СОРТИРОВКА ПУЗЫРЬКОМ

Выполнил: Артемов Е. А. студент группы 143

Научный руководитель: старший преподаватель Григорьев С. В.

17 декабря 2015

ПРЕДМЕТНАЯ ОБЛАСТЬ

- Сортировка пузырьком – один из простейших алгоритмов сортировки
- Применение
  1. Как учебный алгоритм
  2. 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

Цель работы: реализация алгоритма сортировки пузырьком

Задачи:

- Исследовать существующий алгоритм
- Реализовать алгоритм, работающий со списками
- Протестировать собственную реализацию

СТРУКТУРА СПИСКА

```
type MyList<'t> =
   | Empty
   | Cons of 't * MyList<'t>
```

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

(Cons(4, Cons(2, Cons(3, Cons(1, Empty)))))

ТРЕБОВАНИЯ

Требования к реализации:

1. Обрабатываются пустые списки
2. Обрабатываются частично или полностью отсортированные списки
3. Результат работы программы – полностью отсортированный список чисел

РЕАЛИЗАЦИЯ

Составляющие:

- Цикл, в котором происходят следующие действия:
  1. Элементы попарно сравниваются
  2. 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
  - Список, порядок элементов в котором обратен возрастающему
  - Пустой список
  - Список случайных значений

РЕЗУЛЬТАТЫ

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