Ron Joshua Recrio
Github root directory: https://github.com/recrio/submissions.2/tree/master/Lab%205

Date Submitted: 12/13/2019

Task 00: Execute provided code

Task 01:

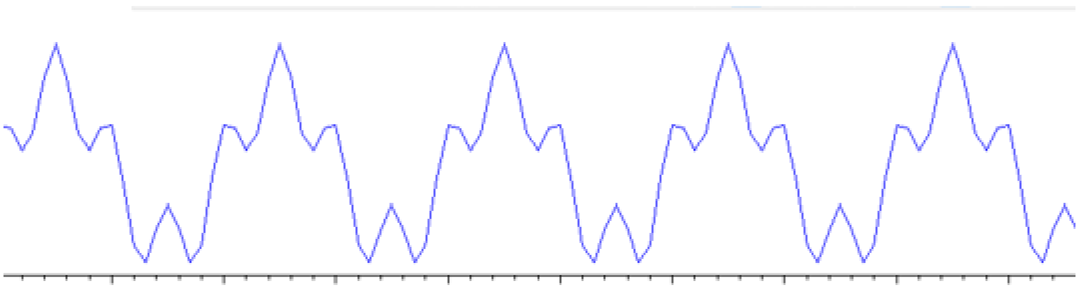
Modified Code:

```
}
#include <stdint.h>
#include <stdbool.h>
#include <math.h>
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "driverlib/fpu.h"
#include "driverlib/sysctl.h"
#include "driverlib/rom.h"
// If there is no previous definition of PI, Define it here.
#ifndef M_PI
#define M_PI                    3.14159265358979323846
#endif
// Define a series length of 100
#define SERIES_LENGTH 100
// A float array of length 100
float gSeriesData[SERIES_LENGTH];
// Counter i = 0;
int32_t i32DataCount = 0;

int main(void)
{
    float fRadians;
    // Faster speed for less accuracy
    ROM_FPULazyStackingEnable();
    ROM_FPUEnable();
    // Set Clock speed
    ROM_SysCtlClockSet(SYSCTL_SYSDIV_4 | SYSCTL_USE_PLL | SYSCTL_XTAL_16MHZ |
SYSCTL_OSC_MAIN);
    // 2pi/100
    fRadians = ((2 * M_PI) / SERIES_LENGTH);
    // for i < 100;
    while(i32DataCount < SERIES_LENGTH)
    {
        // Saving points in a sine wave in gSeries Data
        gSeriesData[i32DataCount] = sinf(fRadians * i32DataCount);
        // i++
        i32DataCount++;
    }

    while(1)
    {
    }
}
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

Task 02:

gSeriesData:



```
Modified Code:
#include <stdint.h>
#include <stdbool.h>
#include <math.h>
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "driverlib/fpu.h"
#include "driverlib/sysctl.h"
#include "driverlib/rom.h"
// If there is no previous definition of PI, Define it here.
#ifndef M_PI
#define M_PI 3.14159265358979323846
#endif
// Define a series length of 100
#define SERIES_LENGTH 100
// A float array of length 100
float gSeriesData[SERIES_LENGTH];
// Counter i = 0;
int32_t i32DataCount = 0;

int main(void)
{
    float fRadians;
    // Faster speed for less accuracy
    ROM_FPULazyStackingEnable();
    ROM_FPUEnable();
    // Set Clock speed
    ROM_SysCtlClockSet(SYSCTL_SYSDIV_4 | SYSCTL_USE_PLL | SYSCTL_XTAL_16MHZ |
SYSCTL_OSC_MAIN);
    // 2pi/100
    fRadians = ((2 * M_PI) / SERIES_LENGTH);
    // for i < 100;
    while(i32DataCount < SERIES_LENGTH)
    {
        // Saving points in a sine wave in gSeries Data
        gSeriesData[i32DataCount] = (1.5 + sinf(fRadians * (50*i32DataCount)) +
(0.5*cosf(fRadians*(200*i32DataCount)));
        // i++
        i32DataCount++;
    }
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

```
    while(1)
    {
    }
}
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.