```c
#include <stdio.h>
#include <math.h>
#include <time.h>

int a[50], n, j, i, temp, flag, ub, lb, pivot, start, end, loc;

void swap(int *xp, int *yp)
{
   int temp = *xp;
   *xp = *yp;
   *yp = temp;
}

void insertionsort(int a[], int n)
{
   for (i = 1; i < n; i++)
   {
       temp = a[i];
       j = i - 1;
       while (i >= 0 && a[j] > temp)
       {
           a[j + 1] = a[j];
           j--;
       }
       a[j + 1] = temp;
   }
}

void bubblesort(int a[], int n)
{

   for (i = 0; i < n - 1; i++)
   {
       flag = 0;
       for (j = 0; j < n - 1; j++)
       {
           if (a[j] > a[j + 1])
           {
               temp = a[j];
               a[j] = a[j + 1];
               a[j + 1] = temp;
               flag = 1;
           }
       }
   }
```

```
}

void selectionsort(int a[], int n)
{

   for (i = 0; i < n - 1; i++)
   {
       int min = i;
       for (j = i + 1; j < n; j++)
       {
           if (a[j] < a[min])
           {
               min = j;
           }
       }
       if (min != i)
       {
           swap(&a[i], &a[min]);
       }
   }
}

int partition(int arr[50], int lb, int ub)
{
   pivot = arr[lb];
   start = lb;
   end = ub;
   while (start < end)
   {
       while (arr[start] <= pivot)
       {
           start++;
       }
       while (arr[end] > pivot)
       {
           end--;
       }
       if (start < end)
       {
