```
#include "mthread_internal.h"
#include <assert.h>
#include <stdarg.h>
#include <string.h>
#ifdef TWO_LEVEL
#include <pthread.h>
#endif


void
__not_implemented (const char *func, char *file, int line)
{
  fprintf (stderr, "Function %s in file %s at line %d not implemented\n",
           func, file, line);
  abort ();
}

void *safe_malloc(size_t size){
  void * tmp;
  tmp = malloc(size);
  assert(tmp ≠ NULL);
  return tmp;
}

#ifdef TWO_LEVEL
static pthread_mutex_t mthread_fprintf_lock = PTHREAD_MUTEX_INITIALIZER;
#endif
static char* mthread_output_log_name = "mthread_log";
static FILE* mthread_output_log = NULL;

int mthread_log_init(){
  mthread_output_log = fopen(mthread_output_log_name,"w");
  return 0;
}

#define MTHREAD_LOG_PART 15

int mthread_log(char* part, const char *format, ...){
  char msg[4096];
  char part2[MTHREAD_LOG_PART +1];
  va_list ap;
  int i;
  int len;
#ifdef TWO_LEVEL
  int res;
  pthread_mutex_lock(&mthread_fprintf_lock);
#endif
  for(i = 0; i < MTHREAD_LOG_PART; i++){
    part2[i] = ' ';
  }
  len = strlen(part);
  if(len ≥ MTHREAD_LOG_PART){
    len = MTHREAD_LOG_PART;
  }
```

```
  memcpy(part2,part,len);
  part2[MTHREAD_LOG_PART] = '\0';

  sprintf(msg,"[LWP %02d Thread %p %s INFO:] %s",mthread_get_vp_rank(),
          mthread_self(),part2,format);

  va_start(ap, format);
  res = vfprintf(mthread_output_log, msg, ap);
  va_end(ap);
  fflush(mthread_output_log);
#ifdef TWO_LEVEL
  pthread_mutex_unlock(&mthread_fprintf_lock);
#endif
  return res;

}

int fprintf(FILE *stream, const char *format, ...){
  va_list ap;
#ifdef TWO_LEVEL
  int res;
  pthread_mutex_lock(&mthread_fprintf_lock);
#endif
  va_start(ap, format);
  res = vfprintf(stream, format, ap);
  va_end(ap);
  fflush(stream);
#ifdef TWO_LEVEL
  pthread_mutex_unlock(&mthread_fprintf_lock);
#endif
  return res;
}
```