**前言**

Linux有一个特性，叫做“一切皆文件”。Linux将所有的设备、文件和操作都视为文件，并提供了统一的文件操作接口。这种特性带来了许多优点和应用，例如统一的文件操作接口、方便的文本处理和系统管理、方便的设备管理和更高的安全性。

**1. linux的文件类型**

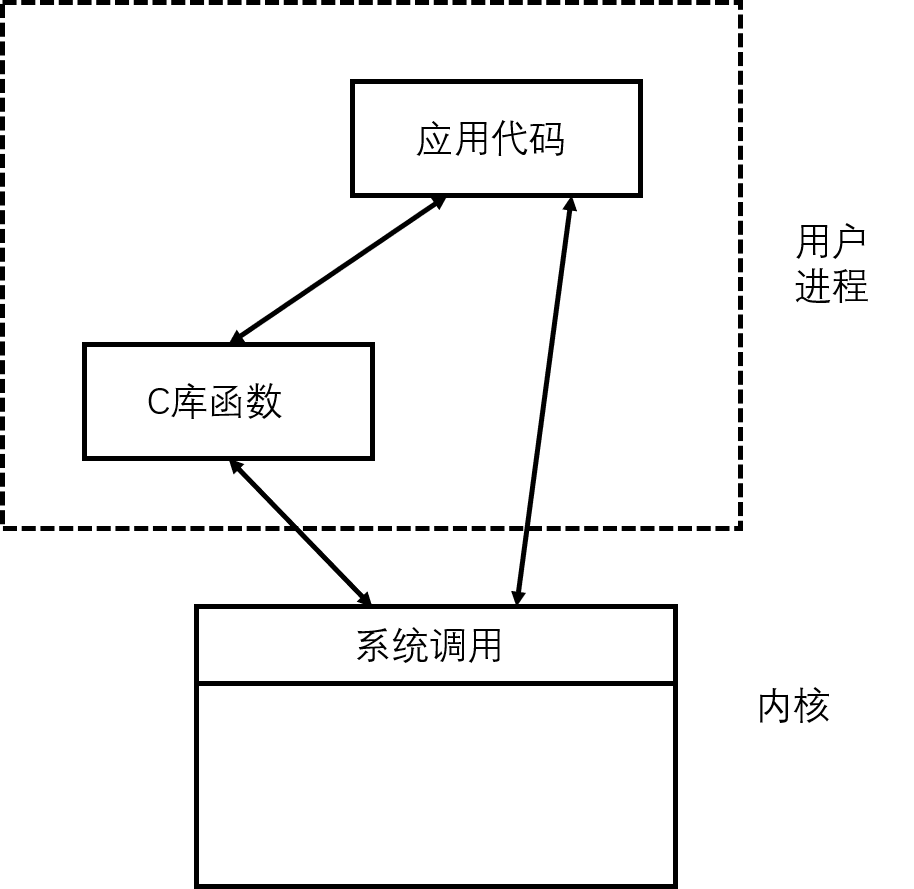
|  |  |
| --- | --- |
| 文件类型 | 文件属性 |
| 常规文件 | r |
| 目标文件 | d |
| 字符设备文件 | c |
| 块设备文件 | b |
| 管道文件 | p |
| 套接字文件 | s |
| 符号链接文件(相当于window快捷方式) | l |

**2. IO的概念**

|  |  |
| --- | --- |
| I(input) | 输入设备，比如键盘、鼠标 |
| O(output) | 输出设备，比如显示器 |
| I/O | 既使输入也是输出，比如U盘、网口 |

**3. 系统调用和库函数**

系统调用就是操作系统提供的接口函数。如果我们把系统调用封装成库函数就可以起到隔离的作用，提供程序的可移植性。Printf就是库函数然后调用了系统调用才在显示器上显示字符。如下图所示，用户的应用代码可以通过封装成库函数来进行系统内核的操作，也可以通过系统调用直接进行系统内核操作。