           swap(&arr[start], &arr[end]);
       }
   }
   swap(&arr[lb], &arr[end]);
   return end;
}
```

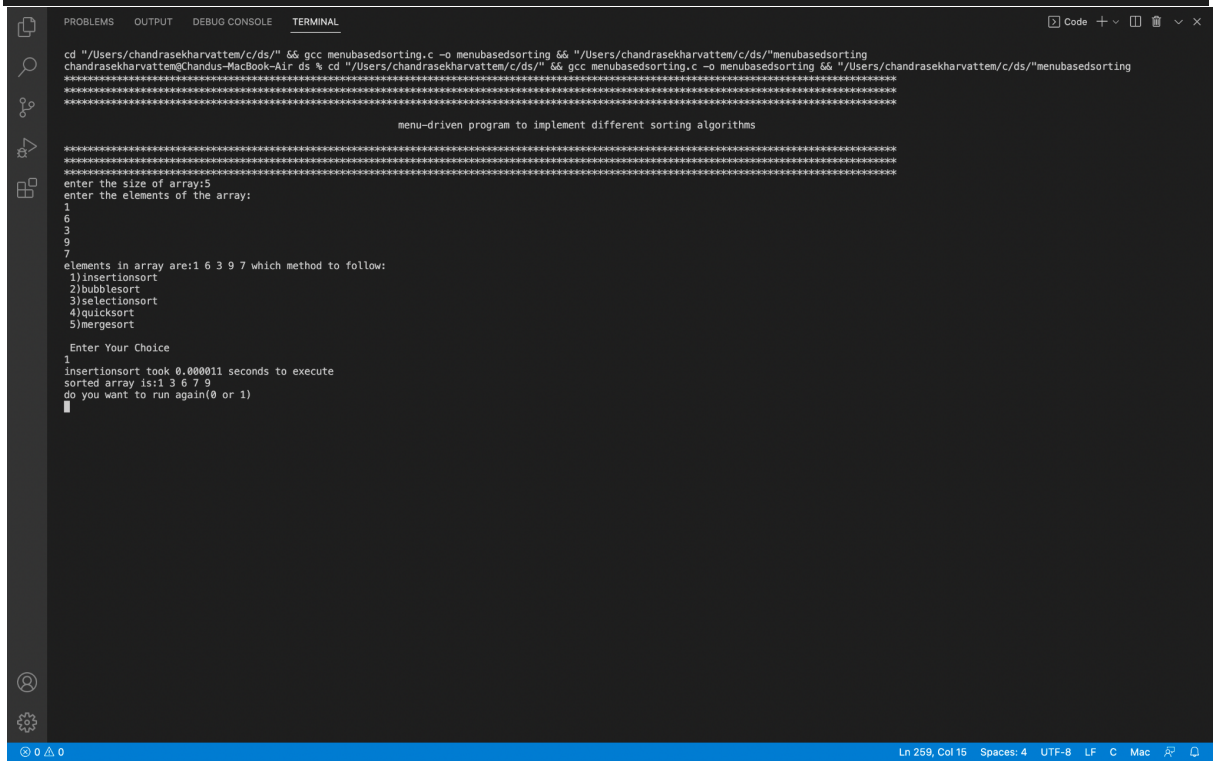
```
void quicksort(int arr[50], int lb, int ub)
{
   if (lb < ub)
   {
       loc = partition(a, lb, ub);
       quicksort(a, lb, loc - 1);
       quicksort(a, loc + 1, ub);
   }
}

void merge(int arr[], int l, int m, int r)
{

   int i, j, k;
   int n1 = m - l + 1;
   int n2 = r - m;

   int L[n1], R[n2];

   for (i = 0; i < n1; i++)
       L[i] = arr[l + i];
   for (j = 0; j < n2; j++)
       R[j] = arr[m + 1 + j];

   i = 0;
   j = 0;
   k = l;
   while (i < n1 && j < n2)
   {
       if (L[i] <= R[j])
       {
           arr[k] = L[i];
           i++;
       }
       else
       {
           arr[k] = R[j];
           j++;
       }
       k++;
   }

   while (i < n1)
   {
```

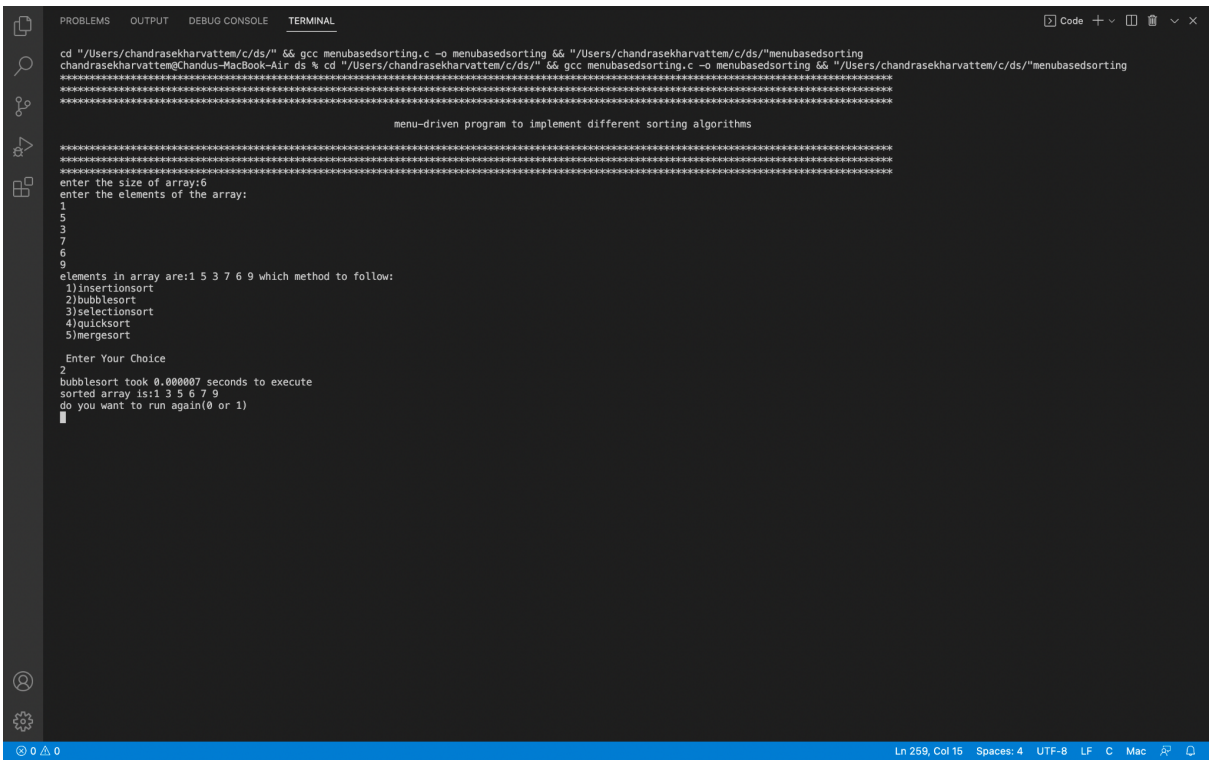
```
        arr[k] = L[i];
        i++;
        k++;
    }

    while (j < n2)
    {
        arr[k] = R[j];
        j++;
        k++;
    }
}

void mergesort(int arr[], int l, int r)
{
    if (l < r)
    {

        int m = l + (r - l) / 2;

        mergesort(arr, l, m);
        mergesort(arr, m + 1, r);

        merge(arr, l, m, r);
    }
}

void sortedarray()
{
    printf("sorted array is:");

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

int main()
{
    int choice;
    int t1, t2, t3, t4, t5;
