

Date : 27-08-2025

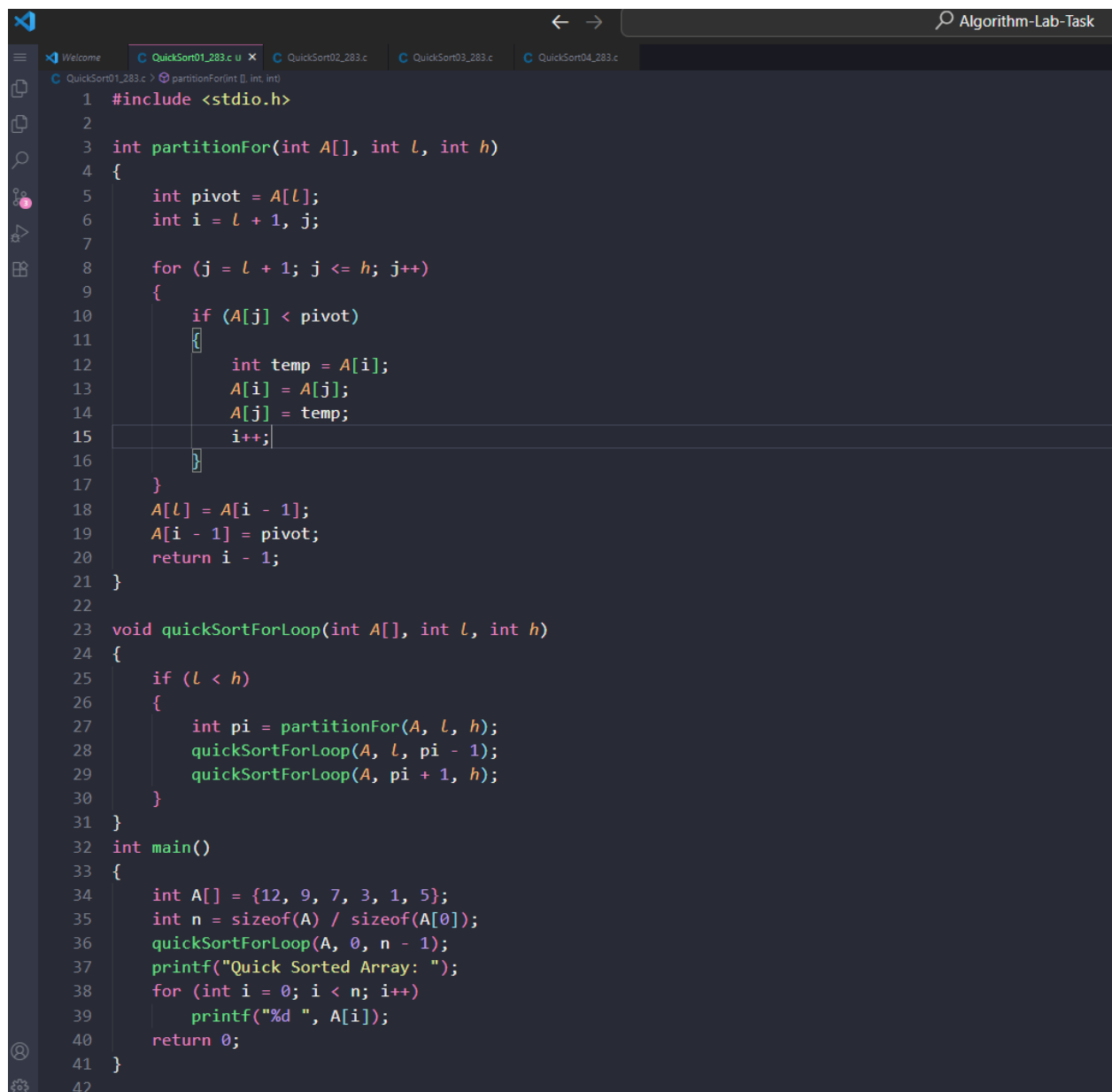
Name : Amit Sarkar

Intake : 51

ID : 22235103283

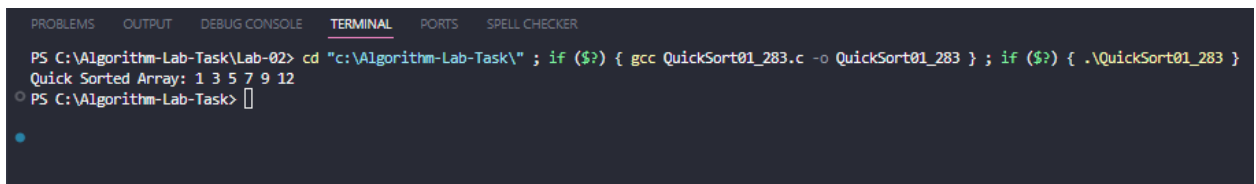
Lab Report 02

1. Take the pivot element as the first element of the array. Using For Loop.



```
1 #include <stdio.h>
2
3 int partitionFor(int A[], int l, int h)
4 {
5     int pivot = A[l];
6     int i = l + 1, j;
7
8     for (j = l + 1; j <= h; j++)
9     {
10         if (A[j] < pivot)
11         {
12             int temp = A[i];
13             A[i] = A[j];
14             A[j] = temp;
15             i++;
16         }
17     }
18     A[l] = A[i - 1];
19     A[i - 1] = pivot;
20     return i - 1;
21 }
22
23 void quickSortForLoop(int A[], int l, int h)
24 {
25     if (l < h)
26     {
27         int pi = partitionFor(A, l, h);
28         quickSortForLoop(A, l, pi - 1);
29         quickSortForLoop(A, pi + 1, h);
30     }
31 }
32
33 int main()
34 {
35     int A[] = {12, 9, 7, 3, 1, 5};
36     int n = sizeof(A) / sizeof(A[0]);
37     quickSortForLoop(A, 0, n - 1);
38     printf("Quick Sorted Array: ");
39     for (int i = 0; i < n; i++)
40         printf("%d ", A[i]);
41     return 0;
42 }
```

OutPut :

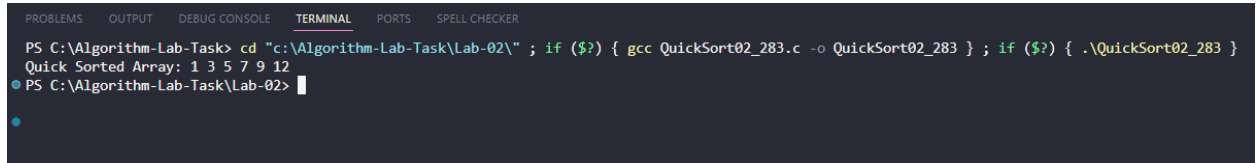


```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SPELL CHECKER
