

PM Lab-01

Создано системой Doxygen 1.9.4



|  |    |
|--|----|
| 1 Иерархический список классов         | 1  |
| 1.1 Иерархия классов                   | 1  |
| 2 Алфавитный указатель классов         | 3  |
| 2.1 Классы                             | 3  |
| 3 Список файлов                        | 5  |
| 3.1 Файлы                              | 5  |
| 4 Классы                               | 7  |
| 4.1 Класс <code>AutoInfo</code>        | 7  |
| 4.1.1 Подробное описание               | 8  |
| 4.1.2 Конструктор(ы)                   | 8  |
| 4.1.2.1 <code>AutoInfo()</code> [1/2]  | 8  |
| 4.1.2.2 <code>AutoInfo()</code> [2/2]  | 8  |
| 4.1.2.3 <code>~AutoInfo()</code>       | 8  |
| 4.1.3 Методы                           | 8  |
| 4.1.3.1 <code>operator"!=()</code>     | 9  |
| 4.1.3.2 <code>operator&lt;()</code>    | 9  |
| 4.1.3.3 <code>operator&lt;=()</code>   | 9  |
| 4.1.3.4 <code>operator==()</code>      | 10 |
| 4.1.3.5 <code>operator&gt;()</code>    | 10 |
| 4.1.3.6 <code>operator&gt;=()</code>   | 10 |
| 4.1.4 Данные класса                    | 11 |
| 4.1.4.1 <code>carBrand</code>          | 11 |
| 4.1.4.2 <code>color</code>             | 11 |
| 4.1.4.3 <code>licensePlate</code>      | 11 |
| 4.1.4.4 <code>ownerName</code>         | 11 |
| 4.1.4.5 <code>productionYear</code>    | 11 |
| 4.2 Класс <code>MergeSort</code>       | 12 |
| 4.2.1 Подробное описание               | 12 |
| 4.2.2 Конструктор(ы)                   | 12 |
| 4.2.2.1 <code>MergeSort()</code>       | 12 |
| 4.2.3 Методы                           | 12 |
| 4.2.3.1 <code>merge()</code>           | 12 |
| 4.2.3.2 <code>mergeSortHelper()</code> | 13 |
| 4.2.3.3 <code>sort()</code>            | 13 |
| 4.3 Класс <code>QuickSort</code>       | 14 |
| 4.3.1 Подробное описание               | 15 |
| 4.3.2 Конструктор(ы)                   | 15 |
| 4.3.2.1 <code>QuickSort()</code>       | 15 |
| 4.3.3 Методы                           | 15 |
| 4.3.3.1 <code>partition()</code>       | 15 |
| 4.3.3.2 <code>quickSortHelper()</code> | 16 |

---

|  |    |
|--|----|
| 4.3.3.3 sort()   | 16 |
| 4.4 Класс SelectionSort                                    | 17 |
| 4.4.1 Подробное описание                                   | 17 |
| 4.4.2 Конструктор(ы)                                       | 17 |
| 4.4.2.1 SelectionSort()                                    | 18 |
| 4.4.3 Методы   | 18 |
| 4.4.3.1 sort()   | 18 |
| 4.5 Класс SortMethod                                       | 18 |
| 4.5.1 Подробное описание                                   | 19 |
| 4.5.2 Конструктор(ы)                                       | 19 |
| 4.5.2.1 SortMethod()                                       | 19 |
| 4.5.3 Методы   | 19 |
| 4.5.3.1 sort()   | 19 |
| 4.5.4 Данные класса  | 20 |
| 4.5.4.1 name   | 20 |
| 5 Файлы  | 21 |
| 5.1 Файл /mnt/hgfs/D/HSE/МП/Lab-01/include/AutoInfo.hpp    | 21 |
| 5.1.1 Подробное описание                                   | 21 |
| 5.2 AutoInfo.hpp   | 22 |
| 5.3 Файл /mnt/hgfs/D/HSE/МП/Lab-01/include/SortMethods.hpp | 22 |
| 5.3.1 Подробное описание                                   | 22 |
| 5.4 SortMethods.hpp  | 23 |
| 5.5 Файл /mnt/hgfs/D/HSE/МП/Lab-01/src/AutoInfo.cpp        | 23 |
| 5.6 Файл /mnt/hgfs/D/HSE/МП/Lab-01/src/main.cpp            | 23 |
| 5.6.1 Подробное описание                                   | 24 |
| 5.6.2 Функции  | 24 |
| 5.6.2.1 main()   | 24 |
| 5.6.2.2 sort_iteration()                                   | 25 |
| 5.6.2.3 split()  | 25 |
| 5.7 Файл /mnt/hgfs/D/HSE/МП/Lab-01/src/SortMethods.cpp     | 25 |
| 5.7.1 Подробное описание                                   | 25 |
| Предметный указатель                                       | 27 |

# Глава 1

## Иерархический список классов

### 1.1 Иерархия классов

Иерархия классов.

|                         |    |
|-------------------------|----|
| AutoInfo . . . . .      | 7  |
| SortMethod . . . . .    | 18 |
| MergeSort . . . . .     | 12 |
| QuickSort . . . . .     | 14 |
| SelectionSort . . . . . | 17 |