    int m=1;

printf("**************************************************************************
************************************************************\n");
```

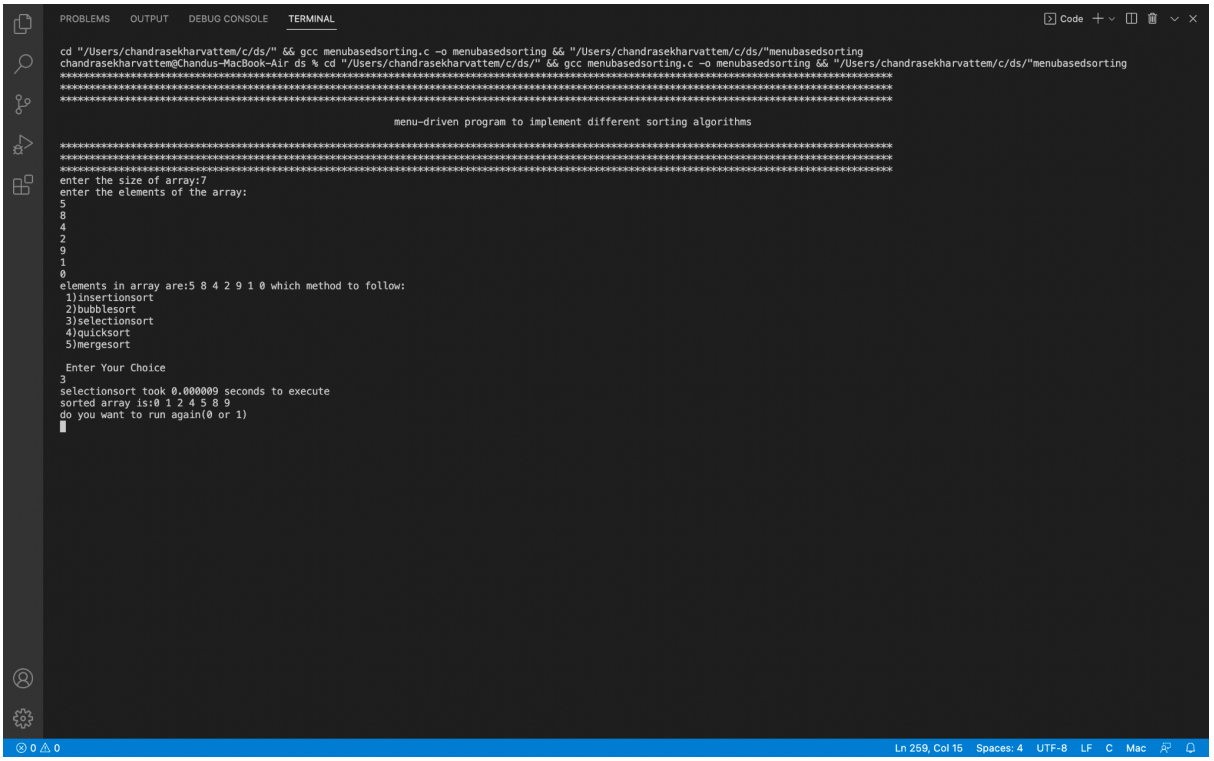
```
printf("*********************************************************************
**************************************************************\n");

printf("*********************************************************************
**************************************************************\n");
   printf("\n  \t \t \t \t \t \t \t menu-driven program to implement different
sorting algorithms \n \n");

printf("*********************************************************************
**************************************************************\n");

printf("*********************************************************************
**************************************************************\n");

printf("*********************************************************************
**************************************************************\n");
   while(m)
   {printf("enter the size of array:");
   scanf("%d", &n);
   printf("enter the elements of the array:\n");

   for (i = 0; i < n; i++)
   {
       scanf("%d", &a[i]);
   }

   printf("elements in array are:");

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

   printf("which method to follow: \n 1)insertionsort \n 2)bubblesort \n
