OS Lab Assignment 4

## Name: Keshav Garg

## ID: 2018UCP1674

GitHub Repo link: [**Click**](https://github.com/gargk747/OS-Lab/tree/master/Assignment%204)

**PIPE:**

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/wait.h>

#include<string.h>

int main()

{

int i;

char buff[100],buff1[100];

char filename[] = "input.dat";

int p1[2],p2[2],p3[2];

FILE \*file;

file = fopen(filename,"r");

if (pipe(p1) < 0 || pipe(p2) < 0 || pipe(p3) < 0)

{

perror("\n Can't open pipe\n");

exit(-1);

}

int f1 = fork();

if(f1 < 0)

{

printf("\n Error creating fork.\n");

exit(-1);

}

else if(f1 == 0)

{

// process 2

read(p1[0], buff, sizeof(buff));

// manage spaces

int j;

j=0;

i=0;

// trim starting spaces

while(buff[i]==' ') i++;

for(;i<sizeof(buff)/sizeof(buff[0]);i++)

{

// multiple spaces

if(buff[i]==' ')

{

if(j>0 && buff1[j-1]!=' ')

buff1[j++]=' ';

}

// punctuation marks

else if(buff[i]==',' || buff[i]=='.' || buff[i]=='?' || buff[i]=='!')

{

if(j>0 && buff1[j-1]==' ')

{

buff1[j-1]=buff[i];

buff1[j++]=' ';

}

else

{

buff1[j++]=buff[i];

}

}

// characters

else

{

buff1[j]=buff[i];

j++;

}

}

printf("\n Process 2: %s\n", buff1);

write(p2[1], buff1, sizeof(buff1));

}

else if(f1 > 0)

{

int f2 = fork();

if(f2 < 0)

{

printf("\n Error creating fork.\n");

exit(-1);

}

else if(f2 == 0)

{

// process 3

read(p2[0], buff, sizeof(buff));

for(i = 0;i<sizeof(buff)/sizeof(buff[0]);i++)

{

if(i==0)

{

if(buff[i]>='a' && buff[i]<='z')

{

buff[i]=buff[i]-'a'+'A';

}

}

else

{

if(buff[i]>='A' && buff[i]<='Z')

{

buff[i]=buff[i]-'A'+'a';

}

}

}

printf("\n Process 3: %s\n",buff);

int count = 0;

for(i=0;i<sizeof(buff)/sizeof(buff[0]);i++)

{

if(buff[i]==' ')

count++;

}

count++;

write(p3[1], &count, sizeof(count));

}

else

{

// process 1

while (fgets(buff, sizeof(buff), file) != NULL) {

printf("\n Process 1: %s\n", buff);

write(p1[1], buff, sizeof(buff));

}

int c;

read(p3[0],&c,sizeof(c));

printf("\n Process 1: Word count- %d\n",c);

}

}

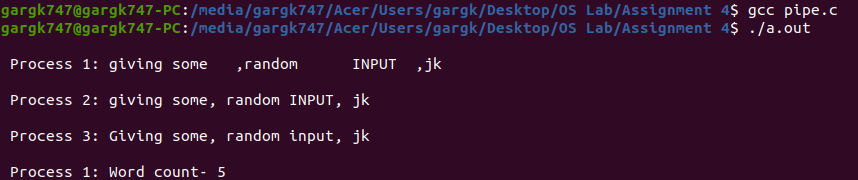
return 0;

}

Input.dat:

giving some ,random INPUT ,jk

Output:



**SHELL:**

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/wait.h>

#include<string.h>

int main()

{

char s[100];

printf("Shell started.\n");

printf(">> ");

//scanf("%[^\n]s",s);

fgets(s,100,stdin);

int f;

while(1)

{

//exit command

if(strcmp(s,"exit\n")==0)

{

printf("Shell terminated.\n");

break;

}

else

{

//creating fork

f = fork();

if(f<0)

{

printf("Fork can not be created.\n");

break;

}

else if(f==0)

{

s[strlen(s)-1]=s[strlen(s)];

int i=0;

for(;i<strlen(s);i++)

{

if(s[i]==' ')

{

break;

}

}

char s1[100];

char s2[100];

memcpy(s1, &s[0], i );

s1[i] = '\0';

s2[0]=0;

i++;

if(i<strlen(s))

{

memcpy(s2, &s[i],strlen(s)-i);

s2[strlen(s)-i]='\0';

}

char \*myargs[3];

myargs[0] = strdup(s1);

if(s2[0]!=0)

myargs[1] = strdup(s2);

else

myargs[1] = NULL;

myargs[2] = NULL;

execvp(myargs[0], myargs);

exit(0);

}

else if(f>0)

{

wait(NULL);

printf(">> ");

fgets(s,100,stdin);

}

}

}

return 0;

}

Output:

