Course Name: Computer Architecture Lab

Course Number and Section: **14:332:333:03**

Experiment: [Experiment # 2 – Introduction to C Programming Language]

Lab Instructor: Christos Mitropoulos

Date Performed: September 26 2018

Date Submitted: October 10 2018

Submitted by: Vancha Verma 173004061

Course Name: Computer Architecture Lab

Course Number and Section: **14:332:333:03**

! Important: Please include this page in your report if the submission is a paper submission. For electronic submission (email or Sakai) please omit this page.

----------For Lab Instructor Use ONLY----------

GRADE: _____

COMMENTS:

Electrical and Computer Engineering Department
School of Engineering
Rutgers University, Piscataway, NJ 08854
ECE Lab Report Structure

Exercise 1:

1. Explain the changes you made.
V0 decided the number of time "RU" was printed by the for loop. Since it has to be printed 3 times, i changed the value to be 3. V1 decided which case from the switch case was printed. To print "Werblin Rec Center", I picked the third one. Thus the value of V1 is now 3. To print "Go", V3 needed to be 3 since the rest would "Boo". By changing V2 to a nonzero name, I was able to print "Rutgers" rather than "Penn State"

2. Explain the minimum number of distinct values needed for the preprocessor macros.
The minimum value required it 1. When all the values are equal to 3, it was give the expected response.

3. What does the -o flag do with gcc?
It indicated that the file was just filed by gcc will have the selected name rather than "a.out".

Exercise 2:

1. Explain how do you set the breakpoint at main, and how you run up to that breakpoint.
To set breakpoint, I typed in the command: (gdb) b hello c:main.

2. A list containing the additional gdb commands.

Command	Defination
run	run any command
q or quit	exit gdb
next	execute the next line of C
step	used to debug functions
s	single step command
print	print a list variables and their current values
display variable	prints the value of all the variable at every step

Exercise 3:

1. Explain the bug and your fix to the function.
The code had a logic error. The while loop must check the both a *and* b are not equal to null before running.

Exercise 4:

1. Describe how you run CGDB to completion on the executable created by compiling interactive_hello.c without getting stuck.
To run the CGDB without getting stuck, use standard input/output using redirection. "a.out < inputfile" can be used to take an input.

Exercise 5:

1. Implement ll_cycle.c with the completed ll_has_cycle() function. Comment your code and provide explanation for your solution.

```
//create both pointers and set them equal to the first node in the linked list

node* hare = head;

node* tortise = head;


//runs through the linked list till either hare equals null or they are at the same value. don't need to check tortise because hare will hit all nodes before tortise can

while (hare != NULL){

     hare = hare->next; //first hare iteration

     //make sure its not null

     if (hare == NULL){

          return 0;

     }//end if

     hare = hare->next; //second hare iteration
```

```
        tortise = tortise->next; //iterate tortise


        //if they are at the same spot, return 1 to indicated that
        the list is cyclical

        if (hare == tortise){

             return 1;

        }//end if

}//end while
```

.