Subject: Procedures and floating-point operations

Project Programs (75%): You are required to write the following three programs:

Program 1: Practice Leaf Procedures (*15 points*)

Create a MIPS assembly code that reads three positive numbers ***a***, ***b,*** and ***c*** as input parameters. The code shall execute in MARS to prompt the user to enter three positive integers represented in decimal, each separated by Enter key. The program shall find the **minimum number that is not equal to zero**. Your program should print out the minimum number on the screen.

Output Sample

a	b	c	Expected F
0	5	3	3
100	50	33	33
78	0	450	78
10005	2000	3500	2000

Program 2: Practice floating-point operations (*30 points*)

Create a MIPS assembly code that calculates the BMI for a person based on the following formula:

BMI = Weight (lbs)/Height(inches)^2 x 703.

Please read the following requirements:

1. All data should be defined as single precision floating point data.
2. The code shall execute in MARS to prompt the user to enter the weight (in pounds) and height (in feet).
3. Your program should convert feet into inches, then apply the inches value in the equation. Please see the example below for clarification.
4. After the program calculates the BMI, the program should identify the person as underweight,

normal, or overweight based on the following conditions:

a. If BMI < 18.5, the person is *underweight*.

b. If BMI >= 18.5 and BMI <= 24.9, the person is *normal* weight.

c. If BMI >= 25.0, the person is *overweight.*

5. The program should continue by asking the user to input different numbers.

Example:

Assume, you are required to find the BMI and the status of a person whose weight is 150 lbs and height is 5.5 feet using your program.

1. First, the program asks the user to input the weight in pounds and height in feet
2. After, the user inserted the weight = 150 lbs and height = 5.5 feet
3. The program converts feet into inches by following this formula: *Inches = Feet x 12*
4. Then, the program applies the inches and pounds in the BMI formula given above. BMI = Weight (lbs)/Height(inches)^2 x 703 = (150lbs/66inches^2) x 703 = 24.2.
5. Then, the program determines the user status, which is in this case it is normal. Below is a sample of output.

```
**BMI Calculator by Rasha**

Enter weight in pounds: 150
Enter height in inches: 5.5


Enter weight in pounds: 24.207989
 BMI index is Normal
Enter weight in pounds: 150
Enter height in inches: 7.7


Enter weight in pounds: 12.351016
 BMI index is Underweight
Enter weight in pounds: 300
Enter height in inches: 5.5


Enter weight in pounds: 48.415977
 BMI index is Overweight
Enter weight in pounds:
```

Program 3: Practice non-leaf procedures and stacks (*30 points*)

Create a MIPS assembly code that calculates your total work time in hours. The work is calculated based on number of homeworks and the number of exercises the person accomplishes per day. There is an average time (in hours) to complete each homework and finish each exercise. For example, a person completed three homeworks and two exercises in a day. The average time to complete homework for this person is one hour and the average time to complete an exercise is two hours. The total time to complete all work is 3 hws x 1 hour + 2 exercises x 2 hours = 7 hours. The following C code is a program that calculate the average work time per day. You can use it as a reference for you to write your MIPS assembly code.

Notes:

a. The program inputs are:
    a. number of homeworks.
    b. average time to complete each homework in hours.
    c. number of exercises.
    d. average time to complete each exercise in hours.

b. The program output is:
    a. Total work time in hours.

c. Consider the inputs and output as **positive integer** numbers.

d. To insert the inputs, write a code to prompt the user to enter four positive integers represented in decimal, each separated by Enter key.

e. You program should print out the total work time.

f. Remember you are required to use nested (non-leaf) procedure and stack to preserve the return address

Output Sample

#hws	#hw_hours	#exercises	#ex_hours	Total work hours
3	1	2	4	11
1	2	3	2	8
4	1	1	2	6

Sample of the program in C language

```
int hw_func (int num_hws, int avg_hours) {
      int total_hw_time;
      total_hw_time = num_hws x avg_hours;
      return total_hw_time;
}
int exercise_func (int num_exercises, int avg_hours) {
      int total_exercise_time;
      total_exercise_time = num_exercises x avg_hours;
      return total_exercise_time;
}
int total (int nhws, int hours, int nexercises, int ehours) {
      int total;
      total = hw_func (nhws, hours) + exercise_func (nexercises, ehours);
      return total;
}
void main( ) {
      int num_homeworks = 4;
      int average_hws_hours = 1;
      int num_exercises_per day = 2;
      int average_exercise_hours = 2;
      int      total_time      =      total(num_homeworks,      average_hws_hours,
num_exercises_perday, average_exercises_hours);
printf("%d\n", total_time);
}
```

Project Report (40%)

The project report will be graded out of 100, and the points will be distributed as following:

A. **Professional preparation:** Submit the report as typed document saved as a PDF.

B. Report Content (100 points):

You are required to provide the answers for the following numbered section headings:

1.0 **Program Input/Output (20 points):** Describe (or list) the inputs and outputs of each program.

2.0 **Program Design (20 points):** Describe briefly how each code (program 1, program 2 and program 3) operates.

3.0 **Symbol Table (20 points):** Create a 2-column Table describing all Registers used and their specific Purpose in the code, where each register is listed on a separate row and identified by register name $t0, $s0, etc., as well as any Labels used and their purpose on separate rows. You may take the table below as sample.

Registers	Purpose & Labels
$a0	Argument of syscall to print string
:	:

4.0 **Learning Coverage (20 points):** Provide a list of at least 5 technical topics learned from this project.

5.0 **Test Results (20 points):** Provide screen shot(s) of **two 2** proper MIPS code executions in MARS for your Test Plan inputs. To illustrate, you are required to execute program 1, program 2 and program 3 at least 2 times and, in each time, you enter different input sets, and you get different outputs. Take a screen shot of each execution (“MARS Message” console) and include it in your report.

Project Submission

1. Create a **folder** and name it as **p2_username**, where *username* is youruniversity username.
2. In MARS, save the code of program 1 as **c1.asm**, save the code ofprogram 2 as **c2.asm** and program 3 as **c3.asm**.
3. Save your **report** as a PDF file and name it as **p2_report.pdf**.
4. Place the three asm files and the report document inside p1_usernamefolder.
5. Compress (or zip) the folder.
6. Then, upload the compressed folder on Blackboard.