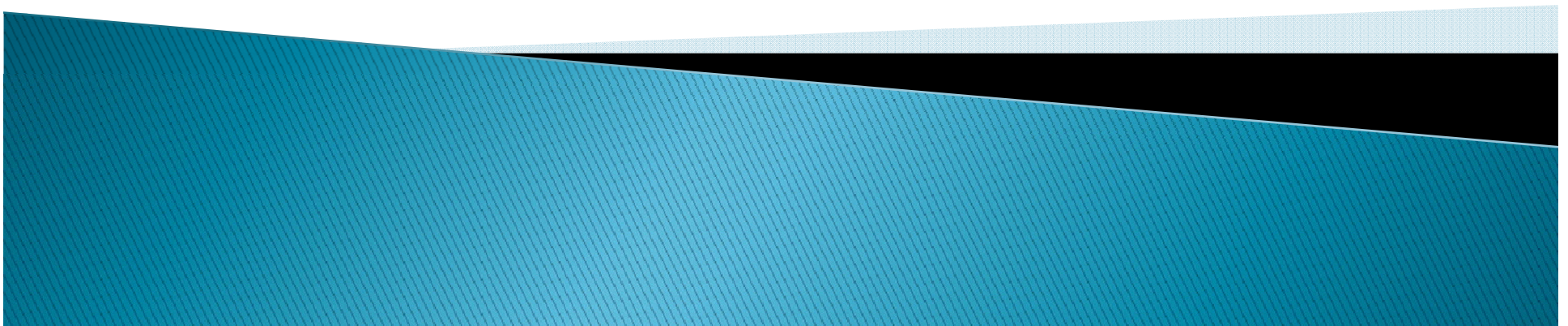


# C programming language

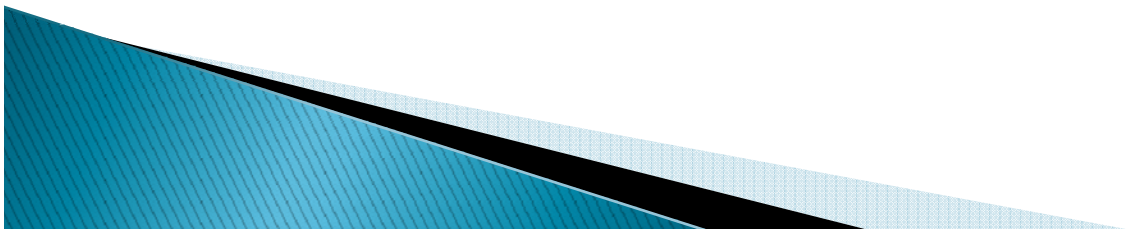
CS 210 Tutorial 11

File input and output/Encryption-Decryption



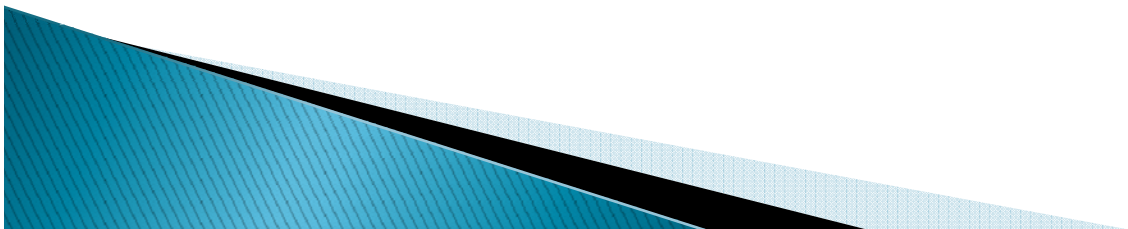
# File I/O in C

- ▶ Other than reading and writing to screen, we can also
- ▶ read and write files
- ▶ What you need:
  - ▶ • `stdio.h`
- ▶ What you do:
  - ▶ • `#include <stdio.h>`
  - ▶ • Declare a `File *` variable (called as a file handler) for the file reference.



# File I/O in C

- ▶ What you do next:
- ▶ Define the File \* by calling fopen, with the correct mode.
- ▶ Check if the file is NULL. If so, quit, with some error message.
- ▶ Depending on what you want:
  - fprintf(filepointer, "Print this into the file.\n");
  - fgets(s, n, filepointer);
- ▶ ALWAYS call fclose(filepointer); at the end.



# Open a text file and read line by line: demonstration2

```
▶ #include <stdio.h>

▶ int main ( void )
▶ {
▶     //static const
▶     char filename[] = "textfile.txt";
▶     FILE *file = fopen ( filename, "r" );
▶     if ( file != NULL )
▶     {
▶         char line [ 128 ]; /* or other suitable maximum line size */

▶         while ( fgets ( line, sizeof line, file ) != NULL ) /* read a line */
▶         {
▶             printf(line);
▶         }
▶         fclose ( file );
▶     }
▶     else
▶     {
▶         perror ( filename ); /* why didn't the file open? */
▶     }
▶     return 0;
▶ }
```

