**ASSIGNMENT NO. 2**

**ORPHAN PROCESS**

#include<unistd.h>

#include<stdio.h>

#include<sys/types.h>

#include<sys/wait.h>

void quicksort(int a[20],int lb,int ub, int n)

{

int down,up,pivot,temp,k;

pivot=a[lb];

down=lb;

up=ub;

if(lb<ub)

{

while(down<up)

{

while(a[down]<=pivot)

down++;

while(a[up]>pivot)

up--;

if(down<up)

{

temp=a[down];

a[down]=a[up];

a[up]=temp;

}

}//end of outer while

a[lb]=a[up];

a[up]=pivot;

quicksort(a,lb,up-1,n);

quicksort(a,up+1,ub,n);

}//end of outer if

}//end of quicksort()

void merge(int a[],int i1,int j1,int i2,int j2)

{

int temp[50]; //array used for merging

int i,j,k;

i=i1; //beginning of the first list

j=i2; //beginning of the second list

k=0;

while(i<=j1 && j<=j2) //while elements in both lists

{

if(a[i]<a[j])

temp[k++]=a[i++];

else

temp[k++]=a[j++];

}

while(i<=j1) //copy remaining elements of the first list

temp[k++]=a[i++];

while(j<=j2) //copy remaining elements of the second list

temp[k++]=a[j++];

//Transfer elements from temp[] back to a[]

for(i=i1,j=0;i<=j2;i++,j++)

a[i]=temp[j];

}

void mergesort(int a[],int i,int j)

{

int mid;

if(i<j)

{

mid=(i+j)/2;

mergesort(a,i,mid); //left recursion

mergesort(a,mid+1,j); //right recursion

merge(a,i,mid,mid+1,j); //merging of two sorted sub-arrays

}

}

int main()

{

pid\_t pid;

int a[20],n,i, cpid, status;

printf("\nEnter number of terms in array(less than 20): ");

scanf("%d",&n);

printf("\nEnter elements:\n");

for(i=0;i<n;i++)

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

pid=fork();

if(pid<0)

{

printf("Error in fork");

}

else

{

if(pid==0)

{

printf("In Child process with process id: %d \n", getpid());

printf("It's Parent is: %d\n", getppid());

sleep(10);

mergesort(a,0,n-1);

printf("\nArray after Merge Sorting :");

for(i=0;i<n;i++)

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

printf("\n\n");

printf("Child process with process id: %d \n", getpid());

printf("Child Process Completed\n");

}

else

{

printf("\nIn Parent process with process id: %d \n", getpid());

cpid=wait(&status);//for normal termination

printf("\nQuicksort:");

quicksort(a,0,n-1,n);

printf("\nArray after Quick sorting: ");

for(i=0;i<n;i++)

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

if(WIFEXITED(status))

printf("\nNormal Termination, Exit Status: %d\n", WEXITSTATUS(status));//exit status=0 for normal compilation,other +Ve no for abnormal compilation.

else if(WIFSIGNALED(status))

printf("\nAbnormal Termination, Signal Number = %d\n", WTERMSIG(status));

printf("\nParent Process Completed!\n");

}

}

return 0;

}

**OUTPUT**

[it@localhost ~]$ gcc demoOrphan.c

[it@localhost ~]$ ./a.out

Enter number of terms in array(less than 20): 5

Enter elements:

4

2

8

5

7

In Child process with process id: 2360

It's Parent is: 2359

In Parent process with process id: 2359

Array after Merge Sorting :2 4 5 7 8

Child process with process id: 2360

Child Process Completed

Quicksort:

Array after Quick sorting: 2 4 5 7 8

Normal Termination, Exit Status: 0

Parent Process Completed!

[it@localhost ~]$