

|      |                  |      |          |
|------|------------------|------|----------|
| 课程名称 | 操作系统             | 课程编号 | A2130330 |
| 实验地点 | C410/411         |      |          |
| 实验名称 | Assignment3 进程通信 |      |          |

### 一、实验内容

使用 Ubuntu 操作系统或任何其他 Linux 发行版来创建两个 C 程序。从一个包含整数的文件中读取，并将输入文件中的偶数和奇数的和写入显示器的程序 consumer.c。一个程序 producer.c，创建一个包含 20 个整数的文件，并与消费者共享该文件。创建的文件必须命名为 numbers.txt。消费者程序和生产者程序都必须使用普通管道进行通信。你的解决方案必须包含一个子进程来执行你的消费者程序。你可以使用 fork()、pipe()、dup()、dup2()、read()、write()、open()、create()，以及任何 exec()系列系统调用。

### 二、实验步骤及方案

1. producer 中，pipe()创建管道，fork()创建父子进程。
2. 父进程中关闭 fd[0]。按照要求打印出父子进程的 pid。使用 open 创建 numbers.txt。将数字 0-19 通过 write 写入 numbers.txt，注意 write 写入应为指针类型，因此需使用 sprintf 将 int 转为字符串后写入。数字写入完成后，使用 write 将文件名写入到 pipe 中，供子进程中 consumer 使用。
3. 子进程中，关闭 fd[1]。从 pipe 中读出文件名，并作为参数，在 execl 调用 consumer 时传入。
4. consumer 中已传入文件名，使用 open 打开 numbers.txt。从中读出数字并存入数组。借助 for 循环，使用与 1 的按位与操作判断 0-19 的奇偶性，输出偶数和奇数和。

### 三、结果及分析

实验源码:

Producer.c:

```
#include<stdio.h>
#include<stdlib.h>
```

```

#include<unistd.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
#include <sys/wait.h>

int main()
{
    int p[2]; //pipe
    int pid;
    int status;

    // Generates pipe
    if(pipe(p) < 0)
    {
        perror("Pipe Error");
        exit(1);
    }

    // Generates child process
    pid = fork();
    if(pid < 0)
    {
        perror("Problem forking");
        exit(1);
    }

    // Child process
    if(pid == 0)
    {
        close(p[1]); // Close the write pipe
        char FileName[20] = {0}; // To store the file name

        printf("Child Process: My pid = %d. My parent pid = %d.\n", getpid(),
getppid());

        read(p[0], FileName, 11); // Gets file name from parent process
        execl("./consumer", "consumer", FileName, NULL); // Executes consumer
        exit(0);
    }
    else
    {
        close(p[0]); // Close the read pipe
        char FileName[] = "numbers.txt";
        int fd;
    }
}

```

```

printf("Parent Process: My pid = %d. I created child pid = %d.\n",
getpid(),pid);

// Creates numbers.txt
fd = open("numbers.txt", O_WRONLY|O_CREAT, 0777);
if(fd == -1)
{
    perror("Producer File Error");
    exit(1);
}
printf("File numbers.txt fd is:%d\n", fd);

// Generates 20 integers into file
char s[sizeof(int)] = {0};
for(int i = 0; i < 20; i++)
{
    sprintf(s, "%d", i);
    write(fd, s, sizeof(int));
}

write(p[1], FileName, sizeof(FileName)); // Pass the name of file to child
wait(&status);
}
return 0;
}

```

### Consumer.c:

```

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

int main(int argc, char *argv[])
{
    int sum = 0;
    int fd;
    int numbers[20];
    char s[sizeof(int)] = {0};

    // Opens the file
    fd = open(argv[1], O_RDONLY);
    if(fd == -1)
    {
        perror("Consumer File Error");
    }
}

