ICSI333 System Fundamentals

Note: Students are expected to start the activities as soon as the description is available and seek feedback as needed. Although some activities are not graded for credit, they are contiguous study of the lecture or used as stepping-stones for the projects. Skipping any activities would impact the learning significantly.

Objectives:

- Practice strings and characters

Reading:

- Lecture notes

Submission (5 points):

- All required C programs must be submitted on Duifene on time.

Instructions:

There are no objects in C. A C string is a null-terminated string and can be stored as an array of characters. A pointer to the first character of a string is often used when manipulating a string. A char variable can be used to store a character either as a character or an integer. The following example will display "`As a character: A, as a number: 65`".

```
char a = 65;
printf ("As a character: %c, as a number: %d",a, a);
```

Task #1. Read strings from `stdio`

Answer the following questions referring example 1:

- Why doesn't `scanf` need `&`?
- If each of the following strings were entered, what would be the outputs?

```
Michael Jordan
Michael Jordan Jordan
MichaelJordan
```

Example 1. Use `scanf()` to read a string.

```
#include <stdio.h>

int main(){
      char name[20];

      printf("Enter name: ");
      scanf("%s", name);
      printf("Your name is %s.", name);

      return 0;
}
```

Answer the following questions referring example 2:

- If each of the following strings were entered, what would be the outputs?

```
Michael Jordan
Michael Jordan Jordan
MichaelJordan
```

Example 2: Use `getchar()` to read a line of text.

```
#include <stdio.h>
```

```
int main(){
	char name[30], ch;
	int i = 0;

	printf("Enter name: ");
	while(ch != '\n'){ // terminates if user hit enter
		ch = getchar();
		name[i] = ch;
		i++;
	}
	name[i] = '\0'; // inserting null character at end
	printf("Name: %s", name);

	return 0;
}
```

Answer the following questions referring example 3:

- Why must we use `gets()` carefully?
- If each of the following strings were entered, what would be the outputs?

```
Michael Jordan
Michael Jordan Jordan
MichaelJordan
```

Example 3: Standard library function to read a line of text.

```
#include <stdio.h>
int main(){
	char name[30];

	printf("Enter name: ");
	gets(name); //Read string from user.

	printf("Name: ");
	puts(name); //Display string.

	return 0;
}
```

Task #2. Reverse String [optional]

Study the recursive solution to how to reverse a string (Fig. 8.10 from D&D + Lecture "More about Strings and Pointer Arithmetic") and the following example. Discuss the pros and cons of using recursions. Rewrite the following example using pointer arithmetic.

```
#include<stdio.h>
#include<string.h>

int main(){
	char str[100], temp;
	int i, j = 0;

	printf("\nEnter the string :");
	gets(str);

	i = 0;
	j = strlen(str) - 1;
	while (i < j) {
		temp = str[i];
		str[i] = str[j];
		str[j] = temp;
		i++;
		j--;
	}
```

```
        printf("\nReverse string is :%s", str);

        return (0);
    }
```

Task #3. Write a C program.

Choose one from below. You must use `scanf` for input.

- Given a user-entered string, count the number of uppercase and lowercase letters in the string without using standard functions, and display the result(s).
- Given two user-entered strings, concatenate two strings without using library functions, and display the result(s).
- Given a user-entered string, calculate the length of the string using pointer, and display the result(s).

Linux Basic Commands

`ls [option(s)] [file(s)]`	If you run ls without any additional parameters, the program will list the contents of the current directory in short form. `-l` detailed list `-a` displays hidden files
`cp [option(s)] sourcefile targetfile`	Copies sourcefile to targetfile. `-I` waits for confirmation, if necessary, before an existing targetfile is overwritten `-r` copies recursively (includes subdirectories)
`mv [option(s)] sourcefile targetfile`	Copies sourcefile to targetfile then deletes the original sourcefile. `-b` creates a backup copy of the sourcefile before moving. `-I` waits for confirmation, if necessary, before an existing targetfile is overwritten.
`rm [option(s)] file(s)`	Removes the specified files from the file system. Directories are not removed by rm unless the option `-r` is used. `-r` deletes any existing subdirectories `-I` waits for confirmation before deleting each file
`cd [options(s)] [directory]`	Changes the current directory. cd without any parameters changes to the user's home directory.
`mkdir [option(s)] directoryname`	Creates a new directory.
`rmdir [option(s)] directoryname`	Deletes the specified directory, provided it is already empty.
`cat [option(s)] file(s)`	The cat command displays the contents of a file, printing the entire contents to the screen without interruption. `-n` numbers the output on the left margin
`cal`	Displays the calendar of the current month.
`date`	Displays current time and date.
`whoami`	Reveals the user who is currently logged in.
`whatis`	Gives a one-line description about the command. It can be used as a quick reference for any command.

`man`	Manual Pages, for more detailed information, Linux provides man pages and info pages. To see a command's manual page, man command is used.
`pwd`	Prints the absolute path to current working directory.
`vi`	A text editor for Linux operating system.
`gedit`	The default GUI text editor in the Ubuntu operating system.
`emacs`	Another text editor for Linux operating system.

Vi commands:

To save and quit:

Commands	Action
:wq	Save and quit
:w	Save
:q	Quit
:w fname	Save as fname
ZZ	Save and quit
:q!	Quit discarding changes made
:w!	Save (and write to non-writable file)

More commands:

Copy-pasting within the terminal: Ctrl+Shift+C/V

Resources and Credits:

- Teaching materials from Professor Kuperman at UAlbany
- Vi reference: https://www.ele.uri.edu/faculty/vetter/Other-stuff/vi/vi-quick-ref.pdf