```c
/*
* ques1.c
* To sort a set of elements using Quick sort algorithm.
*   Created On: 4-19-2018
*       Author: shahique
*/

#include <stdio.h>
#include <stdbool.h>
#define MAX 7

int intArray[MAX] = {4,6,3,2,1,9,7};

void printline(int count) {
   int i;

   for(i = 0;i <count-1;i++) {
      printf("=");
   }

   printf("=\n");
}

void display() {
   int i;
   printf("[");

   // navigate through all items
   for(i = 0;i<MAX;i++) {
      printf("%d ",intArray[i]);
   }

   printf("]\n");
}

void swap(int num1, int num2) {
   int temp = intArray[num1];
   intArray[num1] = intArray[num2];
   intArray[num2] = temp;
}

int partition(int left, int right, int pivot) {
   int leftPointer = right -1;
   int rightPointer = right;

   while(true) {
      while(intArray[++leftPointer] < pivot) {
         //do nothing
      }

      while(rightPointer > 0 && intArray[--rightPointer] > pivot) {
         //do nothing
      }

      if(leftPointer >= rightPointer) {
         break;
      } else {
         printf(" item swapped :%d,%d\n", intArray[leftPointer],intArray[rightPointer]);
         swap(leftPointer,rightPointer);
      }
   }

   printf(" pivot swapped :%d,%d\n", intArray[leftPointer],intArray[right]);
   swap(leftPointer,right);
   printf("Updated Array: ");
   display();
```

```
   return leftPointer;
}

void quickSort(int left, int right) {
   if(right-left <= 0) {
      return;
   } else {
      int pivot = intArray[right];
      int partitionPoint = partition(left, right, pivot);
      quickSort(left,partitionPoint-1);
      quickSort(partitionPoint+1,right);
   }
}

main() {
   printf("Input Array: ");
   display();
   printline(50);
   quickSort(0,MAX-1);
   printf("Output Array: ");
   display();
   printline(50);
}
```