OPERATING SYSTEM

UNIT-1 BANANA PROBLEM

NAME: Keerthan P.V

SRN: PES2UG23CS272

SECTION: 4E

Q2. Create a global array with values [1, 6, 2, 4, 5, 8, 9, 0]. Sort the same within the child process, and display the values in the parent process. Explain, are the displayed values in the sorted order? If not, why?

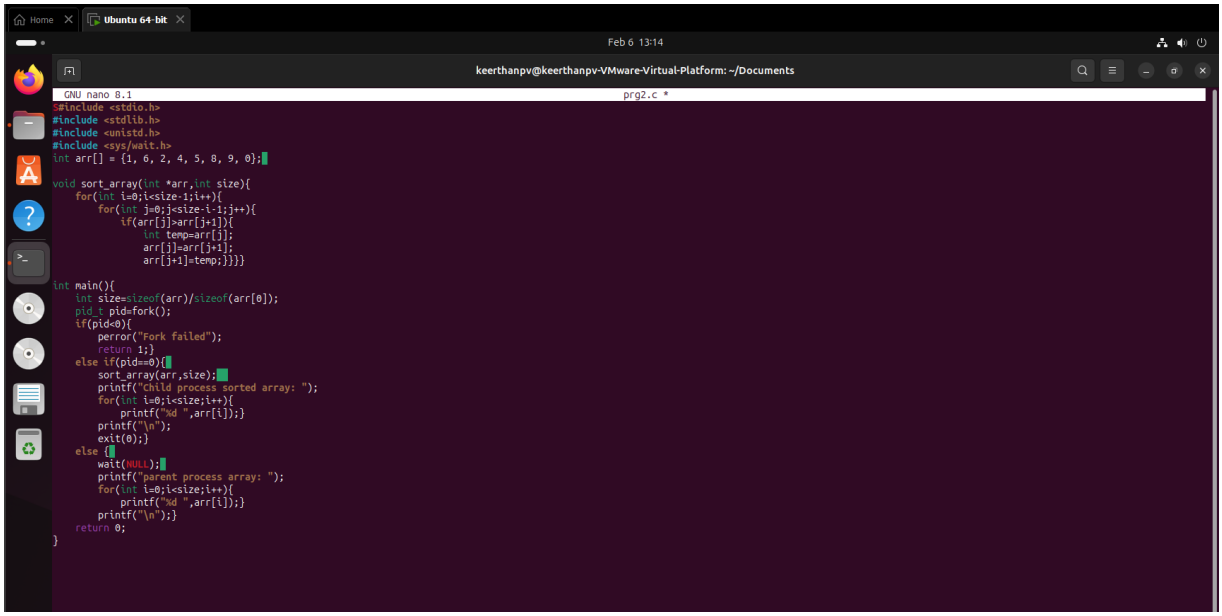
```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>
int arr[] = {1, 6, 2, 4, 5, 8, 9, 0};


void sort_array(int *arr,int size){
   for(int i=0;i<size-1;i++){
      for(int j=0;j<size-i-1;j++){
         if(arr[j]>arr[j+1]){
            int temp=arr[j];
```

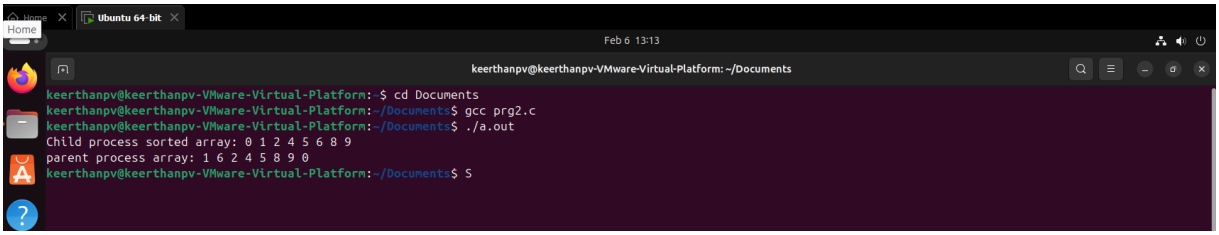
```
            arr[j]=arr[j+1];
            arr[j+1]=temp;}}}}

int main(){
    int size=sizeof(arr)/sizeof(arr[0]);
    pid_t pid=fork();
    if(pid<0){
        perror("Fork failed");
        return 1;}
    else if(pid==0){
        sort_array(arr,size);
        printf("Child process sorted array: ");
        for(int i=0;i<size;i++){
            printf("%d ",arr[i]);}
        printf("\n");
        exit(0);}
    else {
        wait(NULL);
        printf("parent process array: ");
        for(int i=0;i<size;i++){
            printf("%d ",arr[i]);}
```

```
        printf("\n");}
    return 0;
}
```



OUTPUT:



When a child process is created using fork(), it receives a separate copy of the parent's memory space. As a result, any modifications made in the child process do not affect the parent process. In this case, the array is sorted within the child process, but since the parent and child have independent memory copies, the changes remain confined to the child process. Therefore, when the parent process prints the array, it displays the original unsorted version. However, if the array were printed within the child process, it would correctly show the sorted version.