Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Курсовой проект**

**“Реализация трех программ, связанных между собой конвейерами”**

**по курсу**

**«Операционные системы»**

Студент: Уваров А. П.

Группа: М8О–206Б–19

Преподаватель: Соколов Андрей Алексеевич

Оценка: \_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2020.

**Постановка задачи**

**Задача:** Создание трех программ, которые будут обмениваться данными через pipe. Программа А берет ввод от пользователя и пересылает строки программе С, продолжает пересылать ввод от пользователя построчно после того, как С вернет информацию о получении предыдущей строки. Программа В выводит количество байт переданных программе С и количество байт принятых программой С.

**Общие сведения о программе**

Для работы используются pipe, в нужный момент запускаются программы B, C и в аргументах пересылаются дескрипторы нужных pipe'ов. Те выводят на экран требуемую информацию.

**Основные файлы программы**

**Makefile:**

**all: run**

**run: getl.o**

**gcc A.c getl.o -o A**

**gcc B.c getl.o -o B**

**gcc C.c getl.o -o C**

**getl.o: getl.c**

**gcc -c getl.c**

**clean:**

**rm A B C getl.o**

**A.c:**

**#include <stdio.h>**

**#include <unistd.h>**

**#include <stdlib.h>**

**#include <signal.h>**

**#include <string.h>**

**#include "getl.h"**

**int id1, id2;**

**int pAC[2];**

**int pAB[2];**

**int pCA[2];**

**int pCB[2];**

**int main(){**

**pipe(pAC);**

**pipe(pAB);**

**pipe(pCA);**

**pipe(pCB);**

**id1 = fork();**

**if (id1 == -1)**

**{**

**perror("Fork trouble!\n");**

**return 1;**

**}**

**else if(id1 == 0)**

**{**

**char dAB[3] = "";**

**sprintf(dAB, "%d", pAB[0]);**

**char dCB[3] = "";**

**sprintf(dCB, "%d", pCB[0]);**

**close(pAC[0]);**

**close(pAC[1]);**

**close(pCA[0]);**

**close(pCA[1]);**

**close(pAB[1]);**

**close(pCB[1]);**

**char fileName[] = "./B";**

**execl(fileName, fileName, dAB, dCB, NULL);**

**}**

**else**

**{**

**id2 = fork();**

**if (id2 == -1)**

**{**

**perror("fork2");**

**return 1;**

**}**

**else if(id2 == 0)**

**{**

**char dAC[3] = "";**

**sprintf(dAC, "%d", pAC[0]);**

**char dCA[3] = "";**

**sprintf(dCA, "%d", pCA[1]);**

**char dCB[3] = "";**

**sprintf(dCB, "%d", pCB[1]);**

**close(pAC[1]);**

**close(pCA[0]);**

**close(pAB[0]);**

**close(pAB[1]);**

**close(pCB[0]);**

**char name[] = "./C";**

**execl(name, name, dAC, dCA, dCB, NULL);**

**}**

**else**

**{**

**char\* line = NULL;**

**int size;**

**while((size = get\_line(&line, 0)) != 0)**

**{**

**write(pAC[1], &size, sizeof(int));**

**write(pAC[1], line, size\*sizeof(char));**

**int ok;**

**read(pCA[0], &ok, sizeof(int));**

**write(pAB[1], &size, sizeof(int));**

**}**

**free(line);**

**kill(id1, 10);**

**kill(id2, 10);**

**}**

**}**

**close(pAC[0]);**

**close(pAC[1]);**

**close(pCA[0]);**

**close(pCA[1]);**

**close(pAB[0]);**

**close(pAB[1]);**

**close(pCB[0]);**

**close(pCB[1]);**

**}**

**B.c:**

**#include <stdio.h>**

**#include <unistd.h>**

**#include <stdlib.h>**

**#include <signal.h>**

**int pAB;**

**int pCB;**

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

**pAB = atoi(argv[1]);**

**pCB = atoi(argv[2]);**

**int sizeA;**

**int sizeB;**

**while(read(pAB, &sizeA, sizeof(int)) > 0 && read(pCB, &sizeB, sizeof(int)) > 0){**

**printf("[B programm] A length: %d; C length: %d\n", sizeA, sizeB);**

**}**

**}**

**C.c:**

**#include <stdio.h>**

**#include <unistd.h>**

**#include <stdlib.h>**

**#include <signal.h>**

**int pAC;**

**int pCA;**

**int pCB;**

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

**pAC = atoi(argv[1]);**

**pCA = atoi(argv[2]);**

**pCB = atoi(argv[3]);**

**int sizeA;**

**while(read(pAC, &sizeA, sizeof(int))> 0){**

**char line[sizeA+1];**

**line[sizeA] = '\0';**

**read(pAC, line, sizeA \* sizeof(char));**

**printf("[C Programm] Got from A: %s\n", line);**

**int good = 1;**

**write(pCA, &good, sizeof(int));**

**write(pCB, &sizeA, sizeof(int));**

**}**

**}**

**Getl.c:**

**#include <unistd.h>**

**#include <stdlib.h>**

**#include <stdio.h>**

**#include "getl.h"**

**int get\_char(int fd)**

**{**

**char c;**

**if (read(fd, &c, 1) == 1)**

**return (unsigned char)c;**

**return EOF;**

**}**

**int get\_line(char\*\* line\_, int fd)**

**{**

**free(\*line\_);**

**int c;**

**int size = 0;**

**int cap = 4;**

**char\* line = (char\*)malloc(cap \* sizeof(char));**

**while((c = get\_char(fd)) != EOF && c != '\n' && c!= '\r')**

**{**

**if(size == cap)**

**{**

**cap \*= 2;**

**line = (char\*)realloc(line, cap \* sizeof(char));**

**if(line == NULL)**

**exit(-1);**

**}**

**line[size] = c;**

**++size;**

**}**

**if(size == cap)**

**{**

**cap += 1;**

**line = (char\*)realloc(line, cap \* sizeof(char));**

**if(line == NULL)**

**exit(-1);**

**}**

**line[size] = '\0';**

**\*line\_ = line;**

**return size;**

**}**

**Пример работы**

andrey@andrey-VivoBook:~ /Рабочий стол/OC/OChelp/KP/good$ ./A

qwertyu

[C Programm] Got from A: qwertyu

[B programm] A length: 7; C length: 7

xcvbnm,nbvcvbnm,.

[C Programm] Got from A: xcvbnm,nbvcvbnm,.

[B programm] A length: 17; C length: 17

uygfhgvf

[C Programm] Got from A: uygfhgvf

[B programm] A length: 8; C length: 8

test for kp labs

[C Programm] Got from A: test for kp labs

[B programm] A length: 16; C length: 16

**Вывод**

Я реализова л общение между 3 независимыми программами через конвейеры, я применил передачу дескриптора не всего конвейера, а лишь ввода или вывода. Снова познакомился с командой execl и fork.