Burak Korkmaz
21601296

OPERATING SYSTEMS PROJECT 1

In the first part of the project I created called **bilshell**, which is a Linux shell that run commands. It has two modes which are interactive and batch mode. In batch mode it invokes with a text file name.

"*./bilshell N inputfilename*"

In the interactive mode it invokes as normal shell.

"bilshell-$: "

In the second part of the project, I used pipes to execute interprocess communication among two commands. I used two childs to manage pipes. Read() and write() system calls are done in parent.

time ./bilshell 4096 infile.txt -> invoke of the batch mode

time ./bilshell 4096 -> invoke of interactive mode

For the last part, I made experiments to calculate the programs that I produce produce.c and consumer.c . For the result of the experiment., I observed that the N (number of read at one time of pipe) effects the time. As N increases, the time decreases. So, N number increases the of the performance of the compound command execution. As I mentioned on the first homework, time command is used for the measure the time in program execution. It has three different times as real, user, and sys. real is the clock time. It calculates time from stars to end of a call. user gives the CPU time that spent outside the kernel in an execution. sys gives the CPU time that spent inside the kernel in an execution.

M = 10 , N=1

```
burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1 infile.txt
./producer 10 | ./consumer 10
character-count: 10
read-call-count: 10


real    0m0,181s
user    0m0,002s
sys     0m0,000s
```

M = 10 , N=10

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 10 infile.txt
./producer 10 | ./consumer 10
character-count: 10
read-call-count: 1

real 0m0,199s
user 0m0,002s
sys 0m0,002s

M = 100 , N=1

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1 infile.txt
./producer 100 | ./consumer 100
character-count: 100
read-call-count: 100

real 0m0,179s
user 0m0,002s
sys 0m0,000s

M = 100 , N=10

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 10 infile.txt
./producer 100 | ./consumer 100
character-count: 100
read-call-count: 10

real 0m0,200s
user 0m0,007s
sys 0m0,001s

M = 100 , N=100

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 100 infile.txt
./producer 100 | ./consumer 100
character-count: 100
read-call-count: 1

real 0m0,178s
user 0m0,001s
sys 0m0,001s

M = 1000 , N=1

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1 infile.txt
./producer 1000 | ./consumer 1000
character-count: 1000
read-call-count: 1000

real 0m0,194s
user 0m0,009s
sys 0m0,000s

M = 1000 , N=10

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 10 infile.txt
./producer 1000 | ./consumer 1000
character-count: 1000
read-call-count: 100

real 0m0,201s
user 0m0,002s
sys 0m0,006s

M = 1000 , N=100

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 100 infile.txt
./producer 1000 | ./consumer 1000
character-count: 1000
read-call-count: 10

real 0m0,194s
user 0m0,005s
sys 0m0,004s

M = 1000 , N=1000

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1000 infile.txt
./producer 1000 | ./consumer 1000
character-count: 1000
read-call-count: 1

real 0m0,177s
user 0m0,003s
sys 0m0,001s

M = 10000 , N=1

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1 infile.txt
./producer 10000 | ./consumer 10000
character-count: 10000
read-call-count: 10000

real 0m0,191s
user 0m0,005s
sys 0m0,012s

M = 10000 , N=10

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 10 infile.txt
./producer 10000 | ./consumer 10000
character-count: 10000
read-call-count: 1000

real 0m0,195s
user 0m0,001s
sys 0m0,012s

M = 10000 , N=100

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 100 infile.txt
./producer 10000 | ./consumer 10000
character-count: 10000
read-call-count: 100

real 0m0,211s
user 0m0,002s
sys 0m0,005s

M = 10000 , N=1000

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1000 infile.txt
./producer 10000 | ./consumer 10000
character-count: 10000
read-call-count: 10

real 0m0,182s
user 0m0,007s
sys 0m0,000s

M = 10000 , N=10000

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 10000 infile.txt
./producer 10000 | ./consumer 10000
character-count: 10000
read-call-count: 1

real 0m0,194s
user 0m0,003s
sys 0m0,005s

M = 100000 , N=1

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1 infile.txt
./producer 100000 | ./consumer 100000
character-count: 100000
read-call-count: 100000

real 0m0,411s
user 0m0,045s
sys 0m0,277s

M = 100000 , N=10

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 10 infile.txt
./producer 100000 | ./consumer 100000
character-count: 100010
read-call-count: 10001

