CS241 Lecture 14 Lawrence Angrave
Working With threads and locks.

0 Is the following code 'dangerous' on a 64 bit machine?

```
int bad = (int) "Hello";
puts( (char*) bad);
```

1. Where are the critical sections in the following code?
Fix any errors you notice.
Modify the code to be thread safe

```
link_t* head;


void*list_insert(int v) {
  link_t* link = malloc( sizeof(link_t*)));
  link->value = v;
  link-> next = head;
  head = link;
}
link* list_remove() {
   link_t* result = head;
   if(result) head = result->next;
   return result;
}
```

2. Notice any mistakes? What do you expect to happen?

```
pthread_t tid1,tid2;
pthread_mutex_t m;

int counter;
void*myfunc2(void*param) {
 int i=0; // stack variable
 for(; i < 1000000;i++) {
   pthread_mutex_lock( &m);
   counter ++;
  }
 return NULL;
}
int main() {
 pthread_create(&tid1, 0, myfunc2, NULL);
 pthread_create(&tid2, 0, myfunc2, NULL);
 pthread_join(tid1,NULL);
 pthread_join(tid2,NULL);
 printf("%d\n", counter );
}
```

3. What is a counting semaphore?

3. Case study: Parallelize *AngraveCoin* miner

```
void search(long start, long end) {
  printf("Searching from 0x%lx to 0x%lx\n", start , end);
  for(long i = start; i <end; i++) {
    char message[100];
    sprintf(message,"AngraveCoin:%lx", i);

    unsigned char *res;

    res = SHA256(message, strlen(message), NULL);
    int iscoin;
    iscoin = (res[0] == 0)&&(res[1] == 0)&&(res[2] == 0);

    if(iscoin)
        printf("%lx %02x %02x %02x '%s'\n", i, res[0],
res[1], res[2] , message);
  }
  printf("Finished %lx to %lx\n", start, end);
}


long array[] = {0L, 1L <<25, 1L <<27, 1L <<33};
int main() {
  search(array[0], array[1]);
  search(array[1], array[2]);
  search(array[2], array[3]);
  return 0;
}
```