3)selectionsort \n 4)quicksort\n 5)mergesort\n");
   printf("\n Enter Your Choice \n");
   scanf("%d", &choice);

   switch (choice)
   {
   case 1:
       t1 = clock();
       insertionsort(a, n);
```

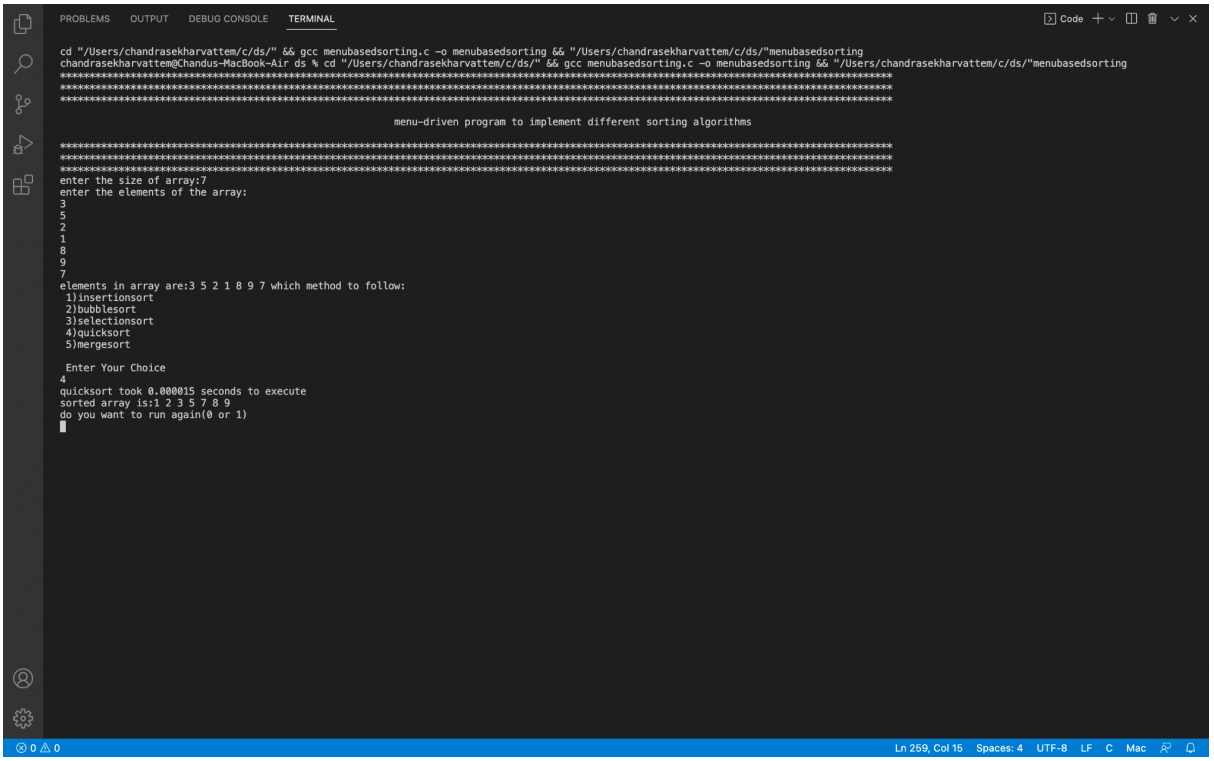
```
        t1 = clock() - t1;
        double time_taken1 = ((double)t1) / CLOCKS_PER_SEC;
        printf("insertionsort took %f seconds to execute \n", time_taken1);
        break;

    case 2:
        t2 = clock();
        bubblesort(a, n);

        t2 = clock() - t2;
        double time_taken2 = ((double)t2) / CLOCKS_PER_SEC;
        printf("bubblesort took %f seconds to execute \n", time_taken2);
        break;

    case 3:
        t3 = clock();
        selectionsort(a, n);

        t3 = clock() - t3;
        double time_taken3 = ((double)t3) / CLOCKS_PER_SEC;
        printf("selectionsort took %f seconds to execute \n", time_taken3);
        break;

    case 4:

        t4 = clock();
        quicksort(a, 0, n - 1);

        t4 = clock() - t4;
        double time_taken4 = ((double)t4) / CLOCKS_PER_SEC;
        printf("quicksort took %f seconds to execute \n", time_taken4);
        break;

    case 5:
        t5 = clock();
        mergesort(a, 0, n - 1);

        t5 = clock() - t5;
        double time_taken5 = ((double)t5) / CLOCKS_PER_SEC;
        printf("mergesort took %f seconds to execute \n", time_taken5);