real 0m0,145s
user 0m0,018s
sys 0m0,042s

M = 100000 , N=100

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 100 infile.txt
./producer 100000 | ./consumer 100000
character-count: 100100
read-call-count: 1001

real 0m0,144s
user 0m0,036s
sys 0m0,022s

M = 100000 , N=1000

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1000 infile.txt
./producer 100000 | ./consumer 100000
character-count: 114000
read-call-count: 114

real 0m0,121s
user 0m0,008s
sys 0m0,045s

M = 100000 , N=10000

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 10000 infile.txt
./producer 100000 | ./consumer 100000
character-count: 150000
read-call-count: 15

real 0m0,114s
user 0m0,017s
sys 0m0,037s

M = 100000 , N=100000

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 100000 infile.txt
./producer 100000 | ./consumer 100000

real 0m0,104s
user 0m0,001s
sys 0m0,001s

M = 1000000 , N=1

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1 infile.txt
./producer 1000000 | ./consumer 1000000
character-count: 1000000
read-call-count: 1000000

real 0m3,086s
user 0m0,360s
sys 0m2,458s

M = 1000000 , N=10

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 10 infile.txt
./producer 1000000 | ./consumer 1000000
character-count: 1000010
read-call-count: 100001

real 0m0,697s
user 0m0,159s
sys 0m0,376s

M = 1000000 , N=100

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 100 infile.txt
./producer 1000000 | ./consumer 1000000
character-count: 1000100
read-call-count: 10001

real 0m0,576s
user 0m0,143s
sys 0m0,302s

M = 1000000 , N=1000

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1000 infile.txt
./producer 1000000 | ./consumer 1000000
character-count: 1012000
read-call-count: 1012

real 0m0,604s
user 0m0,130s
sys 0m0,322s

M = 1000000 , N=10000

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 10000 infile.txt
./producer 1000000 | ./consumer 1000000
character-count: 1050000
read-call-count: 105

real 0m0,550s
user 0m0,131s
sys 0m0,306s

M = 1000000 , N=100000

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 100000 infile.txt
./producer 1000000 | ./consumer 1000000

real 0m0,492s
user 0m0,000s
sys 0m0,002s

M = 1000000 , N=1000000

burak@burak-ubuntu:~/Desktop/Project 1$ time ./bilshell 1000000 infile.txt
./producer 1000000 | ./consumer 1000000

real 0m0,524s
user 0m0,003s
sys 0m0,000s

producer.c

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <readline/readline.h>
#include <readline/history.h>

int main(int n, char* args[]){
        int count = atoi(args[1]);
        char alphanumeric[] ="1234567890QWERTYUIOPASDFGHJKLZXCVBNM";
        for(int i = 0 ; i< count ; i++){
            int random = rand() % 37;
            char forwrite[1];
            forwrite[0]=alphanumeric[random];
            printf("%c",forwrite[0]);
        }
        return 0;
}
```

producer.c

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <readline/readline.h>
#include <readline/history.h>

int main(int n, char* args[]){
   int count = atoi(args[1]);
   char arraychar[1];
   for(int i = 0 ; i <count ; i++){
       read(0,arraychar,1);
   }
   return 0;
}
```

