EXPERIMENT: -6 (Creation of Multithreaded Processes using Pthread Library)

Objective: Introduce the operations on threads, which include initialization, creation, join and exit functions of thread using pthread library.

Lab Exercise Solution(s):

Q1. Write a program using pthread to concatenate the strings, where multiple strings are passed to thread function.

Ans1. **Solution: -**

Step1: - nano 6_1.c

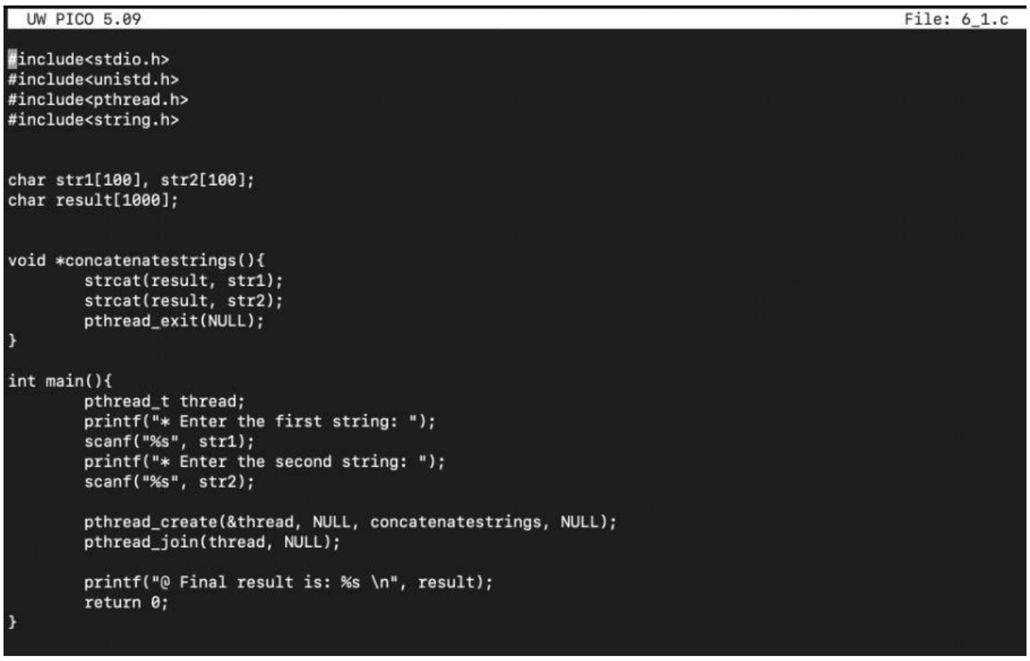
Step2: -

```
#include<stdio.h>

#include<unistd.h>

#include<pthread.h>

#include<string.h>



char str1[100], str2[100];

char result[1000];



void *concatenatestrings(){

        strcat(result, str1);

        strcat(result, str2);

        pthread_exit(NULL);

}


int main(){

        pthread_t thread;

        printf("* Enter the first string: ");

        scanf("%s", str1);

        printf("* Enter the second string: ");
        scanf("%s", str2);

        pthread_create(&thread, NULL, concatenatestrings, NULL);
        pthread_join(thread, NULL);

        printf("@ Final result is: %s \n", result);
        return 0;
}
```