PS C:\Algorithm-Lab-Task\Lab-02> cd "c:\Algorithm-Lab-Task\" ; if ($?) { gcc QuickSort01_283.c -o QuickSort01_283 } ; if ($?) { .\QuickSort01_283 }
Quick Sorted Array: 1 3 5 7 9 12
PS C:\Algorithm-Lab-Task> 
```

2. Take the pivot element as the first element of the array. Using While Loop.

```
1 #include <stdio.h>
2
3 int partitionFor(int A[], int l, int h)
4 {
5     int pivot = A[l];
6     int i = l + 1, j;
7
8     for (j = l + 1; j <= h; j++)
9     {
10         if (A[j] < pivot)
11         {
12             int temp = A[i];
13             A[i] = A[j];
14             A[j] = temp;
15             i++;
16         }
17     }
18     A[l] = A[i - 1];
19     A[i - 1] = pivot;
20     return i - 1;
21 }
22
23 void quickSortWhileLoop(int A[], int l, int h)
24 {
25     if (l < h)
26     {
27         int pi = partitionFor(A, l, h);
28         quickSortWhileLoop(A, l, pi - 1);
29         quickSortWhileLoop(A, pi + 1, h);
30     }
31 }
32
33 int main()
34 {
35     int A[] = {12, 9, 7, 3, 1, 5};
36     int n = sizeof(A) / sizeof(A[0]);
37     quickSortWhileLoop(A, 0, n - 1);
38     printf("Quick Sorted Array: ");
39     for (int i = 0; i < n; i++)
40         printf("%d ", A[i]);
41     return 0;
42 }
```

OutPut :



