---

**Lab Exercise 13**            **Threading Applications**

1. Write a multithreaded program that calculates various statistical values for a list of numbers. This program will be passed a series of numbers on the command line and will then create three separate worker threads. One thread will determine the average of the numbers, the second will determine the maximum value, and the third will determine the minimum value. For example, suppose your program is passed the integers

    90 81 78 95 79 72 85

    The program will report
        The average value is 82
        The minimum value is 72
        The maximum value is 95

    The variables representing the average, minimum, and maximum values will be stored globally. The worker threads will    set these values, and the parent thread will output the values once the workers have exited. (We could obviously expand this program by creating additional threads that determine other statistical values, such as median and standard deviation.)

Example:
  Main thread creates a thread to find the summation of given 'n'

```
#include < pthread.h>
#include <stdio.h>
int sum; /* this data is shared by the thread(s) */
void *runner(void *param); /* threads call this function */
int main(int argc, char *argv[])
{
pthread_t tid; /* the thread identifier */
pthread_attr_t attr; /* set of thread attributes */
if (argc != 2)
{
fprintf(stderr,"usage: a.out <integer value>\n");
return -1;
}
if (atoi(argv[1]) < 0)
{
```

```c
        fprintf(stderr,"%d must be >= 0 \n",atoi(argv[1]));
        return -1;
    }
    /* get the default attributes */
    pthread_attr_init(&attr);
    /* create the thread */
    pthread_create(&tid,&attr,runner,argv[1]);
    /* wait for the thread to exit */
    pthread_join(tid,NULL);
    printf("sum = %d\n",sum);
}
/* The thread will begin control in this function */
void *runner(void *param)
{
    int i, upper = atoi(param);
    sum = 0;
    for (i = 1; i <= upper; i++)
        sum += i;
    pthread_exit(0);
}
```