Step3: - gcc 6_1.c

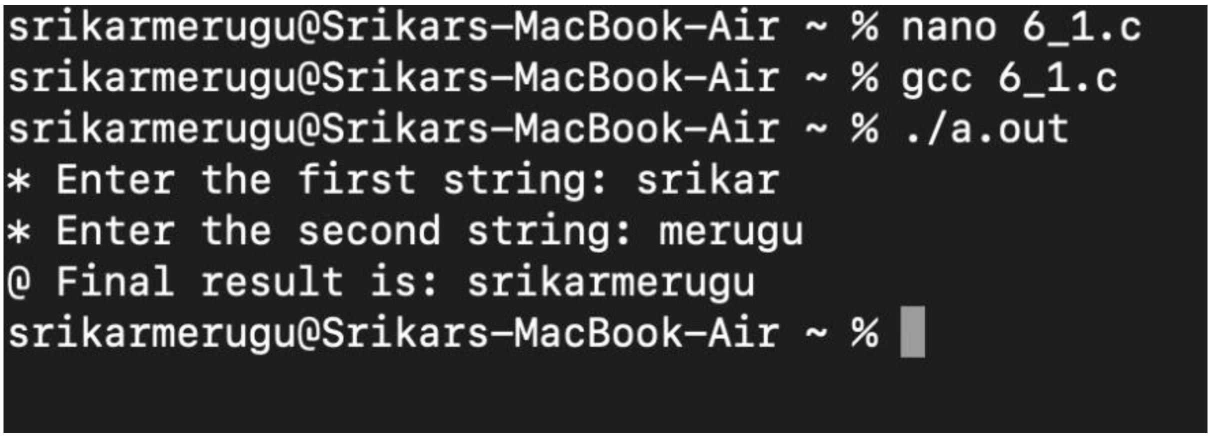
Step4: - ./a.out

nano 6_1.c



```
UW PICO 5.09                                                    File: 6_1.c

#include<stdio.h>
#include<unistd.h>
#include<pthread.h>
#include<string.h>


char str1[100], str2[100];
char result[1000];


void *concatenatestrings(){
        strcat(result, str1);
        strcat(result, str2);
        pthread_exit(NULL);
}

int main(){
        pthread_t thread;
        printf("* Enter the first string: ");
        scanf("%s", str1);
        printf("* Enter the second string: ");
        scanf("%s", str2);

        pthread_create(&thread, NULL, concatenatestrings, NULL);
        pthread_join(thread, NULL);

        printf("@ Final result is: %s \n", result);
        return 0;
}
```

Output of the program



```
srikarmerugu@Srikars-MacBook-Air ~ % nano 6_1.c
srikarmerugu@Srikars-MacBook-Air ~ % gcc 6_1.c
srikarmerugu@Srikars-MacBook-Air ~ % ./a.out
* Enter the first string: srikar
* Enter the second string: merugu
@ Final result is: srikarmerugu
srikarmerugu@Srikars-MacBook-Air ~ %
```

Q2. Write a program using pthread to find the length of string, where strings are passed to thread function.

Ans2. **Solution: -**

Step1: - nano 6_2.c

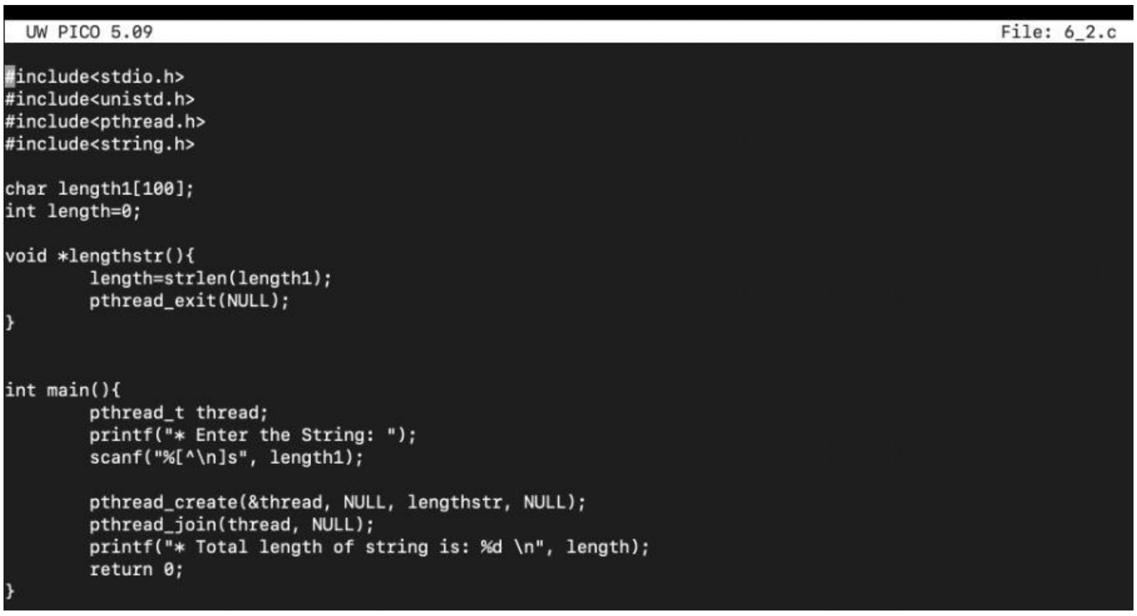
Step2: -

```
#include<stdio.h>

#include<unistd.h>

#include<pthread.h>

#include<string.h>


char length1[100];

int length=0;


void *lengthstr(){

        length=strlen(length1);

        pthread_exit(NULL);

}



int main(){

        pthread_t thread;

        printf("* Enter the String: ");

        scanf("%[^\n]s", length1);


        pthread_create(&thread, NULL, lengthstr, NULL);

        pthread_join(thread, NULL);

        printf("* Total length of string is: %d \n", length);

        return 0;

}
```

