

CSC111

# Administrivia

- No labs THIS week (week of March 14<sup>th</sup>)
  - Office hours instead (in all lab sections)
  - Labs resume week of March 21<sup>st</sup>
- Assignment 5 due date extended to Tuesday March 15<sup>th</sup>
- Assignment 6 Released today
- Midterm 2 – March 21<sup>st</sup> in class
  - Covers material up until and including today's class
  - 70 minutes
  - You are allowed one piece of letter paper with anything on it.
  - New material both days this week (not tested)

File I/O  
(file input/output)

# Steps to access a file for I/O

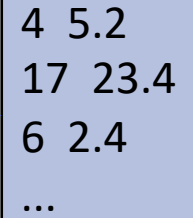
- Declare File pointer
  - `FILE* file_handle;`
- Open the file given its name and location
  - Use `file_handle = fopen("path", mode);`
- check to see the file was opened successfully
  - `if(file_handle == NULL){//error message}`
- Access the file + do operations
  - `fscanf(file_handle, "pattern", variable_pointer(s));`
  - `fprintf(file_handle, "pattern", variable(s));`
- Close the file
  - `fclose(file_handle);`

# Example reading from a file

```
FILE* file_handle;
int i;
double n;
file_handle = fopen("myfile.txt", "r");

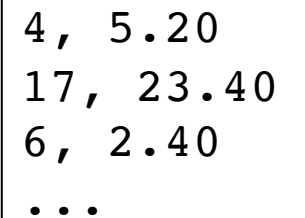
if (file_handle == NULL) {
    printf("error opening file\n");
} else {
    // read into i, n while 2 numbers are successfully read
    while( fscanf(file_handle, "%d %lf", &i, &n) ==2 ) {
        printf("%d, %.2f\n", i, n);
    }
    fclose(file_handle);
}
```

file\_handle



4 5.2  
17 23.4  
6 2.4  
...

**OUTPUT:**



4, 5.20  
17, 23.40  
6, 2.40  
...

# Example writing to a file

```
FILE* file_handle;
int i;
double limit = 100;
file_handle = fopen("myfile.txt", "w");

if (file_handle == NULL) {
    printf("error opening file\n");
} else {
    // overwrite myfile.txt with numbers 0 to limit-1
    for(i=0; i<limit; i++) {
        fprintf(file_handle, "%d\n", i);
    }
    fclose(file_handle);
}
```

processing unformatted  
input

# Variable type - char

- `char demo = 'c';`
  - Uses single quotes
- Stores a single character
- Print using the `%c` with `printf` or `fprintf`.
- `'\n'` is a character...



# reading unformatted input with `getchar`

Function included in `stdio.h`

## Documentation

Purpose: reads a single unformatted character from the keyboard

Parameters: none

Returns: `int`, the character read in as an unsigned char  
and cast to an `int`  
(can be stored in a char variable)

## Example

```
char ch;  
ch = getchar(); // ch is the first character the users enters
```

```
char val;  
printf("type text into the keyboard followed by the enter-key to stop\n");  
val = getchar();  
while (val != '\n'){  
    printf("%c\n", val);  
    val = getchar();  
}  
printf("done\n");
```

### Sample run:

type text into the keyboard  
followed by the enter-key to stop

hi !

h

i

!

done

Demo

# reading unformatted input with `fgetc`

Function included in `stdio.h`

## Documentation

Purpose: reads an unformatted character from the valid `file_handle`

Parameters: `FILE* file_handle`

Returns: `int`, the character read in as an unsigned char  
and cast to an `int`  
(can be stored in a `char` variable)

## Example

```
FILE* file_handle;  
// initialization and checking of file_handle omitted here  
char ch;  
ch = fgetc(file_handle); // ch is the next character in the file
```

```
FILE* file_handle = fopen("name.txt", "r");
```

```
if (file_handle != NULL) {  
    char val;  
    val = fgetc(file_handle);  
    while (val != EOF){  
        printf("%c\n", val);  
        val = fgetc(file_handle);  
    }  
    fclose(file_handle);  
    printf("done");  
}
```

**if the file is empty,**  
**Output:**  
done

**if the file contains:**

hi !!

**Output:**

h

i

!

!

done

# ctype.h

- The `ctype` library contains functions that operate on `char` types
- Given the prototypes and documentation of these functions, you should find them helpful in processing characters
- To use these functions in your program include the header file:  
`#include <ctype.h>`
- We do not expect you to memorize the functions but given the documentation you should know how and where to use them:

[https://www.tutorialspoint.com/c\\_standard\\_library/ctype\\_h.htm](https://www.tutorialspoint.com/c_standard_library/ctype_h.htm)

# cctype functions

- `isalnum(char c)`
  - Checks to see if a character is alphanumeric
- `isalpha()`
  - Checks to see if a character is alphabetic
- `isdigit()`
  - Checks to see if a character is a decimal digit
- `isupper()/islower()`
  - Checks to see if a character is upper/lower case
- `isspace()`
  - Checks for a white space characters
- ...and more
- Functions return a 1 if true and 0 if false.
- All accept a character as a parameter

```
char val;  
printf("type text into the keyboard followed by the enter-key to stop\n");  
val = getchar();  
while (val != '\n'){  
    printf("%c\n", val);  
    val = getchar();  
}  
printf("done\n");
```

### Sample run:

type text into the keyboard  
followed by the enter-key to stop

hi !

h

i

!

done



```
FILE* file_handle = fopen("name.txt", "r");
```

```
if (file_handle != NULL) {  
    char val;  
    val = fgetc(file_handle);  
    while (val != EOF){  
        printf("%c\n", val);  
        val = fgetc(file_handle);  
    }  
    fclose(file_handle);  
    printf("done");  
}
```

**if the file is empty,**  
**Output:**  
done

**if the file contains:**

hi !!

**Output:**

h

i

!

!

done

# cctype functions

- `isalnum(char c)`
  - Checks to see if a character is alphanumeric
- `isalpha()`
  - Checks to see if a character is alphabetic
- `isdigit()`
  - Checks to see if a character is a decimal digit
- `isupper()/islower()`
  - Checks to see if a character is upper/lower case
- `isspace()`
  - Checks for a white space characters
- ...and more
- Functions return a non zero value if true and 0 if false.
- All accept a character as a parameter

Demo – Modify demo 1 or 2...

# Design a function that....

- That will prompt the user to enter input through the keyboard.
- The function should compute and return the number of characters entered by the user that are not a space (ie. a newline, tab, space, etc).
- The user will enter an arbitrary number of characters and finally the value # to indicate that there are no more values to enter.
- Recall the functions included in ctype.h: `int getchar()` and `int isspace(int c)`

# Design a function that...

- Will prompt the user to enter input through the keyboard.
- The function should compute and print the percentage of characters entered by the user that are uppercase and the percentage of characters that are lowercase.
- The user will enter an arbitrary number of characters and finally the value # to indicate that there are no more values to enter.
- Print the percentage values with 1 significant figure. If no text was entered, print "no text entered".
- For example, if the user enters the following 5 characters: b 6a\$BCf#
  - The function should print: 25.0% uppercase and 37.5% lowercase since there are 2 uppercase letters, 3 lowercase letters and 3 other non-letter characters (space, 6 and \$).
  - Recall the functions included in ctype.h: `int isupper(int c)` and `int islower(int c)`