# Read file and make a new file with the same content

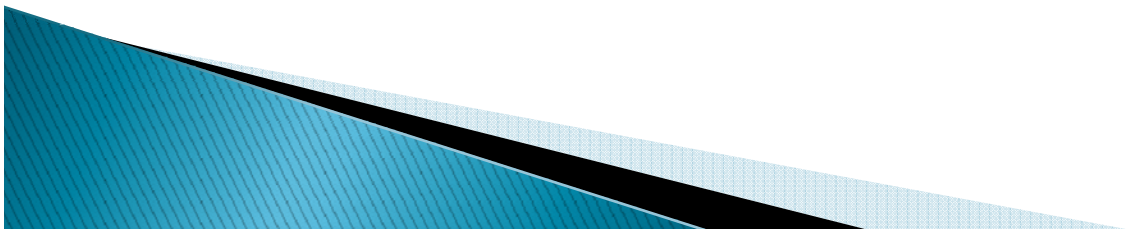
## ▶ Demonstration3:

```
▶ #include <stdio.h>
▶ /* Library function prototypes
▶ FILE * fopen(const char * filename, const
▶ char *
▶ mode);
▶ int fclose(FILE * stream); */
▶ /* Local function prototypes */
▶ int main(int argc, char * argv[])
▶ {
▶ char * inputname;
▶ char * outputname;
▶ FILE * inputfile;
▶ FILE * outputfile;
▶ int charread = 0;
▶ if (argc != 3)
▶ {
▶ printf("Usage: textcopy inputfile
▶ outputfile.\n");
▶ return 1;
▶ }
▶ inputname = argv[1];
▶ outputname = argv[2];
```

```
■ /* Open/ create files. */
■ inputfile = fopen(inputname,"r"); /* mode "Read"
■ */
■ outputfile = fopen(outputname,"w"); /* mode
■ "Write" */
■ if (inputfile == NULL || outputfile == NULL) /* files
■ did not open */
■ {
■ printf("Files could not be opened\n");
■ return 1; /* quit now */
■ }
■ while ((charread = fgetc(inputfile)) != EOF)
■ {
■ printf("%c", charread);
■ fprintf(outputfile, "%c", charread);
■ }
■ /* close the file */
■ if (fclose(inputfile) != 0 || fclose(outputfile) != 0)
■ {
■ printf("Close file error.");
■ }
■ return 0;
■ }
```

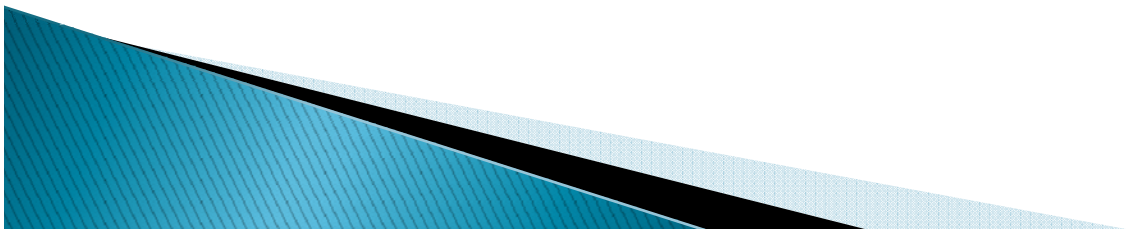
# Copy files

- ▶ Open copyfile.sln
- ▶ Modify with some introduction on encryption.
- ▶ Demonstration shown in class.
- ▶ Double way encryption/decryption using a public key.
- ▶ Using Exclusive OR with one byte
  - no of keys =  $2^8 = 256$



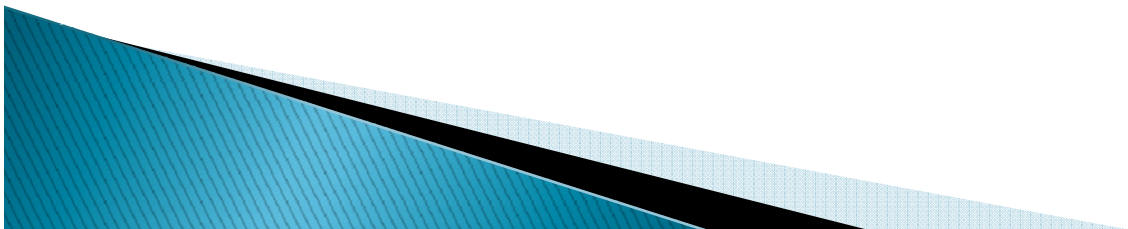
# Exercise

- ▶ *Exercise 1: Use File input/output*
- ▶ *Create a double way encryption/decryption machine, which uses a public key which is longer than 10 bytes to encrypt and decrypt `textfile.txt`*
- ▶ *So number of keys can be  $2^{(8*10)}$*



# Exercise

- ▶ *Exercise 2: Use File input/output*
- ▶ *Read all line of one code file and take out all comment lines then store in a new file.*
- ▶ *commentOut.exe*
- ▶ *Intput file1.txt and output file2.txt*
- ▶ *File1.txt is code file with comments*
- ▶ *File2.txt is new file without any comments*





# Exercise

- ▶ *Exercise 3:*
- ▶ *Apply the above exercise to develop a semi-automatic marking program which is used to mark your assignment 3 part 1.*
- ▶ *The program will read your txt file.*
- ▶ *Take away all lines started with #*
- ▶ *Read each line and compare with correct answers, if the same then add 1 to total mark.*
- ▶ *Output total mark and possible comments to a result.txt file.*