Step3: - gcc 6_2.c

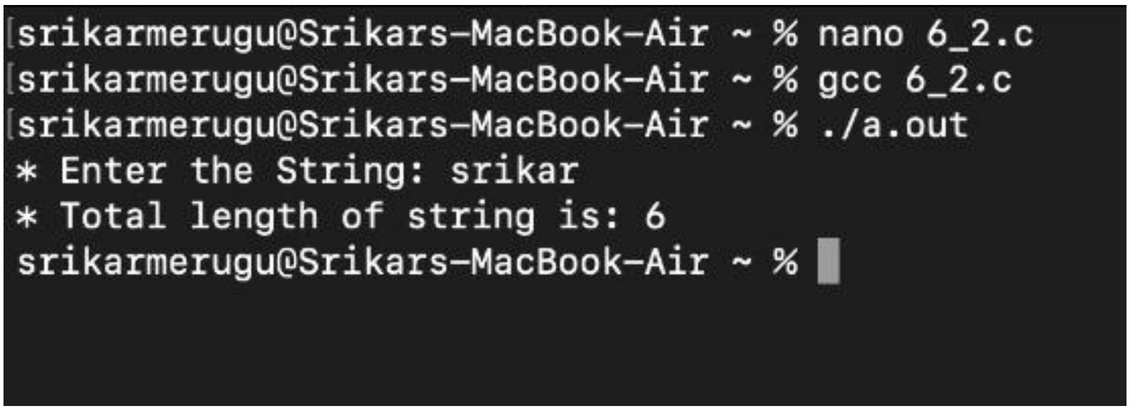
Step4: - ./a.out

nano 6_2.c



```
UW PICO 5.09                                                    File: 6_2.c

#include<stdio.h>
#include<unistd.h>
#include<pthread.h>
#include<string.h>

char length1[100];
int length=0;

void *lengthstr(){
        length=strlen(length1);
        pthread_exit(NULL);
}


int main(){
        pthread_t thread;
        printf("* Enter the String: ");
        scanf("%[^\n]s", length1);

        pthread_create(&thread, NULL, lengthstr, NULL);
        pthread_join(thread, NULL);
        printf("* Total length of string is: %d \n", length);
        return 0;
}
```

Output of the program



```
[srikarmerugu@Srikars-MacBook-Air ~ % nano 6_2.c
[srikarmerugu@Srikars-MacBook-Air ~ % gcc 6_2.c
[srikarmerugu@Srikars-MacBook-Air ~ % ./a.out
* Enter the String: srikar
* Total length of string is: 6
srikarmerugu@Srikars-MacBook-Air ~ %
```

Q3. Write a program that performs statistical operations of calculating the average, maximum and minimum for a set of numbers. Create three threads where each performs their respective operations.