​编辑

**4. 流的概念**

标准IO用结构体类型来存放打开文件的相关信息，标准IO的所有操作都是围绕FILE来进行的。FILE又被称为流(stream)。流主要分为文本流和二进制流。

**5. Windows和linux的换行符区别**

**5.1 Window换行符**

* 文本流：换行符为'\n'
* 二进制流：换行符为'\r''\n'

**5.2 Linux换行符**

* 换行符为'\n'

**6. 流的缓冲类型**

|  |  |
| --- | --- |
| 缓冲类型 | 特点 |
| 全缓冲 | 当流的缓冲区无数据或无空间时才执行实际I/O操作 |
| 行缓冲 | 当在输入和输出中遇到换行符('\n')时，进行I/O操作 |
| 无缓冲 | 数据直接写入文件，流不进行缓冲 |

**7. 标准I/O**

标准I/O预定义3个流，程序运行时自动打开

|  |  |  |  |
| --- | --- | --- | --- |
| 标准输入流 | 0 | STDIN\_FILENO | stdin |
| 标准输出流 | 1 | STDOUT\_FILENO | stdout |
| 标准错误流 | 2 | STDERR\_FILENO | stderr |

**8. 缓冲区实验**

**8.1 实验测试**

**8.1.1 编写代码**

使用printf打印"hello world\n"，代码如下，其中while循环是为了防止程序执行结束。

#include <stdio.h>//标准IO头文件

int main(int argc,char\* argv[])

{

printf("hello world\n");

while(1)

{

sleep(1);

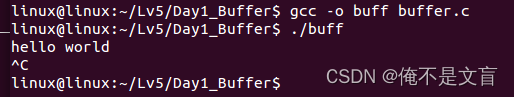
}

return 0;

}

**8.1.2 编译输出**

编译输出内容如下图

​

编辑

说明当在输出中遇到换行符('\n')，进行了输出操作。

**8.1.3 实验修改1**

去掉换行符'\n'

#include <stdio.h>//标准IO头文件

int main(int argc,char\* argv[])

{

printf("hello world");

while(1)

{

sleep(1);

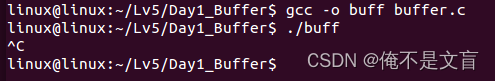
}

return 0;

}



hello world并没有成功打印，使用Ctrl+C终止程序

​编辑

**8.1.4 实验修改2**

引入for循环进行1024个字符打印

#include <stdio.h>//标准IO头文件

int main(int argc,char\* argv[])

{

int i=0;

for(i=0;i<1024;i++)

{

printf("a");

}

//printf("hello world\n");

while(1)

{

sleep(1);

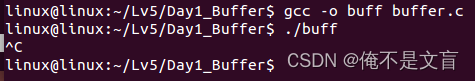
}

return 0;

}



 依然没有成功打印字符，使用Ctrl+C终止程序

​编辑

**8.1.5 实验修改3**

引入for循环进行1025个字符打印

#include <stdio.h>//标准IO头文件

int main(int argc,char\* argv[])

{

int i=0;

for(i=0;i<1025;i++)

{

printf("a");

}

//printf("hello world\n");

while(1)

{

sleep(1);

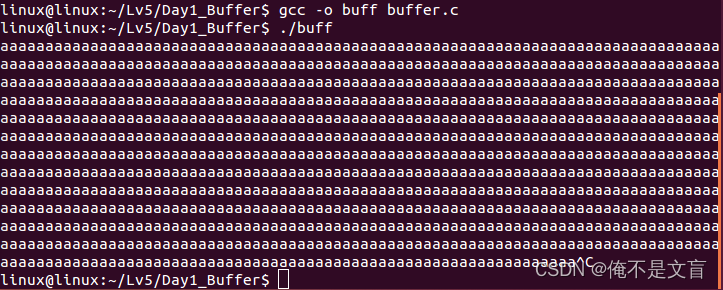
}

return 0;

}



 成功打印字符

​编辑

**8.2 实验结论**

根据实验结果，标准输出的缓冲模式为行缓冲模式，且最大的容量为1024个字节。