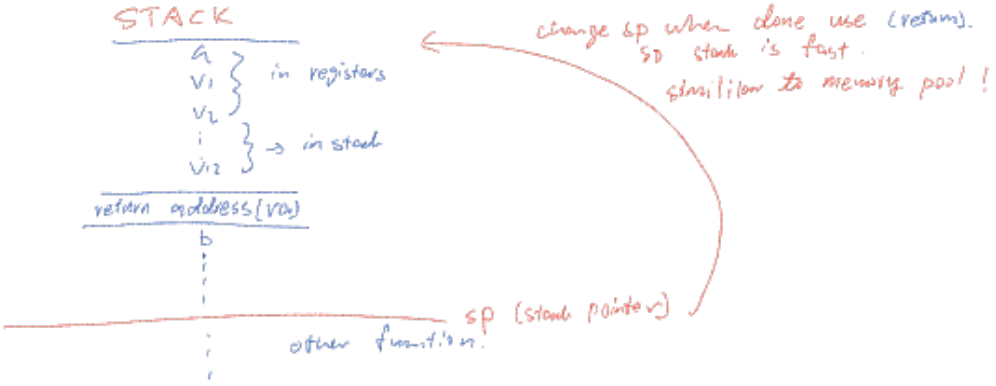
#1 Thread of Execution

```
int f(int a, int v1,int v2, ...10 more) {
   int b=a+1;
   if(b>99) return 3;
    return b;
}
```

How does return 'work'? i.e. How does the CPU know what to do next?

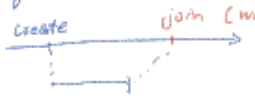


#2 Introduction to threads and pthreads

pthread_create

pthread_join

pthread_exit



inside thread, call exit (last thing in threads)
That thread is DONE!!

```
#include <pthread.h>

int pthread_create(pthread_t *thread,
   pthread_attr_t *attr,
   void *(*start_routine) (void *),
   void *arg);

int pthread_join(pthread_t thread, void **retval);

void pthread_exit(void *retval);
```

function pointer

(return value would be exit value)

for

Compile and link with -pthread

#3 My program calls pthread_create twice. How many stacks does my process have?

3 threads including main

#4 What is the difference between a process and a thread?

threads live in process

```c
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

void* hello(void*arg) {
  sleep(1);
  printf("hello!\n");
  pthread_exit(NULL); // nothing happens after this

  return NULL;
}

int main(int argc, char**argv) {
  void *ptr = "Banana";
  pthread_t tid;
  pthread_create( &tid, NULL, hello, ptr );

  return EXIT_SUCCESS;
}
```

not waiting for threads, so nothing will print

```c
void* hello(void*arg) {
  sleep(1);
  char* mesg = arg;

  printf("hello %s!\n", mesg);

  if( mesg[0] == 'T') {
    pthread_exit((void*) 0xdeadc0de); // nothing happens after this
  }

  return (void*) 0xdeadbeef;
}

int main(int argc, char**argv) {
  void *ptr = "Banana";
  pthread_t tid1, tid2;
  pthread_create( &tid1, NULL, hello, ptr );
  pthread_create( &tid2, NULL, hello, "Tomato" );
  void* retValue1;
  void* retValue2;
  pthread_join( tid1,& retValue1);
  pthread_join( tid2,& retValue2);

  printf("retValue1 is %p", retValue1) ;
  printf("retValue2 is %p", retValue2) ;
```

avoid typos!!!

Before returning, a successful call to **pthread_create**() stores the ID of the new thread in the buffer pointed to by *thread*; this identifier is used to refer to the thread in subsequent calls to other pthreads functions.

#5 What does `pthread_cancel` do?
and are there alternatives?

can stop, but

common alternative: globle pls_stop = 1

#6 Differences between `exit()` and `pthread_exit()`?

stops the whole process

stops thread

...so why would you call `pthread_exit` in your main method?

#7 Give four ways that a thread can be terminated

p exit

return

exit

main threads return from main

signal that terminates

#8 Hello World with pthreads?

#9 What happens if I call pthread_create 100 or 10000 times?