Ans3. **Solution: -**

Step1: - nano 6_3.c

Step2: -

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

int arr[10] = {99, 22, 00, 88, 11, 102, 33, 66, 44, 55};

void *sort(){
        for(int i=0; i<10; i++){
                for(int j=0; j<10; j++){
                        if(arr[i] < arr[j]){
                                int temp = arr[i];
                                arr[i] = arr[j];
                                arr[j] = temp;
                        }
                }
        }
}


void *min(){
        int min = arr[0];
        printf("* Minimum element is = %d\n", min);
        pthread_exit(NULL);
```

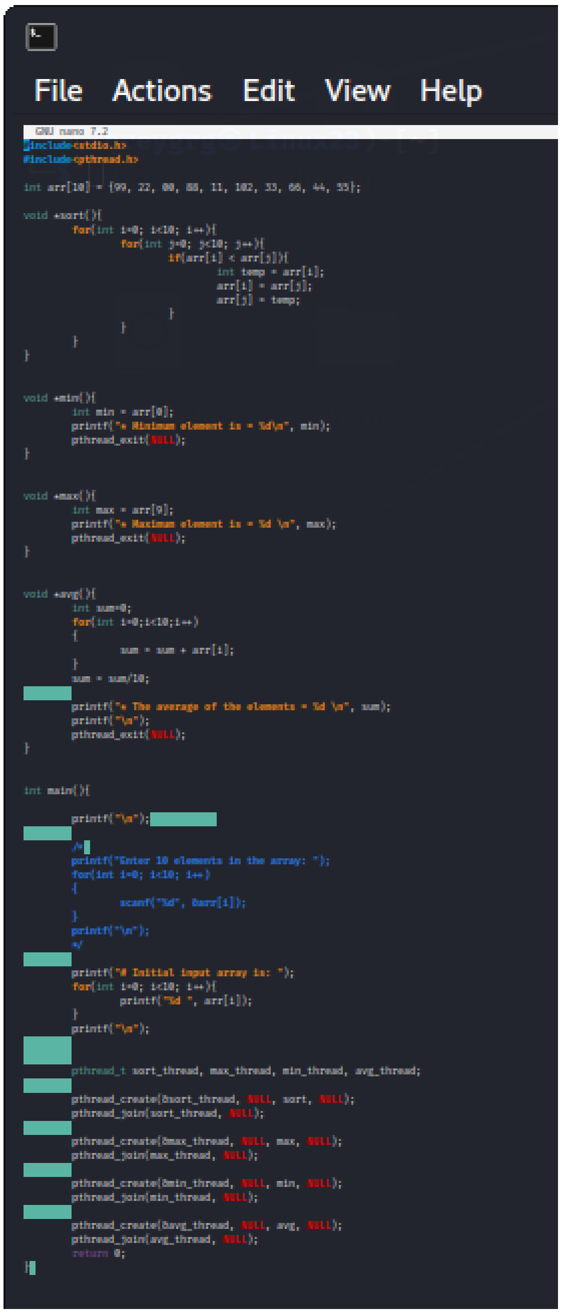
```
}



void *max(){
        int max = arr[9];
        printf("* Maximum element is = %d \n", max);
        pthread_exit(NULL);
}



void *avg(){
        int sum=0;
        for(int i=0;i<10;i++)
        {
                sum = sum + arr[i];
        }
        sum = sum/10;

        printf("* The average of the elements = %d \n", sum);
        printf("\n");
        pthread_exit(NULL);
}

int main(){

        printf("\n");

