//FORK

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/wait.h>
#include<sys/types.h>
 int main(){  pid_t t; int arr[10]; int n;
printf("Enter the number of elements: ");
scanf("%d",&n);
printf("Enter the numbers: ");
 for(int i=0;i<n;i++){
scanf("%d",&arr[i]);
}  t = fork(); if(t==-1){ printf("Failed to
Create Child Process...");
} else if(t==0){     //sleep(3);     printf("I am Child
having id: %d My Parent has id:
%d\n",getpid(),getppid());
         for(int i=0;i<n;i++){
for(int j=0;j<n-i-1;j++){
if(arr[j]>arr[j+1]){
int temp = arr[j];
arr[j] = arr[j+1];
arr[j+1] = temp;
            }
        }
}
    printf("Array is sorted by Child\n");
for(int i=0;i<n;i++){     printf("%d
",arr[i]);
    }


         } else{
//sleep(2);
wait(NULL);//
wait call use to
wait for child
process of
process which
execute wait
printf("\nI am
parent having
id: %d My child
has id:
%d\n",getpid(),t
);


    printf("\n");
}
//sleep(3);
 return 0;
}
```

//EXECVE

//There are two codes, P.c and C.c .You have to first complie both the code and and save C.c executable file as C. and then run the P.c file

//P.c

```
#include <stdio.h>
#include <unistd.h>
#include<sys/types.h>
#include<stdlib.h>
 int main(){

printf("I am Main having id: %d \n",getpid());
 pid_t t;  int
n; int
arr[100];

printf("Enter the number of elements: ");
scanf("%d",&n); printf("Enter the
elements: "); arr[0]=n; for(int
i=1;i<=n;i++){ scanf("%d",&arr[i]);
} n++;//important.............
t = fork();
FILE *f;  if(t==0){ //Child Process
sleep(0.5); printf("CHILD\n");
const char *args[]={"./C",NULL};
execv(args[0],(char* const*)args);
     }
else{
         //Parent Process
for(int i=1;i<=n;i++){
for(int j=1;j<=n-i-1;j++){
if(arr[j]>arr[j+1]){
int temp = arr[j];
arr[j] = arr[j+1];
arr[j+1] = temp;
            }
        }
    }

    //     for (int j = 1; j <= n - i - 1; j++) {
        //        if (arr[j] > arr[j + 1]) {
        //            int temp = arr[j];
        //            arr[j] = arr[j + 1];
        //            arr[j + 1] = temp;
        //        }
        //    }
        //}
printf("Sorted by Parent\n");
for(int i=1;i<n;i++){
printf("%d ",arr[i]);
}printf("\n\n");     f =
fopen("lp1.txt","w");     for (int
i=0;i<n;i++){
fprintf(f,"%d ",arr[i]);
    }
fclose(f);

}




//for(int i=0;i<=n;i++){
//printf("%d ",arr[i]);
//}

//f = fopen("lp1.txt","r");
//for (int i=0;i<=n;i++){
//fscanf(f,"%d",&arr[i]);
//}
//for(int i=0;i<=n;i++){
//printf("%d ",arr[i]);
//}

    //const char *args[] = {"./rev1",data,NULL};
    //execv(args[0],(char* const*)args);
//}
//else{
//char *d="y";
//int a=21;
//const char *args[] = {"./tp",d,NULL};
//execv(args[0],(char* const*)args);

//}


//printf("LAST OF MAIN");
return 0; }
```

//C.c

```
#include <stdio.h>
#include <unistd.h>
#include<sys/types.h>
#include<stdlib.h>  int main(int
argc,char *argv[]){
 int arr[200];  int n;
printf("IN NEW PROCESS...\n");
//printf("HELOO"); FILE
*f; f =
fopen("lp1.txt","r");
fscanf(f,"%d",&n); for
(int i=0;i<n;i++){
fscanf(f,"%d",&arr[i]);
}  printf("Printed in revrse
order\n\n"); for(int i=n-1;i>=0;i--){
printf("%d ",arr[i]);
}

return 0; }
```