```

```

        exit(1);
    }

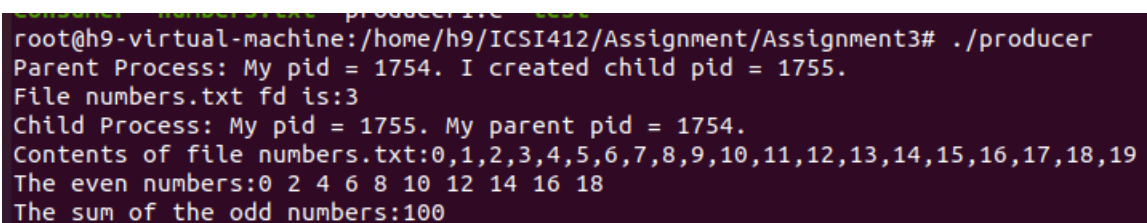
    // Reads 20 integers from file
    printf("Contents of file numbers.txt:");
    for(int i = 0; i < 20; i++)
    {
        read(fd, s, sizeof(int));
        numbers[i] = atoi(s);
        if(i < 19)
        {
            printf("%d,",numbers[i]);
        }
        else
        {
            printf("%d\n",numbers[i]);
        }
    }

    // Outputs the even number and caculates the sum of odd numbers
    printf("The even numbers:");
    for(int i = 0; i < 20; i++)
    {
        if((numbers[i] & 1 )== 0)
        {
            printf("%d ", numbers[i]);
        }
        else
        {
            sum += numbers[i];
        }
    }
    printf("\nThe sum of the odd numbers:%d\n", sum);

    return 0;
}

```

实验结果如下图所示：



```

root@h9-virtual-machine: /home/h9/ICSI412/Assignment/Assignment3# ./producer
Parent Process: My pid = 1754. I created child pid = 1755.
File numbers.txt fd is:3
Child Process: My pid = 1755. My parent pid = 1754.
Contents of file numbers.txt:0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19
The even numbers:0 2 4 6 8 10 12 14 16 18
The sum of the odd numbers:100

```

图 1 输出结果

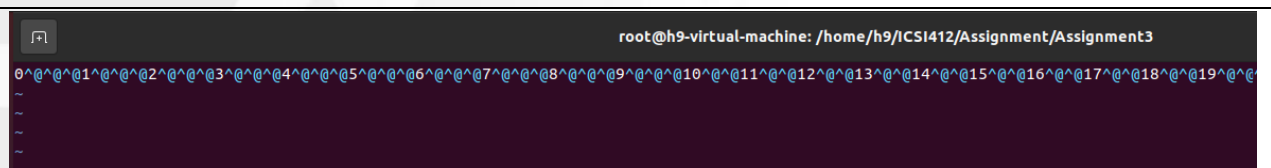
A terminal window with a dark background. The title bar shows 'root@h9-virtual-machine: /home/h9/ICSI412/Assignment/Assignment3'. The terminal displays a single line of text: '0^@1^@2^@3^@4^@5^@6^@7^@8^@9^@10^@11^@12^@13^@14^@15^@16^@17^@18^@19^@'. Below this line are several tilde (~) characters, indicating that the output has been truncated.

图 2 numbers.txt 内容

### 结果分析:

1. numbers.txt 的 fd 为 3 并成功输出。因在创建 txt 之前, 已创建管道, 故 fd=3、4 已被占用。但关闭了父进程的输出管道, 故 fd=3 空缺出来, 被 txt 使用。
2. 父进程 pid=1754, 子进程 pid=1755 在父子进程中成功分别输出。
3. number.txt 的内容成功全部输出。
4. numbers.txt 中偶数成功一一输出, 并最终输出了偶数和。
5. 任务的全部要求成功完成, 输出结果如图。其中 numbers 中内容, 因使用 `sprintf` 向 txt 中写入数字时固定了字符串长度为 int 单位长度 (4 字节), 故结果形式如图所示。

### 四、心得体会

1. 熟练掌握了进程的管理与父子进程间的通信。
2. 熟练掌握并使用了管道通信。
3. 熟练掌握并使用了 `exec` 族函数。
4. 熟练掌握了 linux 中 c 语言 int 与字符串的相互转换, 即借助 `sprintf` 和使用 `atoi`。
5. 可进一步将写入的数字改为随机数。