        /*
        printf("Enter 10 elements in the array: ");
```

```
	for(int i=0; i<10; i++)
	{
		scanf("%d", &arr[i]);
	}
	printf("\n");
	*/

	printf("# Initial input array is: ");
	for(int i=0; i<10; i++){
		printf("%d ", arr[i]);
	}
	printf("\n");



	pthread_t sort_thread, max_thread, min_thread, avg_thread;

	pthread_create(&sort_thread, NULL, sort, NULL);
	pthread_join(sort_thread, NULL);

	pthread_create(&max_thread, NULL, max, NULL);
	pthread_join(max_thread, NULL);

	pthread_create(&min_thread, NULL, min, NULL);
	pthread_join(min_thread, NULL);

	pthread_create(&avg_thread, NULL, avg, NULL);
	pthread_join(avg_thread, NULL);
	return 0;
}
```

Step3: - gcc 6_3.c

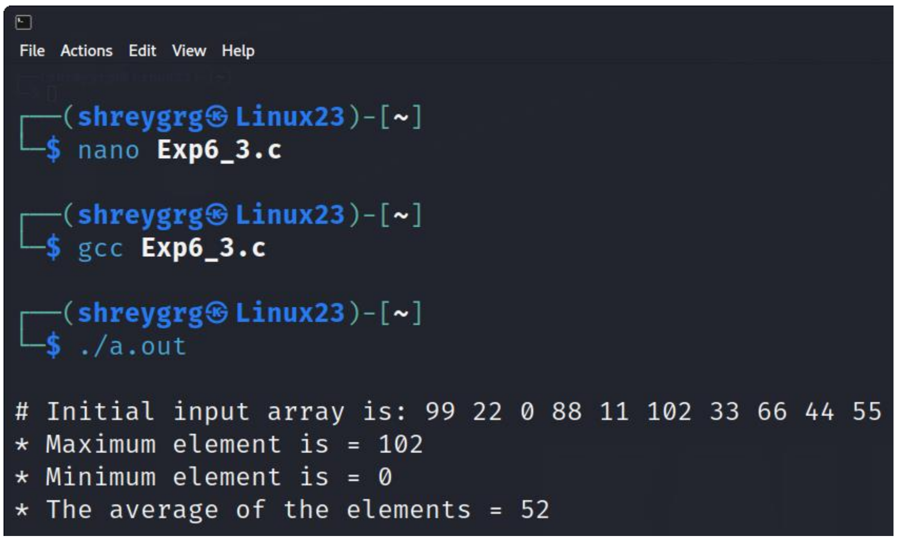
Step4: - ./a.out

nano 6_3.c



```c
#include<stdio.h>
#include<pthread.h>

int arr[10] = {99, 22, 00, 88, 11, 102, 33, 66, 44, 55};

void *sort(){
        for(int i=0; i<10; i++){
                for(int j=0; j<10; j++){
                        if(arr[i] < arr[j]){
                                int temp = arr[i];
                                arr[i] = arr[j];
                                arr[j] = temp;
                        }
                }
        }
}


void *min(){
        int min = arr[0];
        printf("* Minimum element is = %d\n", min);
        pthread_exit(NULL);
}


void *max(){
        int max = arr[9];
        printf("* Maximum element is = %d \n", max);
        pthread_exit(NULL);
}


void *avg(){
        int sum=0;
        for(int i=0;i<10;i++)
        {
                sum = sum + arr[i];
        }
        sum = sum/10;

        printf("* The average of the elements = %d \n", sum);
        printf("\n");
        pthread_exit(NULL);
}


int main(){

        printf("\n");

        /*
        printf("Enter 10 elements in the array: ");
        for(int i=0; i<10; i++)
        {
                scanf("%d", &arr[i]);
        }
        printf("\n");
        */

        printf("# Initial input array is: ");
        for(int i=0; i<10; i++){
                printf("%d ", arr[i]);
        }
        printf("\n");


        pthread_t sort_thread, max_thread, min_thread, avg_thread;

        pthread_create(&sort_thread, NULL, sort, NULL);
        pthread_join(sort_thread, NULL);

        pthread_create(&max_thread, NULL, max, NULL);
        pthread_join(max_thread, NULL);

        pthread_create(&min_thread, NULL, min, NULL);
        pthread_join(min_thread, NULL);

        pthread_create(&avg_thread, NULL, avg, NULL);
        pthread_join(avg_thread, NULL);
        return 0;
}
```

Output of the program



