# Test1

// проверка обычной листа и моего реализация стандартных функций

list: [10, 20, 30] Mylist: [10, 20, 30]

**Добавление по индексу**

list: [10, 101, 20, 30] Mylist: [10, 101, 20, 30]

**замена по индексу**

list: [10, 22, 20, 30] Mylist: [10, 22, 20, 30]

**Вывод по индексу 3**

list: 30 Mylist: 30

**Удаление по индексу 2**

list: [10, 22, 30] Mylist: [10, 22, 30]

**Размер**

list: 3 Mylist: 3

**toArray**

list: [10, 22, 30] Mylist: [10, 22, 30]

**toString**

list: [10, 22, 30] Mylist: [10, 22, 30]

## Вывод все функции работают

# Test2

//сравнение производительности листов

**test add list test add myList**

12099862 16779336

test get list

TestObject{name=Stas, testInt=1000, testDouble=2000.0}

time = 1526301

TestObject{name=Stas, testInt=99000, testDouble=198000.0}

time = 1765400

**test get myList**

TestObject{name=Stas, testInt=1000, testDouble=2000.0}

time = 184520

TestObject{name=Stas, testInt=99000, testDouble=198000.0}

time = 433039

**test size list**

100000 100000

time = 167618 time = 198926

## Вывод : добавление элементов с примерно одинаково, получение элементов примерно в 8-10 раз быстрее чем у стандартного, видимо деление пополам работает быстрее чем сдвиг по битам

# Test3

**test add array test add linked**

7318155 9281095

**test get array**

TestObject{name=Stas, testInt=1000, testDouble=2000.0}

time = 111653

TestObject{name=Stas, testInt=99000, testDouble=198000.0}

time = 210285

**test get linked**

TestObject{name=Stas, testInt=1000, testDouble=2000.0}

time = 75913

TestObject{name=Stas, testInt=99000, testDouble=198000.0}

time = 176485

**test size array test size linked**

100000 100000

time = 269022 time = 59290

## Вывод: ввод и вывод элементов примерно с одинаковой скоростью

# Test4

**test add HashSet test add linkedHashSet test add treeSet**

3640237 3902332 8513096

**test add HashSet**

3309156

t**est add linkedHashSet**

1002111

**test add treeSet**

4936859

## Вывод: TreeSet самый долгий так как он сортирует элементы

# Test5

**test add HashMap test add linkedHashMap test add treeMap**

1116354332 1138499964 607306

**test add HashMap В порядке убывания**

231516216

**test add linkedHashMap В порядке убывания**

248344314

**test add treeMap В порядке убывания**

153212

## Вывод : treeMap на несколько порядков быстрее

# Вывод

1. Было сложно разобраться с итератором, но вроде норм
2. Каким то образом у меня получилось сделать вывод элементов быстрее чем стандартный(хоть мелочь ,а приятно)
3. у Array и Linked примерно одинаковая скорость, так что особой разницы нет
4. treeSet самый долгий + ему нужно пере определять Comparable, поэтому treeSet полезен только когда нужно сортировка, **linkedHashSet и HashSet** примерно одинаковы
5. **treeMap** на несколько порядков оказался быстрее