MAIN (PARENT) PROGRAM :-

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

void bubbleSort(int arr[], int n)
{
    for (int i = 0; i < n - 1; i++)
    {
        for (int j = 0; j < n - i - 1; j++)
        {
            if (arr[j] > arr[j + 1])
            {
                // Swap arr[j] and arr[j + 1]
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

int main()
{
    int n;
    printf("Enter the number of elements in the array: ");
    scanf("%d", &n);
```

```
int arr[n];
printf("Enter %d elements:\n", n);
for (int i = 0; i < n; i++)
{
    scanf("%d", &arr[i]);
}

pid_t pid = fork();

if (pid == -1)
{
    perror("Fork failed");
    return 1;
}

if (pid == 0)
{
    // Child process
    char *args[n + 2];
    args[0] = "display_reverse";
    for (int i = 0; i < n; i++)
    {
        args[i + 1] = malloc(15); // Allocate memory for each argument
        snprintf(args[i + 1], 15, "%d", arr[i]);
    }
    args[n + 1] = NULL;

    execve("./display_reverse", args, NULL);
    perror("Execve failed");
```

```
        return 1;
    }
    else
    {
        // Parent process
        wait(NULL); // Wait for the child process to finish

        // Sort the array using bubble sort
        bubbleSort(arr, n);

        // Pass the sorted array to the child process through command line arguments
        char *args[n + 2];
        args[0] = "display_reverse";
        for (int i = 0; i < n; i++)
        {
            args[i + 1] = malloc(15); // Allocate memory for each argument
            snprintf(args[i + 1], 15, "%d", arr[i]);
        }
        args[n + 1] = NULL;

        execve("./display_reverse", args, NULL);
        perror("Execve failed");
        return 1;
    }

    return 0;
}
```

CHILD (DISPLAY_REVERSE) PROGRAM:-

```
#include <stdio.h>

#include <stdlib.h>


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

{

   printf("Sorted array in reverse order:\n");

   for (int i = argc - 1; i >= 1; i--)

   {

      printf("%s ", argv[i]);

   }

   printf("\n");


   return 0;

}
```

OUTPUT:-

```
ubuntu@ubuntu-VJ:~$ gcc -o display_reverse display_reverse.c
ubuntu@ubuntu-VJ:~$ gcc main.c
ubuntu@ubuntu-VJ:~$ ./a.out
Enter the number of elements in the array: 7
Enter 7 elements:
2
1
5
3
4
8
16
Sorted array in reverse order:
16 8 4 3 5 1 2
Sorted array in reverse order:
16 8 5 4 3 2 1
ubuntu@ubuntu-VJ:~$
```