```c
/* Minicat_basic.c
 *  Created on: Sep 15, 2017
 *  Author: scott Jin
 */
#include <errno.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#define EXIT_FAIL -1
void Read_Write(int fd_in,int fd_out,long buffersize, char* buffer, char* argu,
    char* write_to_file){   // all used parameters for error and syscalls
    int writeout=0,readin=read (fd_in, buffer, buffersize);
    while (readin>0){
        writeout=write(fd_out, buffer, readin);
        if(writeout<0) {                                                    //
            writing error
            fprintf(stderr, "ERROR-->WRITE_FAILURE: Could not write to file
                (%s): %s\n", write_to_file, strerror(errno));
            exit(EXIT_FAIL);
        }
        else if (readin!=writeout){
            readin-=writeout;
            buffer+=writeout;
            write(fd_out, buffer, readin);                                  //
                try again with buffer offset
            if(readin!=writeout){
                    //partial writing error
               fprintf(stderr, "ERROR-->WRITE_FAILURE: Could not write to file
                   (%s): %s\n", write_to_file, strerror(errno));
               exit(EXIT_FAIL);
            }
        }
        readin=read (fd_in, buffer, buffersize);
    }
    if (readin<0) {
        fprintf(stderr,"ERROR-->READ_FAILURE: Could not read from file (%s):
            %s\n",argu, strerror(errno));  //reading error
    }
}
int main (int argc, char ** argv){
    int bflags=0, oflags=0,fd_in=-2,fd_out=STDOUT_FILENO,opt=0;       //
        initialization
    char* write_to_file=0;
    long buffersize=512;
    while ((opt = getopt(argc, argv, "b:o:")) != -1) {
        switch (opt) {
        case 'b':
            if(!strcmp ("-b", optarg)|| bflags++>0){
                    fprintf(stderr,"ERROR->Invalid argument: only one '-b' flag
                        can be accepted.\n");
                    exit(EXIT_FAIL);
            }else if(!strcmp ("-", optarg) || !strcmp ("-o", optarg)){
                    fprintf(stderr, "ERROR->Void argument: no argument
                        provided!\n");
```

```
                exit(EXIT_FAIL);
            }else if((buffersize= strtol(optarg,NULL,0)) < 1){
                fprintf(stderr, "ERROR->Invalid argument: buffer size must be
                    greater than 0\n");
                exit(EXIT_FAIL);
            }break;
        case 'o':
            if( !strcmp ("-o", optarg) || oflags++>0){
                    fprintf(stderr,"ERROR->Invalid argument: only one '-O' flag
                        can be accepted.\n");
                        exit(EXIT_FAIL);
            }else if (!strcmp ("-", optarg)||(!strcmp ("-b", optarg))){
                    fprintf(stderr,"ERROR->Void argument: no/invalid argument
                        provided!\n");
                    exit(EXIT_FAIL);
            }else{
                    fd_out = open(optarg, O_RDWR|O_CREAT|O_TRUNC, 0666);
                    write_to_file=optarg;
                    if(fd_out<0){
                        fprintf(stderr, "ERROR->Failed to open file (%s) for
                            writing: %s\n", optarg, strerror(errno));
                        exit(EXIT_FAIL);
                    }
            }break;
        case '?':
            fprintf(stderr,"ERROR-->Invalid option(or missing argument):
                %c\nUsage: Minicat [-b ###] [-o outfile] [infile1 [ infile2
                [ ... ]]]\n",optopt);
            exit(EXIT_FAIL);
            break;
        default:
            fprintf(stderr,"ERROR-->Incorrect format:%s\nUsage: Minicat [-b
                ###] [-o outfile] [infile1 [ infile2 [ ... ]]]\n",argv[0]);
            exit(EXIT_FAIL);
        }
    }
    char *buffer=malloc(buffersize);
    if (write_to_file==0) write_to_file ="STDOUT_FILENO";//in case writing
        error
    for (; optind < argc; ++optind) {                   //loop through the
        option argument
        //fprintf(stdout,"Processing Argument:%s\n",argv[optind]);
        if(!strcmp ("-", argv[optind])) {
            fd_in = STDIN_FILENO;
            argv[optind]="STDIN_FILENO";
        }else if((fd_in = open(argv[optind], O_RDONLY)) < 0) {
            if(!strcmp ("-b", argv[optind])||!strcmp ("-o", argv[optind])){
                fprintf(stderr, "ERROR->Possibly Wrong  place of option (%s):
                    %s\nUsage: Minicat [-b ###] [-o outfile] [infile1 [ infile2
                    [ ... ]]]\n:", argv[optind], strerror(errno));
            }else{
                fprintf(stderr, "ERROR->Failed to open file (%s) for reading:
                    %s\n", argv[optind], strerror(errno));
            }
                exit(EXIT_FAIL);
        }
```

```
    Read_Write(fd_in,fd_out,buffersize,buffer,argv[optind],write_to_file);
    if(fd_in!=STDIN_FILENO){
        if((close(fd_in))<0) {
            fprintf(stderr, "ERROR->Failed to close Inputfile (%s): %s\n",
                argv[optind], strerror(errno));
            exit(EXIT_FAIL);
        }
    }
}
if(fd_in==-2){
    fd_in = STDIN_FILENO;
    Read_Write(fd_in,fd_out,buffersize,buffer,"STDIN_FILENO",write_to_file)
        ;
}
if(fd_out!=STDOUT_FILENO){
            if ((close (fd_out))<0){
                fprintf(stderr, "ERROR->Failed to close Outputfile (%s):
                    %s\n", write_to_file, strerror(errno));
                exit(EXIT_FAIL);
        }
}
free(buffer);
return(EXIT_SUCCESS);
}
```