Linux程序设计2021笔试回忆

简答题

1.比较硬链接与软链接的区别。写出创建硬链接与软链接的方式（命令行和系统调用）。（12分）

```
//硬：ln <source_file> <link_name>

#include <sys/link.h>
int result = link(source_file, link_name);
    if (result == -1) {
        perror("link error");
        return 1;
}

//软：ln -n <source_file> <link_name>

#include <sys/link.h>
int result = symlink(source_file, link_name);
    if (result == -1) {
        perror("link error");
        return 1;
}
```

2.bash/shell中输入重定向的方式是什么？输入重定向有哪些类型。输入重定向的典型使用案例是什么？重定向的实现原理及参数是什么？（12分）

输入重定向可以分为两种类型：

标准输入重定向（stdin redirection）:将命令行输入重定向到指定的文件中，例如:command < input.txt。

直接输入重定向（input redirection）:将命令行输入重定向到一个命令中，例如:command > output.txt。

典型使用案例包括：

将命令的输出重定向到一个文件中，例如:command > output.txt。这个案例常用于调试命令，将命令的输出重定向到一个文件中，可以方便地查看命令的执行情况。
将标准输入重定向到一个命令中，例如:command < input.txt。这个案例常用于将外部命令的输出作为输入传递给该命令，例如使用cat命令将文件内容传递给grep命令。
重定向的实现原理是将命令行输入流的指针移动到指定文件中，或者将指定的命令作为参数传递给该命令。在 bash/shell 中，可以使用>符号来表示标准输入重定向，使用>符号和文件名来表示直接输入重定向。例如:command < input.txt表示将输入重定向到input.txt文件中，而command > output.txt表示将标准输入重定向到output.txt文件中。

在bash/shell中,输入重定向的参数是>符号和文件名。文件名可以是通配符,例如command < input.txt*表示将输入重定向到所有名为input.txt的文件。如果文件名中包含通配符，则需要使用星号（*）来匹配所有文件，而不是单个文件。

3.`mknod /dev/zero15 c 1 5`是什么意思？各个参数有什么作用？字符型Linux驱动的创建过程是什么？（12分）

4#define的宏是什么？const修饰的常量是什么？C/C++中的函数在功能上有什么相似之处？在使用上有什么不同之处。（14分）

相似之处：

define和const都可以用于定义字符串常量、数字常量、符号常量等。
define和const都可以对变量进行限制，防止程序员在代码中修改常量的值。
不同之处：

define是编译时操作，它在代码编译时就已经确定了常量的值，而在运行时不会改变。而const则是运行时操作，它在代码运行时不能修改常量的值。
define是对单个字符进行定义，例如#define MAX 100,它会在代码中插入一个符号MAX,代表一个常量100。而const则是用于定义变量的常量属性，例如const int a = 100,它会定义一个整型常量a,它的值不能被修改。
define的作用域仅限于定义它的源文件中，而const的作用域是整个程序。
define和const的语法不同，define使用双冒号 (#define) 来定义，而const使用关键字const来定义。
总之，define和const都是用于定义常量的工具，但它们的使用方式和含义略有不同。define主要用于定义符号常量，而const主要用于定义变量的常量属性

编程题

1.用系统调用实现一个copy.cpp模块，输入两个参数（原文件名，目标文件名），将原文件中的内容复制到目标文件中。并且编写Makefile编译链接上述模块。（35分）

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

void copy_file(char *source_file, char *target_file)
{
    int fd_source, fd_target;
    char buffer[1024];
    int len;

    // 打开源文件
    fd_source = open(source_file, O_RDONLY);
    if (fd_source == -1) {
        perror("open source file error");
        exit(EXIT_FAILURE);
    }

    // 打开目标文件
    fd_target = open(target_file, O_CREAT | O_TRUNC | O_WRONLY, 0666);
    if (fd_target == -1) {
        perror("open target file error");
        exit(EXIT_FAILURE);
    }

    // 读取源文件内容
    while ((len = read(fd_source, buffer, sizeof(buffer))) > 0) {
```

```
        write(fd_target, buffer, len);
    }

    // 关闭文件
    close(fd_source);
    close(fd_target);
}

int main(int argc, char *argv[])
{
    if (argc < 2) {
        printf("Usage: %s source_file target_file\n", argv[0]);
        exit(EXIT_FAILURE);
    }

    char *source_file = argv[1];
    char *target_file = argv[2];

    copy_file(source_file, target_file);

    return 0;
}
```

```
all: copy

copy: copy.o
    gcc -o copy copy.o -static -lshell32

o:
    gcc -o $@ $^ -static -lshell32
```

客观题

2021年4月限定特典，和最后一节课的华为操作系统讲座有关，基本是送分题。（15分）