        break;

    default:
        printf("invalid choice\n");
        break;
```

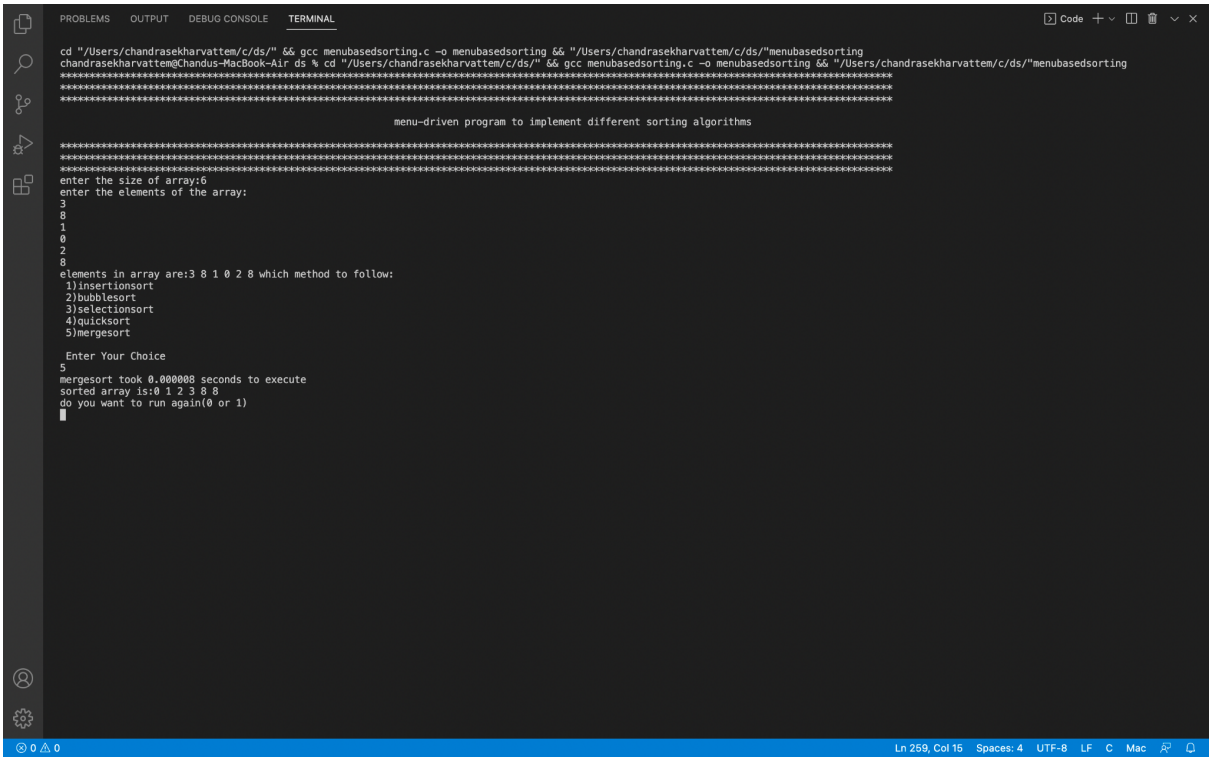
```c
    }
    sortedarray();
    printf("\ndo you want to run again(0 or 1)\n");
    scanf("%d",&m);}
    printf("program ended!");
    return 0;
}
```






PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
cd "/Users/chandrasekharvattem/c/ds/" && gcc menubasedsorting.c -o menubasedsorting && "/Users/chandrasekharvattem/c/ds/"menubasedsorting
chandrasekharvattem@Chandus-MacBook-Air ds % cd "/Users/chandrasekharvattem/c/ds/" && gcc menubasedsorting.c -o menubasedsorting && "/Users/chandrasekharvattem/c/ds/"menubasedsorting
****
****
****
menu-driven program to implement different sorting algorithms
****
****
****
enter the size of array:7
enter the elements of the array:
5
8
4
2
9
1
0
elements in array are:5 8 4 2 9 1 0 which method to follow:
1)insertionsort
2)bubblesort
3)selectionsort
4)quicksort
5)mergesort
Enter Your Choice
3
selectionsort took 0.000009 seconds to execute
sorted array is:0 1 2 4 5 8 9
do you want to run again(0 or 1)
0 0
Ln 259, Col 15
Spaces: 4
UTF-8
LF
C
Mac


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
cd "/Users/chandrasekharvattem/c/ds/" && gcc menubasedsorting.c -o menubasedsorting && "/Users/chandrasekharvattem/c/ds/"menubasedsorting