```
┌──(shreygrg㉿Linux23)-[~]
└─$ nano Exp6_3.c

┌──(shreygrg㉿Linux23)-[~]
└─$ gcc Exp6_3.c

┌──(shreygrg㉿Linux23)-[~]
└─$ ./a.out

# Initial input array is: 99 22 0 88 11 102 33 66 44 55
* Maximum element is = 102
* Minimum element is = 0
* The average of the elements = 52
```

Q4. Write a multithreaded program where an array of integers is passed globally and is divided into two smaller lists and given as input to two threads. The thread will sort their half of the list and will pass the sorted list to a third thread which merges and sorts the list. The final sorted list is printed by the parent thread.

Ans4. **Solution: -**

Step1: - nano 6_4.c

Step2: -

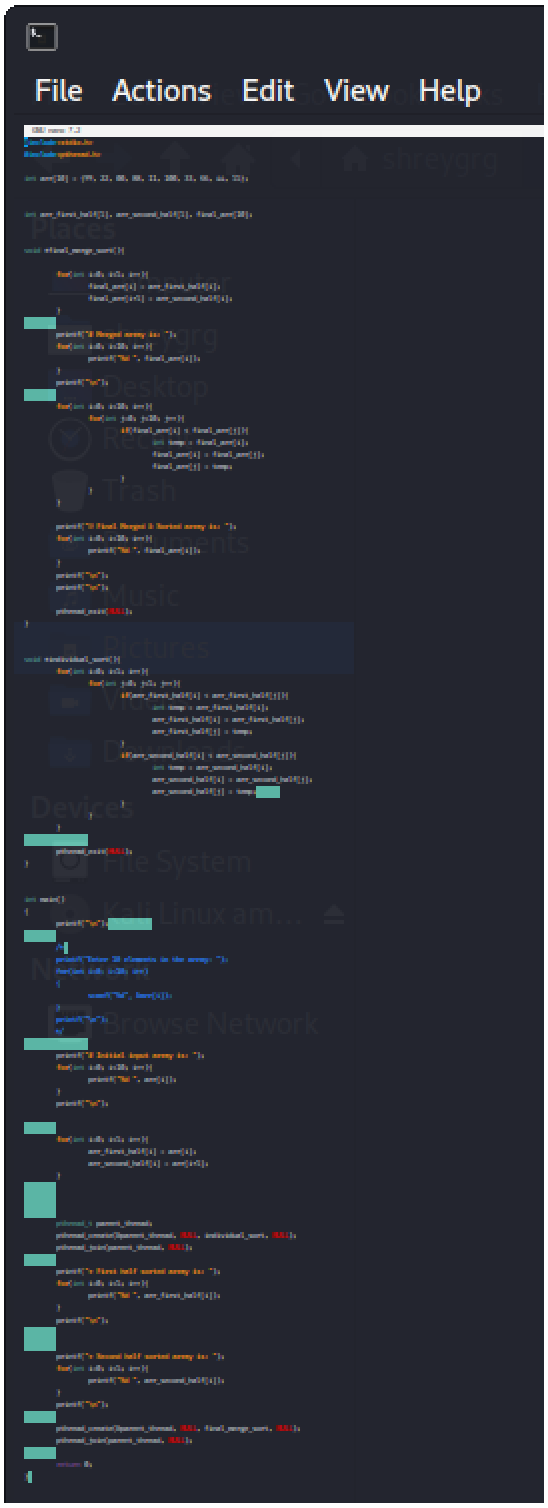
```
#include<stdio.h>

