// Quick Sort

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

// partition function

int partition(int arr[], int low, int high)
{
   // pivot(element to be placed at right position)
   int pivot = arr[high];
   int i = (low-1);

   for(int j = low; j<=high-1;j++)
   {
      if(arr[j]<pivot)
      {
         i++;
         int t;
         t = arr[i];
         arr[i] = arr[j];
         arr[j] = t;
      }
   }

   // Swapping value
   int x;
```

```
    x = arr[i+1];
    arr[i+1] = arr[high];
    arr[high]=x;

    return(i+1);
}

// quick sort function
int quickSort(int arr[],int low, int high)
{
    if(low<high)
    {
        int pi = partition(arr,low,high);
        quickSort(arr,low,pi-1);
        quickSort(arr,pi+1,high);
    }

}

int printArray(int arr[], int size) {
    for (int i = 0; i < size; i++)
    printf("%d ", arr[i]);
    }


int main()
{
    int arr[] = {40,20,8,80,2,10};
    int size = sizeof(arr) / sizeof(arr[0]);
```

```
    quickSort(arr, 0, size - 1);

    printf("Array After Sort: ");
    printArray(arr, size);

    return 0;
}
```

Output:

```
Array After Sort: 2 8 10 20 40 80
Process returned 0 (0x0)   execution time : 0.069 s
Press any key to continue.
```

Time Complexity:

Best Case: $T(n) = 2T(n/2) + 0(n)$