bilshell.c

```c
/*
  Burak Korkmaz
 */

#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <readline/readline.h>
#include <readline/history.h>
#include <time.h>

int  bilshell(char* str){
  char* temp;
  temp = readline("bilshell-$: ");
  if(strlen(temp) != 0){
    add_history(temp);
    strcpy(str,temp);
    return 0;
  }
  else
    return 1;
}

void execute(char** comlist){

  pid_t pid1;
```

```
  pid1 = fork();
  if(pid1 < 0){
    printf("Could not create pipe 1\n");
    exit(1); ;
  }
  else if (pid1 == 0){ /* Child */
    if(execvp(comlist[0],comlist) ==-1)
        printf("command is not executed\n");

  }
  else{ /* Parent */
    wait(NULL);
  }
  return ;
}

void pipeexec(char** comlist, int place,int count){

    char arraychar[count];
    char* pipecommand1[1000];
    char* pipecommand2[1000];

    //printf("entered pipe - 2\n");

    for(int i = 0 ; i < 1000;i++){
        pipecommand1[i]=NULL;
        pipecommand2[i]=NULL;
    }

    for(int i = 0;i < place; i++)
        pipecommand1[i] = comlist[i];
    for(int i = place + 1; comlist[i] != NULL ;i++)
        pipecommand2[i-(place+1)]=comlist[i];

    //printf("entered pipe - 3\n");

    /*for(int i = 0; pipecommand1[i] != NULL ;i++)
        printf("%s ",pipecommand1[i]);
    printf("\n");

    for(int i = 0; pipecommand2[i] != NULL ;i++)
        printf("%s ",pipecommand2[i]);
    printf("\n");
    */



    int charactercount;
    int readcount,charcount;
    int pipe1[2],pipe2[2];
    pid_t pid1, pid2;

    if(pipe(pipe1) < 0){
        printf("Could not create pipe 1\n");
```

```
    exit(1);
}

if(pipe(pipe2) < 0 ){
    printf("Could not create pipe 2\n");
    exit(1);
}

pid1 = fork();

if(pid1 == -1)  {
    printf("Could not create child -fork1");
    exit(1);
}

else if(pid1 == 0){

    close(pipe1[0]);
    close(pipe2[0]);
    close(pipe2[1]);
    dup2(pipe1[1],1);
    close(pipe1[1]);

    if(execvp(pipecommand1[0],pipecommand1) == -1){
        printf("command is not executed\n");
    }

}
else{/*parent*/
    //wait(NULL);
    pid2 = fork();

    if(pid2==-1){
        printf("Could not create child-fork 2");
        exit(1);
    }

    else if(pid2 == 0){

        close(pipe1[0]);
        close(pipe1[1]);
        close(pipe2[1]);
        dup2(pipe2[0],0);
        close(pipe2[0]);

        if(execvp(pipecommand2[0],pipecommand2) == -1){
            printf("command is not executed\n");
        }

    }
    else{/* parent */

        close(pipe1[1]);
        close(pipe2[0]);
```

```
            while((charactercount = (read(pipe1[0],arraychar,count))>0))
{
                charcount += write(pipe2[1],arraychar,count);
                readcount += charactercount;
            }

            close(pipe1[0]);
            close(pipe2[1]);

            wait(NULL);

        }
    }

    printf("character-count: %i\n",charcount);
    printf("read-call-count: %i\n",readcount);

}

void parser(char* str,char** comlist,int* place){

  char* token;
  int i = 0;

  for(int i = 0 ; i < 1000;i++){
        comlist[i]=NULL;
    }
    *place = 0;

  token = strtok(str,"\n " );


  while(token!=NULL){


    comlist[i]= token;
    if(strcmp(comlist[i],"|")==0){
        *place = i;
        //printf("%i\n",*place);
    }
    i++;
    token = strtok(NULL,"\n ");
  }


}

void batch(char** comlist,int count){



    FILE *fl;
    size_t length = 0;
    char* inputstr ;
    fl = fopen(comlist[2], "r");
```

```
    int place = 0;


  while(getline(&inputstr,&length,fl) != -1){
    parser(inputstr,comlist,&place);
    for(int j = 0 ; comlist[j] != NULL ; j++)
        printf("%s ",comlist[j]);
    printf("\n");
    if(place == 0)
       execute(comlist);
   else{
       //printf("place %i - before pipe\n", count);
       pipeexec(comlist,place,count);
   }
  }

  fclose(fl);

}



int main(int n, char* filename[]){
    char inputstr[1000];
    char*  comlist[1000];
    int place;
    //printf("%i\n",n);
    //printf("%s\n",filename[0]);

  while(1){

      int n = atoi(filename[1]);

    if(filename[2] == NULL)  {
        if(bilshell(inputstr))
            continue;
        parser(inputstr,comlist,&place);
    }

    /*if(place != 0)
        exit(0);*/

    if(filename[2] !=NULL){
        //printf("1st place - %i\n",i);
        batch(filename,n);
        exit(1);
        //printf("finished\n");
    }
    else if(strcmp(comlist[0],"exit")==0){
        printf("Bye\n");
        exit(0) ;
    }
    else if(strcmp(comlist[0],"cd")==0)
        chdir(comlist[1]);
    if(place!=0)
```

```
            pipeexec(comlist,place,n);
        else
            execute(comlist);

    }
}
```