LAB 4 - TASK 8 through TASK 9

Simple Calculator / C Library Calls

John Dempsey
COMP-232: Programming Languages
California State University, Channel Islands
February 17, 2025
Hard Due Date: February 26, 2025

In Lab 4, we will complete the following tasks:

1. Task 8 – Simple Calculator
2. Task 9 – C Library Calls

TASK 8. Implement a Simple Calculator Using Function Pointers

Review sample.c below. Your program should be similar to myCaller() calls.

Once you understand what the program is doing, create a new file called **calc.c** which implements a simple calculator. The calculator will perform the four basic arithmetic operations: +, -, *, and /. The program should prompt the user for the operation to perform in an endless loop. For example:

calc> **3 + 6**

9

calc>

You must implement the calculator such that there is one calc function which takes as arguments the numerical value of the two operands and **a pointer to the specified function** (add for +, etc.), plugs the two values into the referenced function, and returns your result.

Your program should work independent of spaces in the input. For instance, both 1+2 and 1 + 2 should work. This is actually very easy to do with scanf. Check out its manual page (i.e., man fscanf).

john@oho:~/LAB4/CALC$ **cat c.c** ← **Sample Program To Understand**

```
#include <stdio.h>

void myProc(int);
void myProc2(int);

void myCaller(void (*)(int), int);

int main(void) {
	myProc(1);		// Call myProc with argument 1
	myProc2(2);		 // Call myProc with argument 2

	myCaller(myProc, 3);	// Call myProc with argument 3
	myCaller(myProc2, 4);	// Call myProc with argument 4
	return 0;
}

void myCaller(void (*f)(int), int param) {
	(*f)(param);		// call function *f with param
}

void myProc(int d) {
	printf("In myProc().\tParameter = %d\n", d);
}

void myProc2(int d) {
	printf("In myProc2().\tParameter = %d\n", d);
}
```

john@oho:~/LAB4/CALC$ **gcc c.c; a.out**

```
In myProc().	Parameter = 1
In myProc2().	Parameter = 2
In myProc().	Parameter = 3
In myProc2().	Parameter = 4
```

TASK 9. C Library Calls

The purpose of this assignment is to practice using additional library calls in a program. The library calls are defined in assert.h, ctype.h, stdlib.h, string.h, and time.h. These are commonly used library calls.

This is an open assignment meaning you can write a program to do anything you like so long as you use at least once each of the C library functions listed below.

To receive full credit, you need to implement each of the library calls below at least once in your program.

assert.h			
assert			
ctype.h			
isalnum	islower	tolower	
isdigit	isupper	toupper	
stdlib.h			
atof	calloc	malloc	system
atoi	free	realloc	
string.h			
strcat	strcpy	strncat	strstr
strchr	strerror	strncmp	strtok
strcmp	strlen	strncpy	
time.h			
asctime	difftime	localtime	sleep

Here are examples of what to code for each function call. You can code your own examples if you like.

assert.h

- assert → Validate that 10+20 is equal to 18 + 12.

---

ctype.h

- isalnum → Check how many alphanumeric characters are in this string: Pass==Word$123
- islower / isupper → Count lowercase vs uppercase letters in Pass==Word$123
- tolower / toupper → Convert the string Pass==Word$123 to all lowercase and then to all uppercase.
- isdigit → Count number of digits in the string Pass==Word$123

---

stdlib.h

- atoi / atof → Convert string input (e.g., 1234 and 56.78) into integer and floating-point number, then print the integer and floating-point number.
- malloc, calloc, realloc, free → Allocate 8192 bytes of memory using malloc, then allocate 4096 of initialized memory using calloc, double the size for one of the memory segments created, then free both memory segments.
- system → Run a system command (e.g., system("clear; ls -l") on Linux to clear screen and list all files in the current directory).

---

string.h

- strlen → Print out the string length for "CSUCI – California State University Channel Islands".
- strcpy / strncpy → Copy the string "CSUCI – California State University Channel Islands" into string, then use strncpy to extract just CSUCI.
- strcat / strncat → Use strncat to copy the string "apple" from the string "apple pie", then use strcat to append " tree" to the string to print "apple tree".
- strcmp / strncmp → Compare the strings CSUCI and CSULB, and print out if they match or not. Then compare the first three characters of CSUCI with CSU and print out if these two strings match.

- strchr → Find the first occurrence of ‘A’ in “TODAY’S THE DAY!”
- strstr → Search for a word in the string. Find “DAY” in “TODAY’S THE DAY!”
- strtok → Tokenize a sentence into words. Print out all tokens in the following string: Tokenization is a process of converting input text into smaller units.
- strerror → Open a file that doesn’t exist. Pass errnum to strerror to print out a description of the error.

---

time.h

- time, localtime, asctime → Use these three functions to print the current local date/time in the following format:

  MM/DD/YYYY hh:mm

  where MM is month, DD is day, YYYY is year, hh is hour, and mm is minutes, e.g., 02/26/2025 15:30 or 02/26/2025 03:30 PM
- difftime → Show how many seconds elapsed between two actions (start/end).
- sleep → Pause program for 4 seconds.