CS 537
Section 6
Programming Assignment 3

Michael Swift

P3

- Due date: Tuesday 11/24
- Goal: write a multithreaded program

- Write a desktop search engine
  - Scanner: creates a list of files to scan
  - Indexer: makes lists of words, adds them to index
  - Searcher: takes words from user, looks them up in index
- Use threads
  - 1 scanner thread
  - 1 searcher thread
  - n indexers

Your job

- I wrote the index data structure
- The scanner is (for now) a file passed on the command line

1. Coordinate access to index
2. Coordinate passing file names from scanner to indexers
   1. use a bounded buffer
3. Parse files into words
   1. use strtok from project 0

Creating threads with pthreads

- pthreads is the official API for threads on Unix

- creating a thread:
  ```
  int pthread_create(pthread_t * thread,
                     pthread_attr_t * attr,
                     void (*start_routine)(void *),
                     void * arg);
  ```
- thread = receives thread identifier
- attr = optional attributes, pass as NULL
- start_routine = function to call
- arg = argument to pass

Example

```
void * sample_thread(void * arg)
{
  char * param = (char *) arg;
  printf("This thread reads %s\n",param);
  return(NULL);
}
int main(int argc, char * argv[])
{
  int result;
  pthread_t thd1;

  result = pthread_create(&thd1, NULL, sample_thread, "Hello");
  pthread_join(thd1, NULL);
  return(0);
}
```


Waiting for a thread

- pthread_join() waits for a thread to terminate
- int pthread_join(pthread_t thread, void **value_ptr);
- The return from the thread is returned as the value
- Q: what if you don't wait for the threads you create?

```
char test_string[] = "Test";

void * sample_thread(void * arg)
{
  char * param = (char *) arg;
  printf("This thread reads %s\n",param);
  return(test_string);
}
Main:
char * res_string1;
pthread_create(&thd1, NULL, sample_thread, "hello");
pthread_join(thd1, (void **) & res_string1);
printf("Thread 1 returned %s\n",res_string1);
```


Compiling threads

- #include <pthread.h>
- gcc –pthread file.c

pthread locking

- pthread locks are called mutexes.
- usage:
  - pthread_mutex_t mutex;
  - pthread_mutex_init(&mutex, NULL);
  - pthread_mutex_lock(&mutex);
  - pthread_mutex_unlock(&mutex);
  - pthread_mutex_destroy(&mutex);

Locking Example

```
int gobal_variable;
pthread_mutex_t lock;

void * sample_thread(void * arg)
{
  pthread_mutex_lock(&lock);
  global_variable++;
  pthread_mutex_unlock(&lock);
  return(NULL);
}
main()
{
  pthread_mutex_init(&lock, NULL);
  // create threads;
  // join threads;
  pthread_mutex_destroy(&lock);
}
```