LAB 4 - TASK 8 through TASK 9

Simple Calculator / C Library Calls

John Dempsey
COMP-232: Programming Languages
California State University, Channel Islands
September 13, 2023
Hard Due Date: September 20, 2023
Soft Due Date:October 4, 2023

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 it 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).

You must use function pointers instead of a switch statement.

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.

Your program does not have to be useful, but simply make use of each of the library calls.

1. Here are the library calls to use:

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

One approach could be to add a comment like "// assert.h example" then use the assert function call. Followed by "// ctype.h examples" followed by code to use

the isalnum, isdigit, islower, isupper, tolower, and toupper function calls, and so on.