Обработка ошибки perror()

#include <stdio.h>

#include <fcntl.h>

#include <errno.h>

main()

{

int fd;

/\* Открыть несуществующий файл, чтобы открыть ошибку \*/

fd = open("nonexist.txt", O\_RDONLY);

if (fd == -1) /\* Ошибка произошла \*/

{

printf("errno = %d\n", errno);

perror("main");

}

fd = open("/", O\_WRONLY); /\* Вызвать другую ошибку \*/

if (fd == -1)

{

printf("errno = %d\n", errno);

perror("main");

}

/\* Выполнить успешный системный вызов \*/

fd = open("nonexist.txt", O\_RDONLY | O\_CREAT, 0644);

printf("errno = %d\n", errno); /\* Внести после успешного вызова \*/

perror("main");

errno = 0; /\* Вручную установить переменную ошибки \*/

perror("main");

}

#include <stdlib.h>

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <sys/types.h>

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

{

int fd;

ssize\_t ret;

char ch;

ssize\_t read\_bytes;

ssize\_t written\_bytes;

char buffer[BUFFER\_SIZE];

fd = open(argv[1], O\_RDONLY);

//обработка ошибок

if (fd < 0)

{

fprintf(stderr, "Cannot open file\n");

exit(1);

}

//чтение

while ((ret = read(fd, &ch, 1)) > 0)

{

putchar(ch);

}

//запись

while ((read\_bytes = read(fd, buffer, BUFFER\_SIZE)) > 0)

{

/\* 1 == stdout \*/

written\_bytes = write(1, buffer, read\_bytes);

if (written\_bytes != read\_bytes)

{

fprintf(stderr, "Cannot write\n");

exit(1);

}

}

//поиск

lseek(fd, 0, SEEK\_SET);

lseek(fd, 20, SEEK\_CUR);

lseek(fd, -10, SEEK\_END);

close(fd);

}

// замена владельца

#include <stdlib.h>

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <sys/types.h>

int main()

{

int flag;

flag = chown("test.txt", -1, 62); /\* Оставить ID пользователя без изменений \*/

if (flag == -1) perror("mychown.c");

}

//изменение прав доступа

#include <stdlib.h>

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <sys/types.h>

int main()

{

int flag;

flag = chmod("test.txt", 0600); /\* Использовать восьмеричную кодировку \*/

if (flag == -1) perror("mychmod.c");

}

//дублирование

#include <stdlib.h>

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <sys/types.h>

int main()

{

int fd1, fd2, fd3;

fd1 = open("test.txt", 0\_RDWR | 0\_TRUNC);

printf("fd1 = %d\n", fd1);

write(fd1, "what's", 6);

fd2 = dup(fd1); /\* сделать копию fd1 \*/

printf("fd2 = %d\n", fd2);

write(fd2, " up", 3);

close(0); /\* Закрыть стандартный ввод\*/

fd3 = dup(fd1); /\* сделать другую копию fd1 \*/

printf("fd3 = %d\n", fd3);

write(0, " doc", 4);

dup2(3, 2); /\* дублировать канал 3 в канал 2 \*/

write(2, "?\n", 2);

}

//действия с дексриптором

#include <stdio.h>

#include <fcntl.h>

main()

{

int fd;

fd = open("test.txt", O\_WRONLY); /\* Открыть файл для записи \*/

write(fd, "hi there\n", 9);

lseek(fd, 0, SEEK\_SET); /\* Поиск начала файла \*/

fcntl(fd, F\_SETFL, O\_WRONLY | O\_APPEND); /\* Установить флаг APPEND \*/

write(fd, " guys\n", 6);

close(fd);

}

//драйвер для num lock

#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

#include <sys/io.h>

#include <sys/ioctl.h>

#include <linux/kd.h>

#include <stdlib.h>

#include <math.h>

int change()

{

int fd;

fd = open("/dev/tty0", O\_NOCTTY);

if (fd == -1)

{

perror("open");

return -1;

}

int state = 0;

if (-1 == ioctl(fd, KDGKBLED, &state))

{

perror("ioctl");

return -1;

}

state ^= K\_NUMLOCK;

if (-1 == ioctl(fd, KDSKBLED, state))

{

perror("ioctl set");

close(fd);

return -1;

}

close(fd);

return 0;

}

int main()

{

while (true)

{

char t = getchar();

switch (t)

{

case ' ':

change();

break;

default:

break;

}

}

return 0;

}

//содержание связей

int main()

{

link("original.txt", "another.txt");

}

//усечение файла

int main()

{

truncate("file1.txt", 10);

truncate("file2.txt", 10);

}