Assignment 2

B. Implement the C program in which main program accepts an array. Main program uses the FORK system call to create a new process called a child process. Parent process sorts an array and passes the sorted array to child process through the command line arguments of EXECVE system call. The child process uses EXECVE system callto load new program which display array in reverse order.

Binary.c

```c
#include <stdio.h>

#include <string.h>

#include <string.h>

#include <stdlib.h>

void binser(int[], int, int);

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

{

   int arr[30];

   char str1[30] = {""}, str2[30] = {""};

   int i, size, ele, j = 0, k = 0, n;

   strcpy(str1, argv[0]);

   n = strlen(str1);

   printf("%d", n);

   for (i = 0; i < strlen(str1); i++)

   {
```

```
        str2[k] = str1[i];

        k++;

        printf("\n%s removed from char array\n", str2);

        if (str1[i] == ' ')

        {

            arr[j] = atoi(str2);

            printf("\n%d putted in int array\n", arr[j]);

            strcpy(str2, "");

            k = 0;

            j++;

        }

    }

    printf("value of j is: %d\n", j);

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

    {

        printf("%d\t", arr[i]);

    }

    printf("\nEnter the elements to be search: \n");

    scanf("%d", &ele);

    binser(arr, j - 1, ele);

    return 0;

}
```

```c
void binser(int arr[30], int r, int ele)

{

   int l, mid;

   l = 0;

   mid = (l + r) / 2;

   while (l <= r)

   {

       if (arr[mid] < ele)

           l = mid + 1;

       else if (arr[mid] == ele)

       {

           printf("\nElement %d found at %d location\n", ele, mid + 1);

           break;

       }

       else

           r = mid - 1;

       mid = (l + r) / 2;

   }

   if (l > r)

       printf("\nElement not Found\n");

}
```

Bubble.c

```c
#include <stdio.h>

#include <string.h>

#include <unistd.h>

#include <stdlib.h>

#include <sys/types.h>

void bubble_sort(int, int[]);

int main()

{

   char str1[30], str2[30];

   char *a1[30] = {str2, "NULL"};

   int arr[30];

   int i, size, status;

   pid_t pid;

   do

   {

       printf("Enter total no. of elements : ");

       scanf("%d", &size);

       printf("Enter the array\n ");

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

       {

           printf("Enter %d element : ", i + 1);

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

       }
```

```
    bubble_sort(size, arr);

    printf("after sorting\n");

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

    {

        sprintf(str1, "%d", arr[i]);

        strcat(str1, " ");

        strcat(str2, str1);

    }

    printf("%s", str2);

    printf("\n");

    if ((pid = vfork()) < 0)

    {

        perror("fork");

        exit(0);

    }

    if (pid == 0)

    {



        execve("./binary", a1, NULL);

        perror("str3");

        exit(1);

    }

} while (wait(&status) != pid);

printf("\n");
```

```
   return 0;

}

void bubble_sort(int m, int arr[])

{

   int i, j, t;



   for (i = 1; i <= m - 1; i++)

       for (j = 1; j <= m - i; j++)

           if (arr[j - 1] >= arr[j])

           {

               t = arr[j - 1];

               arr[j - 1] = arr[j];

               arr[j] = t;

           }

}
```