

OSL  
Assignment – 2B

Process Control System Calls

**Roll no:** 33245

**CODE:**

1. Parent

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

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

int main()
{
    int n, i;
    printf("Enter the number of integers you want to sort: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter %d integers:\n", n);
    for (i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
    }
    // Forking a child process
    pid_t pid = fork();
    if (pid < 0)
    {
        printf("Fork failed.\n");
        exit(1);
    }
}
```

```

else if (pid == 0)
{
    // Child process
    printf("\nArray in Sorted Order\n");
    bubbleSort(arr, n);

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

    printf("\nChild process\n");
    printf("Child Process id %d \n", getpid());
    printf("Parent Process id %d \n", getppid());

    // Converting integer array to string array for execve
    char arr_str[n][10];
    char *args[n + 2];
    args[0] = "./assignchild"; // Executable name for child process

    for (i = 0; i < n; i++)
    {
        sprintf(arr_str[i], "%d", arr[i]);
        args[i + 1] = arr_str[i];
    }

    args[n + 1] = NULL; // Mark the end of arguments
    execve(args[0], args, NULL); // Execute the child process
}
else
{ // Parent process
    printf("\nParent process\n");
    printf("Process id %d \n", getpid());
}

return 0;
}

```

## 2. Child

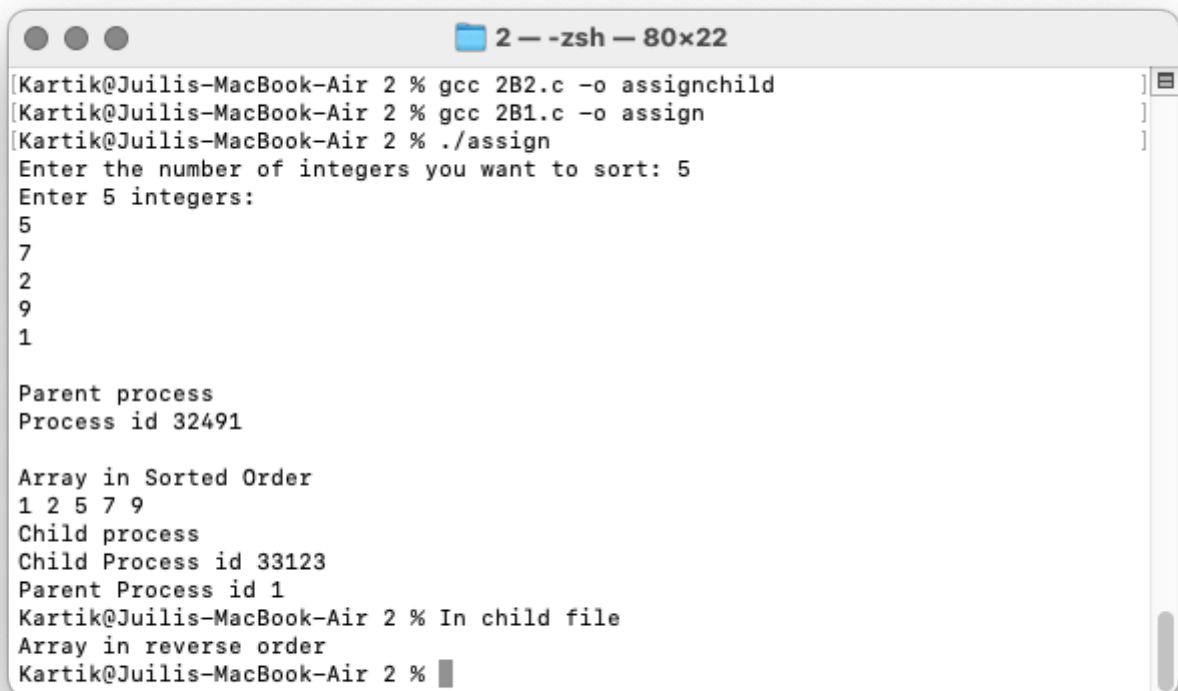
```
#include<stdio.h>
#include<unistd.h>

int main(int argc , char *argv[])
{
    printf("In child file \n");
    printf("Array in reverse order\n");

    for (int i=argc-1; i>=0; i--)
    {
        printf("%s ", argv[i]);
    }

    return 0;
}
```

## OUTPUT:



```
2 — -zsh — 80x22
[Kartik@Juilis-MacBook-Air 2 % gcc 2B2.c -o assignchild
[Kartik@Juilis-MacBook-Air 2 % gcc 2B1.c -o assign
[Kartik@Juilis-MacBook-Air 2 % ./assign
Enter the number of integers you want to sort: 5
Enter 5 integers:
5
7
2
9
1

Parent process
Process id 32491

Array in Sorted Order
1 2 5 7 9
Child process
Child Process id 33123
Parent Process id 1
Kartik@Juilis-MacBook-Air 2 % In child file
Array in reverse order
Kartik@Juilis-MacBook-Air 2 %
```

---