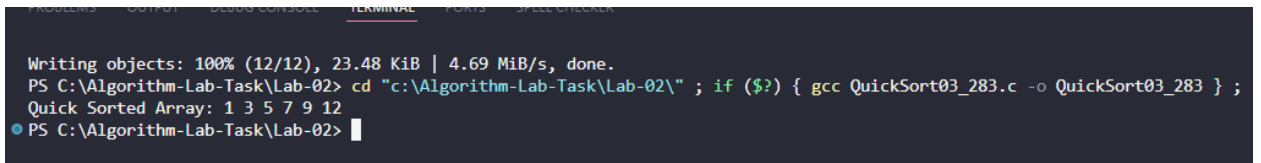
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SPELL CHECKER

PS C:\Algorithm-Lab-Task> cd "c:\Algorithm-Lab-Task\Lab-02\" ; if ($?) { gcc QuickSort02_283.c -o QuickSort02_283 } ; if ($?) { .\QuickSort02_283 }
Quick Sorted Array: 1 3 5 7 9 12
PS C:\Algorithm-Lab-Task\Lab-02>
```

3. *Take the pivot element as the first element of the array. Using Do While*

```
Lab-02 > C QuickSort03_283.c > main()
1  #include <stdio.h>
2
3  int partitionFor(int A[], int l, int h)
4  {
5      int pivot = A[l];
6      int i = l + 1, j;
7
8      for (j = l + 1; j <= h; j++)
9      {
10         if (A[j] < pivot)
11         {
12             int temp = A[i];
13             A[i] = A[j];
14             A[j] = temp;
15             i++;
16         }
17     }
18     A[l] = A[i - 1];
19     A[i - 1] = pivot;
20     return i - 1;
21 }
22
23 void quickSortDoWhileLoop(int A[], int l, int h)
24 {
25     if (l < h)
26     {
27         int pi = partitionFor(A, l, h);
28         quickSortDoWhileLoop(A, l, pi - 1);
29         quickSortDoWhileLoop(A, pi + 1, h);
30     }
31 }
32 int main()
33 {
34     int A[] = {12, 9, 7, 3, 1, 5};
35     int n = sizeof(A) / sizeof(A[0]);
36     quickSortDoWhileLoop(A, 0, n - 1);
37     printf("Quick Sorted Array: ");
38     for (int i = 0; i < n; i++)
39         printf("%d ", A[i]);
40     return 0;
41 }
42
```

OutPut :



```
Writing objects: 100% (12/12), 23.48 KiB | 4.69 MiB/s, done.
PS C:\Algorithm-Lab-Task\Lab-02> cd "c:\Algorithm-Lab-Task\Lab-02\" ; if ($?) { gcc QuickSort03_283.c -o QuickSort03_283 } ;
Quick Sorted Array: 1 3 5 7 9 12
PS C:\Algorithm-Lab-Task\Lab-02> 
```

- **4. Take the pivot element as the Last element of the array. Using For Loop**

Lab-02 > QuickSort04_283.c > main()

```
1  #include <stdio.h>
2
3  ~int partitionLast(int A[], int L, int h)
4  {
5      int pivot = A[h];
6      int i = L - 1;
7
8      ~for (int j = L; j < h; j++)
9      {
10         ~if (A[j] < pivot)
11         {
12             i++;
13             int temp = A[i];
14             A[i] = A[j];
15             A[j] = temp;
16         }
17     }
18     int temp = A[i + 1];
19     A[i + 1] = A[h];
20     A[h] = temp;
21
22     return i + 1;
23 }
24
25 ~void quickSortLastElement(int A[], int L, int h)
26 {
27     ~if (L < h)
28     {
29         int pi = partitionLast(A, L, h);
30         quickSortLastElement(A, L, pi - 1);
31         quickSortLastElement(A, pi + 1, h);
32     }
33 }
34
35 ~int main()
36 {
37     int A[] = {12, 9, 7, 3, 1, 5};
38     int n = sizeof(A) / sizeof(A[0]);
39     quickSortLastElement(A, 0, n - 1);
40     printf("Quick Sorted Array: ");
41     for (int i = 0; i < n; i++)
42         printf("%d ", A[i]);
43
44     return 0;
45 }
46
```

OutPut :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SPELL CHECKER

PS C:\Algorithm-Lab-Task\Lab-02> cd "c:\Algorithm-Lab-Task\Lab-02\" ; if ($?) { gcc QuickSort04_283.c -o QuickSort04_283 } ;
Quick Sorted Array: 1 3 5 7 9 12
● PS C:\Algorithm-Lab-Task\Lab-02> 
```