Assignment T03

Data Structure:

Code:

in thread.h

```
struct thread
 {
.
.
.
.
 int nice_value; // Nice value of a thread
 int recent_cpu; // Recent Cpu usage value of a thread
 };
```

In thread.c

```
static int load_avg; // This int stores the load average value of the system
static struct thread *manager_thread; // an idle_thread like thread used to wake sleeping threads
static struct thread *bsd_thread;     // an idle_thread like thread used to update priority, recent_cpu value of each thread on 4th tick and update load average value on 100th tick
```

New Functions added:

```
void thread_update_priority (struct thread *);
// Updates priority of a thread acc. To the formula based on nice and recent_cpu value as given in pintdoc
void thread_update_recent_cpu (struct thread *);
// Updates recent_cpu value of a thread acc. To the formula given in pintdoc.
void thread_update_load_avg (void);
//  Updates load_average value of the system acc. To the formula given in pintdoc.
void timer_wakeup (void);
// a function which wakes up sleeping threads at their waking time.
```

```
static void manager (void *aux UNUSED);
// a function which states the operations done by manager thread
static void bsd_scheduler (void *aux UNUSED);
// a function which states the operations done by bsd_thread.
// on 4th tick update thread_priority and recent_cpu value
// on 100th tick update load_avg value
```

Functions modified

```
int thread_get_nice (void);
// returns nice value of current running thread.
void thread_set_nice (int);
// sets nice value of the current thread
int thread_get_recent_cpu (void);
// returns recent_cpu value of the current thread.
int thread_get_load_avg (void);
// returns average load value of the system.
```

Algorithm:

Task-01:

1. Made a managerial thread in same pattern as idle thread.
2. This thread gets un-blocked in thread_tick function whenever current ticks becomes equal to the next wake up time. (i.e. a thread needs to be woken up)
3. When this thread is unblocked, it calls timer wakeup function and does necessary steps to wake up the threads.
4. After waking up all the appropriate threads, its blocked again.

Task-02:

1. Made a bsd thread in same pattern as above managerial thread.
2. This thread gets un-blocked in thread_tick function whenever it is fourth tick or 100th tick.
3. If it is fourth tick, then updates priority value of each thread. If it is 100th tick then it updates load average value of the system and updates recent_cpu value of each thread. (Functions used thread_update_priority, thread_update_recent_cpu, thread_update_load_avg).

4. After performing above operations this thread gets blocked.

Task-03:

We incrementing current running thread’s recent cpu value by one on each thread tick. Now on fourth tick we call bsd_thread, which by itself manages decrementing of priority by one. Hence task-03 is done.