3 Assignment: Hello,Pintos again

3.1 Description

Do you remember your first homework? Now, you need to rewrite **void hello_pintos(void)**. Before you start writing code. You need to read **struct thread** , **Understanding Thread**, **Thread Functions** and the whole fifith part --**List**.And maybe you also need to read part of **Synchronization**.

Before you start rewriting, you need to add some code in right place.

```
/* use to store integer */
struct sortNumber {
  int number;
  struct list_elem elem;
};
```

Add this struct in init.h or thread.h.

```
static struct list number_list;
static struct lock number_lock;
```

Add this part in init.c

You can start rewriting **void hello_pintos(void)** now.

In the new function, you need to print a message like "Hello, your name" at first. And then, you should create three threads. The first thread is used to produce some integers in **random** and insert them in **number_list**. The second thread is used to sort the number_list. The number should be in descending order. And the third thread is used to print the number after sort.

tips:

You can get random number by function **void random_bytes (void *, size_t)** in random.h,

Because you are asked to use multithreading to implement. So you should pay attention to the order of the thread executing.You can use lock to make the thread execute in order.

Timetable of the thread execute:

	time				
thread_name	1	2	3	4	5
hello_pintos	execute	wait	wait	wait	execute
produce_n	create	execute	block	block	block
sort_n	create	wait	execute	block	block
print_n	create	wait	wait	execute	block

You should make your program execute like single-thread.

Sample output:

```
========================================================================
                     Bochs x86 Emulator 2.6.2
               Built from SVN snapshot on May 26, 2013
                 Compiled on Dec  9 2014 at 12:18:09
========================================================================
00000000000i[     ] reading configuration from bochsrc.txt
00000000000e[     ] bochsrc.txt:8: 'user_shortcut' will be replaced by new 'ke
00000000000i[     ] installing nogui module as the Bochs GUI
00000000000i[     ] using log file bochsout.txt
PiLo hda1
Loading..........
Kernel command line: -q run hello-pintos
Pintos booting with 4,096 kB RAM...
383 pages available in kernel pool.
383 pages available in user pool.
Calibrating timer...  204,600 loops/s.
Boot complete.
Executing 'hello-pintos':
(hello-pintos) begin
hello LiAng
Produce_number:
46 25 9 -32 0 64 -47 18 64 -6 -8 -58 41 -21 68 -44 1 16 -3 50 16 19 -46 -60 -2
 2 -6 -69 23 3 57 39 10 -12 46 -53 49 -13 47 -19 17 16 -16 40 42 -50 -4 0 -20
3 35 -41 48 -37 -47 -47 62 -35 33 -12 -66 -56 -25 0 17 61 51 59 -30 -69 -42 35
15 0 -45 -61 44 -47
Result:
-69 -69 -66 -66 -65 -63 -61 -60 -58 -56 -56 -53 -50 -47 -47 -47 -47 -46 -46 -4
 -32 -30 -29 -29 -25 -22 -21 -20 -19 -16 -15 -13 -12 -12 -8 -6 -6 -6 -4 -3 -2
16 16 16 17 17 18 19 20 23 25 28 32 33 35 35 39 40 41 42 42 43 44 46 46 47 48
2 64 64 66 67 68 68
(hello-pintos) end
Execution of 'hello-pintos' complete.
Timer: 145 ticks
Thread: 0 idle ticks, 147 kernel ticks, 0 user ticks
Console: 1073 characters output
Keyboard: 0 keys pressed
Powering off...========================================================================
Bochs is exiting with the following message:
```

I used one hundred number here.

3.1 Requirement

- 1.Read **struct thread**, **Understanding Thread**, **Thread Functions** and the whole fifith part --**List**.
- 2.Rewrite **void hello_pintos(void**) in accordance with the requirements.
- 3.download and **fill** the **DESIGNDOC** and archive your code(**the whole pintos folder**) and DESIGNDOC into a **zip** file **named sid_nameInPinYin_vid.zip(e.g 12330441_zhangsan_v0.zip)** and upload to the ftp.