OS-PG Pintos Project

Team details: (Hungry Fools)

B Prabhakar - 201505618
Kuldeep - 201505524
Ankit - 201505523

Part 2

- Defined a funtion *hello_task()* in *init.c* and I am calling the funtion from the *run_actions()* function.
- *helloc_task()* has a call to *run_hello()* function, which is defined in *src/tests/threads/MyFile.c*
- A prototype of the *run_hello()* is delared in the *src/tests/threads/MyFile.h*
- The header file *src/tests/threads/MyFile.h* is inluded in the *src/tests/threads/MyFile.c* and *src/threads/init.c*
- How to run the code?
  prabhakar@laptop:~/OS/pintos/src/threads$ pintos hello

Part 3

- Data Strutures created/modified:
  + Added a variable *sleep_ticks* to the thread struture, for keeping track of the number of ticks the thread has to sleep.
- Test cases passed:
  + alarm-single
  + alarm-zero
  + alarm-negative
  + alarm-simultaneous
- Logic:
  Whenever, a call to *timer_sleep()* comes, I am setting the *sleep_ticks* and then blocking the thread. At every clock tick, I am cheking if the *sleep_ticks* has dropped to zero. If yes, I am unbloking the corrsponding thread.

Part 4

- Data Strutures created/modified:
  + added *get_heighest_priority_element()* to *thread.h* and *thread.c*
- Test cases passed:
  + alarm-priority
  + alarm-change
  + alarm-fifo
- Logic:
  Whenever, a new thread is created or priority of a thread is decreased, I am calling the *get_heighest_priority_element()* to check if there exists a thread with priority higher than the current thread. If yes, I am calling the *thread_yield()* to shedule it.