/\*

EXECV reverse order

Problem Statement: Implement the C program in which main program accepts an array. Main program uses the FORK

system call to create a new process called a child process. Parent process sorts an array and passes the

sorted array to child process through the command line arguments of EXECVE system call. The child

process uses EXECVE system callto load new program which display array in reverse order.

\*/

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <stdlib.h>

#include <sys/wait.h>

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

int status;

int arr[5];

char char\_arr[5];

printf("Enter 5 numbers to sort: ");

for(int i=0; i<5; i++) {

scanf("%d", &arr[i]);

}

for (int i=0; i<5; i++) {

char\_arr[i] = arr[i];

}

int pid = fork();

char \*newargv[20];

newargv[0] = argv[1];

newargv[1] = char\_arr;

newargv[2] = NULL;

if(pid == 0) {

printf("Child process\n");

execve(argv[1], newargv, NULL);

}

else if(pid != 0) {

wait(&status);

printf("\nParent process\n");

}

}

/\* Sub program\*/

#include <stdio.h>

void bubble\_sort(char arr[]) {

for(int i=0; i<5; i++) {

for(int j=i+1; j<5; j++) {

if(arr[i] > arr[j]) {

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

printf("Sorted array: ");

for(int i=0; i<5; i++) {

printf("%d ", arr[i]);

}

}

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

bubble\_sort(argv[1]);

}