chandrasekharvattem@Chandus-MacBook-Air ds % cd "/Users/chandrasekharvattem/c/ds/" && gcc menubasedsorting.c -o menubasedsorting && "/Users/chandrasekharvattem/c/ds/"menubasedsorting
menu-driven program to implement different sorting algorithms
enter the size of array:7
enter the elements of the array:
3
5
2
1
8
9
7
elements in array are:3 5 2 1 8 9 7 which method to follow:
1)insertionsort
2)bubblesort
3)selectionsort
4)quicksort
5)mergesort
Enter Your Choice
4
quicksort took 0.000015 seconds to execute
sorted array is:1 2 3 5 7 8 9
do you want to run again(0 or 1)
0 ⚠ 0
Ln 259, Col 15
Spaces: 4
UTF-8
LF
C
Mac


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
cd "/Users/chandrasekharvattem/c/ds/" && gcc menubasedsorting.c -o menubasedsorting && "/Users/chandrasekharvattem/c/ds/"menubasedsorting
chandrasekharvattem@Chandus-MacBook-Air ds % cd "/Users/chandrasekharvattem/c/ds/" && gcc menubasedsorting.c -o menubasedsorting && "/Users/chandrasekharvattem/c/ds/"menubasedsorting
menu-driven program to implement different sorting algorithms
enter the size of array:6
enter the elements of the array:
3
8
1
0
2
8
elements in array are:3 8 1 0 2 8 which method to follow:
1)insertionsort
2)bubblesort
3)selectionsort
4)quicksort
5)mergesort
Enter Your Choice
5
mergesort took 0.000008 seconds to execute
sorted array is:0 1 2 3 8 8
do you want to run again(0 or 1)
0 ⚠ 0
Ln 259, Col 15
Spaces: 4
UTF-8
LF
C
Mac