СОРТИРОВКА ПУЗЫРЬКОМ

Выполнил: Артемов Е. А. студент группы 143

Научный руководитель: старший преподаватель Григорьев С. В.

17 декабря 2015

ВВЕДЕНИЕ В РАБОТУ И ОСОБЕННОСТИ АЛГОРИТМА

Сортировка пузырьком – один из простейших алгоритмов сортировки, применяющийся как учебный алгоритм. Лежит в основе некоторых более совершенных алгоритмов сортировки.

Сложность алгоритма: $O(n^2)$

ОБЗОР АЛЬТЕРНАТИВНЫХ АЛГОРИТМОВ

1. Шейкерная сортировка
   - Сложность: $O(n^2)$
   - Не рассматривает повторно отсортированные части массива
2. Пирамидальная сортировка
   - Сложность: $O(n \log n)$
   - Количество применяемой служебной памяти не зависит от размера массива
3. Быстрая сортировка
   - Один из самых быстрых известных универсальных алгоритмов сортировки Сложность: $O(n \log n)$
   - является существенно улучшенным вариантом алгоритма пузырьковой сортировки

ЦЕЛИ И ЗАДАЧИ

Задачи:

- Исследовать существующий алгоритм
- Разработать собственный алгоритм для списков
- Реализовать и протестировать алгоритм

Цель работы: реализация алгоритма сортировки пузырьком

СТРУКТУРА СПИСКА

Любой элемент списка описывается формулой:

Список типа A = Пустой список + (A × Список типа A)

Список вида [4; 2; 3; 1] представляется как

(Констр(4 ,Констр(2, Констр(3, Констр(1, Пусто)))))

РЕАЛИЗАЦИЯ

Требования к реализации:

1. Обрабатываются пустые и частично (или полностью) отсортированные списки
2. Корректно выводится ответ – полностью отсортированный список чисел

Составляющие:

- Цикл, в котором происходят следующие действия:
  - Элементы попарно сравниваются
  - Элементы, стоящие в порядке убывания меняются местами
- Вывод

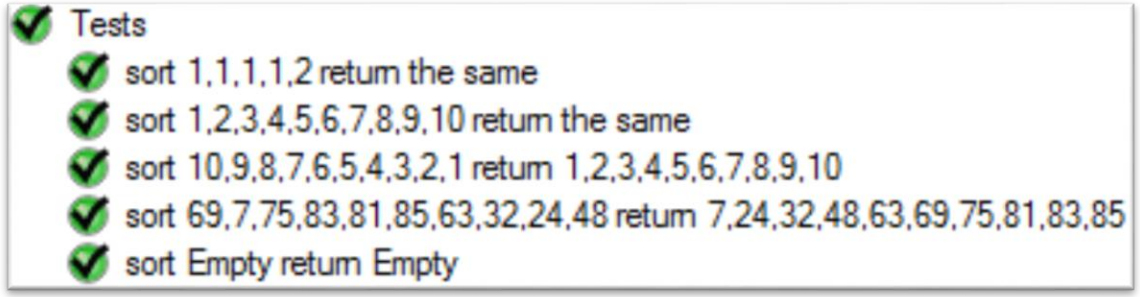
```
let main (inList:MyList<int>) =
    let rec listMove lst =
        match lst with
        | Empty -> lst
        | Cons (hd, Empty) -> lst
        | Cons (hd, tl) ->
            if hd > tl.getHead()
            then Cons (tl.getHead(), listMove (Cons (hd, tl.getTail())))
            else Cons (hd, listMove tl)

    let rec bubblesort lst i =
        if i < inList.length()
        then
            bubblesort (listMove lst) (i + 1)
        else
            lst

    bubblesort inList 0
```

ТЕСТИРОВАНИЕ

- Тестирование проводилось на списках:
  - Отсортированных и с повторяющимися элементами для проверки стабильности
  - Список, порядок элементов в котором обратен возрастающему – нормальная работа алгоритма
- Пустой список – формальная корректность
- Список случайных значений – работа в обычных условиях



РЕЗУЛЬТАТЫ

- Исследован и скорректирован существующий алгоритм
- Разработана собственная реализация алгоритма для списков
- Реализация протестирована