2.3 Модуль QuickSort

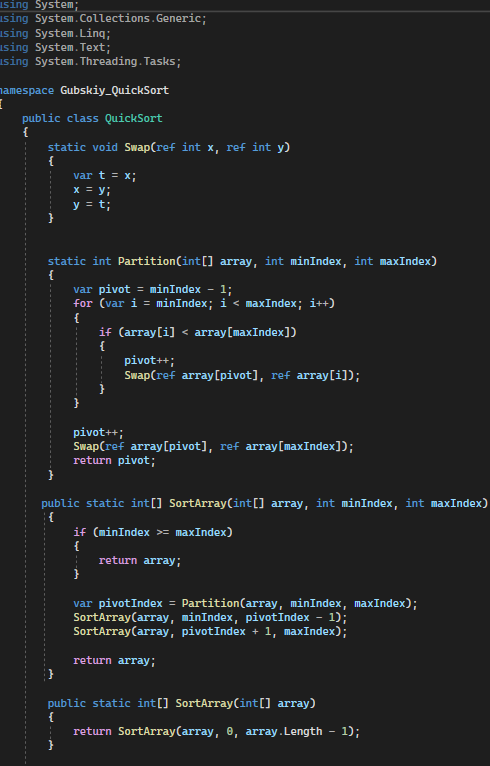


Рисунок 10 – Модуль QuickSort

На данном рисунке представлена реализация метода быстрой сортировки, а именно, весь необходимый для реализации внутри программы код.

Алгоритм быстрой сортировки является рекурсивным, поэтому для простоты процедура на вход будет принимать границы участка массива от l включительно и до r не включительно. Понятно, что для того, чтобы отсортировать весь массив, в качестве параметра l надо передать 0, а в качестве r — n, где по традиции n обозначает длину массива.  
  
В основе алгоритма быстрой сортировке лежит процедура partition. Partition выбирает некоторый элемент массива и переставляет элементы участка массива таким образом, чтобы массив разбился на 2 части: левая часть содержит элементы, которые меньше этого элемента, а правая часть содержит элементы, которые больше или равны этого элемента. Такой разделяющий элемент называется пивотом.  
  
Реализация partiion'а:

partition(l, r):

pivot = a[random(l ... r - 1)]

m = l

for i = l ... r - 1:

if a[i] < pivot:

swap(a[i], a[m])

m++

return m

Пивот в нашем случае выбирается случайным образом. Такой алгоритм называется рандомизированным. На самом деле пивот можно выбирать самым разным образом: либо брать случайный элемент, либо брать первый / последний элемент учаcтка, либо выбирать его каким-то «умным» образом. Выбор пивота является очень важным для итоговой сложности алгоритма сортировки, но об этом несколько позже. Сложность же процедуры partition — O(n), где n = r — l — длина участка.  
  
Теперь используем partition для реализации сортировки:  
  
Реализация partiion'а:

sort(l, r):

if r - l = 1:

return

m = partition(l, r)

sort(l, m)

sort(m, r)

Крайний случай — массив из одного элемента обладает свойством упорядоченности. Если массив длинный, то применяем partition и вызываем процедуру рекурсивно для двух половин массива.  
  
Если прогнать написанную сортировку на примере массива 1 2 2, то можно заметить, что она никогда не закончится. Почему так получилось?  
  
При написании partition мы сделали допущение — все элементы массива должны быть уникальны. В противном случае возвращаемое значение m будет равно l и рекурсия никогда не закончится, потому как sort(l, m) будет вызывать sort(l, l) и sort(l, m). Для решения данной проблемы надо массив разделять не на 2 части (< pivot и >= pivot), а на 3 части (< pivot, = pivot, > pivot) и вызывать рекурсивно сортировку для 1-ой и 3-ей частей. [2]