#include<pthread.h>


int arr[10] = {99, 22, 00, 88, 11, 100, 33, 66, 44, 55};



int arr_first_half[5], arr_second_half[5], final_arr[10];



void *final_merge_sort(){


        for(int i=0; i<5; i++){

                final_arr[i] = arr_first_half[i];

                final_arr[i+5] = arr_second_half[i];

        }


        printf("# Merged array is: ");

        for(int i=0; i<10; i++){

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

        }

        printf("\n");
        for(int i=0; i<10; i++){
                for(int j=0; j<10; j++){
                        if(final_arr[i] < final_arr[j]){
                                int temp = final_arr[i];
                                final_arr[i] = final_arr[j];
                                final_arr[j] = temp;
                        }
                }
        }

        printf("@ Final Merged & Sorted array is: ");
        for(int i=0; i<10; i++){
                printf("%d ", final_arr[i]);
        }
        printf("\n");
        printf("\n");

        pthread_exit(NULL);
}



void *individual_sort(){
        for(int i=0; i<5; i++){
                for(int j=0; j<5; j++){
                        if(arr_first_half[i] < arr_first_half[j]){
                                int temp = arr_first_half[i];
                                arr_first_half[i] = arr_first_half[j];
                                arr_first_half[j] = temp;
```

```
                    }
                    if(arr_second_half[i] < arr_second_half[j]){
                        int temp = arr_second_half[i];
                        arr_second_half[i] = arr_second_half[j];
                        arr_second_half[j] = temp;
                    }
                }
        }

        pthread_exit(NULL);
}

int main()
{
        printf("\n");

        /*
        printf("Enter 10 elements in the array: ");
        for(int i=0; i<10; i++)
        {
                scanf("%d", &arr[i]);
        }
        printf("\n");
        */

        printf("# Initial input array is: ");
        for(int i=0; i<10; i++){
                printf("%d ", arr[i]);
        }
        printf("\n");

        for(int i=0; i<5; i++){
                arr_first_half[i] = arr[i];
                arr_second_half[i] = arr[i+5];
        }

        pthread_t parent_thread;
        pthread_create(&parent_thread, NULL, individual_sort, NULL);
        pthread_join(parent_thread, NULL);

        printf("* First half sorted array is: ");
        for(int i=0; i<5; i++){
                printf("%d ", arr_first_half[i]);
        }
        printf("\n");



        printf("* Second half sorted array is: ");
        for(int i=0; i<5; i++){
                printf("%d ", arr_second_half[i]);
        }
        printf("\n");

        pthread_create(&parent_thread, NULL, final_merge_sort, NULL);
        pthread_join(parent_thread, NULL);

        return 0;
}
```

Step3: - gcc 6_4.c

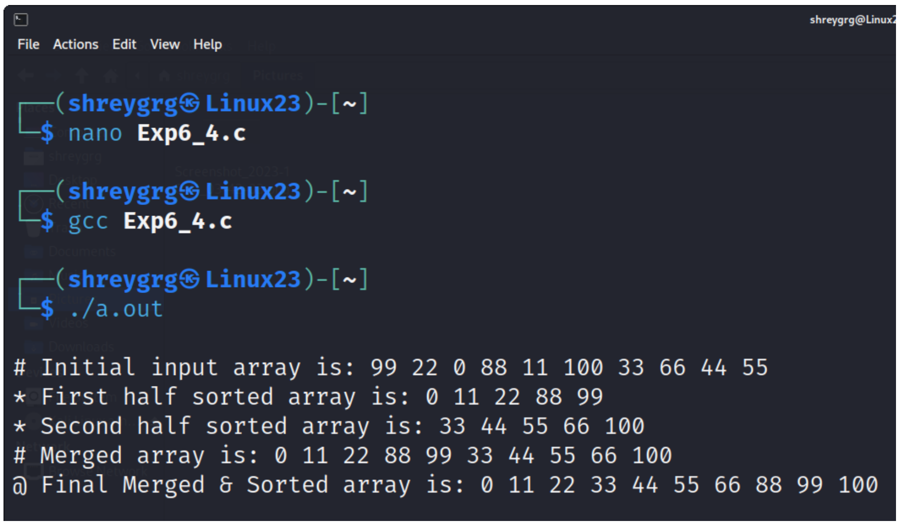
Step4: - ./a.out

nano 6_4.c



