

必应

你好，这是Bing。我可以告诉你如何获取Python程序的返回值。 😊

如果你使用popen或者system函数，你可以通过读取标准输出或者标准错误来获取Python程序的输出。例如，如果你的Python程序是：

```
import sys
print("Hello from Python!")
sys.exit(42)
```

你可以在C语言中这样获取它的输出和返回值：

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    // 使用popen函数
    FILE *fp = popen("python test.py", "r");
    if (fp == NULL) {
        printf("popen failed!\n");
        return -1;
    }
    char buffer[1024];
    while (fgets(buffer, sizeof(buffer), fp) != NULL) {
        printf("%s", buffer);
    }
    int status = pclose(fp);
    printf("The return value is: %d\n", WEXITSTATUS(status));
    // 使用system函数
    status = system("python test.py");
    printf("The return value is: %d\n", WEXITSTATUS(status));
    return 0;
}
```

这样，你就可以看到Python程序的输出和返回值。

如果你使用fork和execve函数，你可以在父进程中使用wait或者waitpid函数来获取子进程的状态，然后使用WEXITSTATUS宏来获取返回值。例如，你可以在C语言中这样获取Python程序的返回值：

```
#include <stdio.h>
#include <unistd.h>
int main() {
    // 使用fork和execve函数
    pid_t pid = fork();
    if (pid == -1) {
        printf("fork failed!\n");
        return -1;
    }
    if (pid == 0) {
        // 子进程
        char *argv[] = {"python", "test.py", NULL};
        char *envp[] = {NULL};
        execve("/usr/bin/python", argv, envp);
        printf("execve failed!\n");
        return -1;
    }
    else {
        // 父进程
        int status;
        waitpid(pid, &status, 0); // 等待子进程结束并获取状态
        printf("The return value is: %d\n", WEXITSTATUS(status));
        return 0;
    }
}
```

这样，你就可以看到Python程序的返回值。