Introduction

In the first part of this book we will introduce the foundations of our topic.

Overview

What a grand adventure we have begun.

Listing 1 simple python code block

```
print("hello, world")
print("hello, world")
print("hello, world")
print("hello, world")
```

Here is the example C code

Listing 2 testfork.c

```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int
main(int argc, char* argv[])
{
  pid_t mypid, cpid, ppid;
  cpid = fork();
  ppid = getppid();
  mypid = getpid();

  if (cpid > 0) {
    /* Parent code */
    printf("hello, from parent with pid %d, pid of child is %d\n", mypid,
cpid);
  } else if (cpid == 0) {
    /* Child code */
    printf("hello, I am child with pid %d, my parent is %d\n", mypid, ppid);
  } else {
    perror("fork failed\n");
    exit(-1);
  }
  return 0;
}
```

Here is another example C code

Listing 3 doforke.c

```c
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/wait.h>
#include "doforke.h"


int
do_fork_exec(char *prog, char *const argv[])
{
  pid_t cpid;
  int status=0;
  cpid = fork();
  if (cpid < 0) {
    perror("fork failed\n");
    exit(-1);
  }

  if (cpid != 0) {
    // parent code, we need to wait for child
    waitpid(cpid,&status,0);
  } else {
    execve(prog, argv, 0);
    perror("should never get here\n");
  }
  return status;
}
```