```
GNU nano 7.2
#include<stdio.h>
#include<pthread.h>

int arr[10] = {99, 22, 0, 88, 11, 100, 33, 66, 44, 55};


int arr_first_half[5], arr_second_half[5], final_arr[10];

void *final_merge_sort(){

        for(int i=0; i<5; i++){
                final_arr[i] = arr_first_half[i];
                final_arr[i+5] = arr_second_half[i];
        }

        printf("# Merged array is: ");
        for(int i=0; i<10; i++){
                printf("%d ", final_arr[i]);
        }
        printf("\n");

        for(int i=0; i<10; i++){
                for(int j=0; j<10; j++){
                        if(final_arr[i] < final_arr[j]){
                                int temp = final_arr[i];
                                final_arr[i] = final_arr[j];
                                final_arr[j] = temp;
                        }
                }
        }

        printf("@ Final Merged & Sorted array is: ");
        for(int i=0; i<10; i++){
                printf("%d ", final_arr[i]);
        }
        printf("\n");
        printf("\n");

        pthread_exit(NULL);
}


void *individual_sort(){
        for(int i=0; i<5; i++){
                for(int j=0; j<5; j++){
                        if(arr_first_half[i] < arr_first_half[j]){
                                int temp = arr_first_half[i];
                                arr_first_half[i] = arr_first_half[j];
                                arr_first_half[j] = temp;
                        }
                        if(arr_second_half[i] < arr_second_half[j]){
                                int temp = arr_second_half[i];
                                arr_second_half[i] = arr_second_half[j];
                                arr_second_half[j] = temp;
                        }
                }
        }

        pthread_exit(NULL);
}


int main()
{
        printf("\n");

        /*
        printf("Enter 10 elements in the array: ");
        for(int i=0; i<10; i++)
        {
                scanf("%d", &arr[i]);
        }
        printf("\n");
        */

        printf("# Initial input array is: ");
        for(int i=0; i<10; i++){
                printf("%d ", arr[i]);
        }
        printf("\n");


        for(int i=0; i<5; i++){
                arr_first_half[i] = arr[i];
                arr_second_half[i] = arr[i+5];
        }



        pthread_t parent_thread;
        pthread_create(&parent_thread, NULL, individual_sort, NULL);
        pthread_join(parent_thread, NULL);

        printf("* First half sorted array is: ");
        for(int i=0; i<5; i++){
                printf("%d ", arr_first_half[i]);
        }
        printf("\n");


        printf("* Second half sorted array is: ");
        for(int i=0; i<5; i++){
                printf("%d ", arr_second_half[i]);
        }
        printf("\n");

        pthread_create(&parent_thread, NULL, final_merge_sort, NULL);
        pthread_join(parent_thread, NULL);

        return 0;
}
```

Output of the program



```
┌──(shreygrg㉿Linux23)-[~]
└─$ nano Exp6_4.c

┌──(shreygrg㉿Linux23)-[~]
└─$ gcc Exp6_4.c

┌──(shreygrg㉿Linux23)-[~]
└─$ ./a.out

# Initial input array is: 99 22 0 88 11 100 33 66 44 55
* First half sorted array is: 0 11 22 88 99
* Second half sorted array is: 33 44 55 66 100
# Merged array is: 0 11 22 88 99 33 44 55 66 100
@ Final Merged & Sorted array is: 0 11 22 33 44 55 66 88 99 100
```

Submitted By: -

Shrey Garg (12218692)

Submitted to: -

Mr. Ashish Kumar Singh

Section: K22QY