## Глава 2

# Алфавитный указатель классов

### 2.1 Классы

Классы с их кратким описанием.

|                               |  |    |
|-------------------------------|--|----|
| <a href="#">AutoInfo</a>      | Stores information about automobiles . . . . .       | 7  |
| <a href="#">MergeSort</a>     | Implementation of merge sort algorithm . . . . .     | 12 |
| <a href="#">QuickSort</a>     | Implementation of quick sort algorithm . . . . .     | 14 |
| <a href="#">SelectionSort</a> | Implementation of selection sort algorithm . . . . . | 17 |
| <a href="#">SortMethod</a>    | Base abstract class for sorting algorithms . . . . . | 18 |





## Глава 3

# Список файлов

### 3.1 Файлы

Полный список файлов.

|  |    |
|--|----|
| <a href="#">/mnt/hgfs/D/HSE/МП/Lab-01/include/AutoInfo.hpp</a>                       |    |
| Class for storing automobile information . . . . .                                   | 21 |
| <a href="#">/mnt/hgfs/D/HSE/МП/Lab-01/include/SortMethods.hpp</a>                    |    |
| Collection of sorting algorithm implementations . . . . .                            | 22 |
| <a href="#">/mnt/hgfs/D/HSE/МП/Lab-01/src/AutoInfo.cpp</a>                           |    |
| Implementation of the <a href="#">AutoInfo</a> class methods . . . . .               | 23 |
| <a href="#">/mnt/hgfs/D/HSE/МП/Lab-01/src/main.cpp</a>                               |    |
| Program, that implements sorting algorithms and compares their performance . . . . . | 23 |
| <a href="#">/mnt/hgfs/D/HSE/МП/Lab-01/src/SortMethods.cpp</a>                        |    |
| Implementation of various sorting algorithms . . . . .                               | 25 |



## Глава 4

# Классы

### 4.1 Класс AutoInfo

Stores information about automobiles.

```
#include <AutoInfo.hpp>
```

#### Открытые члены

- `AutoInfo ()`  
Default constructor.
- `AutoInfo (const std::string &ownerName, const std::string &carBrand, int productionYear, const std::string &licensePlate, const std::string &color)`  
Parameterized constructor.
- `~AutoInfo ()`  
Destructor.
- `bool operator== (const AutoInfo &other) const`  
Equality comparison operator.
- `bool operator!= (const AutoInfo &other) const`  
Inequality comparison operator.
- `bool operator< (const AutoInfo &other) const`  
Less than comparison operator.
- `bool operator> (const AutoInfo &other) const`  
Greater than comparison operator.
- `bool operator<= (const AutoInfo &other) const`  
Less than or equal comparison operator.
- `bool operator>= (const AutoInfo &other) const`  
Greater than or equal comparison operator.

#### Открытые атрибуты

- `std::string ownerName`
- `std::string carBrand`
- `int productionYear`
- `std::string licensePlate`
- `std::string color`

### 4.1.1 Подробное описание

Stores information about automobiles.

This class encapsulates data about a car including owner information, car brand, production year, license plate, and color. It provides comparison operators for sorting purposes.

### 4.1.2 Конструктор(ы)

#### 4.1.2.1 `AutoInfo()` [1/2]

```
AutoInfo::AutoInfo ( )
```

Default constructor.

Creates an [AutoInfo](#) object with empty fields

#### 4.1.2.2 `AutoInfo()` [2/2]

```
AutoInfo::AutoInfo (
    const std::string & ownerName,
    const std::string & carBrand,
    int productionYear,
    const std::string & licensePlate,
    const std::string & color )
```

Parameterized constructor.

Аргументы

|                |                      |
|----------------|----------------------|
| ownerName      | Owner's full name    |
| carBrand       | Car brand/make       |
| productionYear | Production year      |
| licensePlate   | License plate number |
| color          | Car color            |

#### 4.1.2.3 `~AutoInfo()`

```
AutoInfo::~AutoInfo ( )
```

Destructor.

### 4.1.3 Методы

4.1.3.1 `operator!=()`

```
bool AutoInfo::operator!= (
    const AutoInfo & other ) const
```

Inequality comparison operator.

Аргументы

|       |   |
|-------|---|
| other | Another <a href="#">AutoInfo</a> object to compare with |
|-------|---|

Возвращает

true if objects are not equal, false otherwise

4.1.3.2 `operator<()`

```
bool AutoInfo::operator< (
    const AutoInfo & other ) const
```

Less than comparison operator.

Compares according to the following order: licensePlate, productionYear, carBrand, color, ownerName

Аргументы

|       |   |
|-------|---|
| other | Another <a href="#">AutoInfo</a> object to compare with |
|-------|---|

Возвращает

true if this object is less than other, false otherwise

4.1.3.3 `operator<=()`

```
bool AutoInfo::operator<= (
    const AutoInfo & other ) const
```

Less than or equal comparison operator.

Аргументы

|       |   |
|-------|---|
| other | Another <a href="#">AutoInfo</a> object to compare with |
|-------|---|

Возвращает

true if this object is less than or equal to other, false otherwise

#### 4.1.3.4 operator==()

```
bool AutoInfo::operator== (
    const AutoInfo & other ) const
```

Equality comparison operator.

Аргументы

|       |   |
|-------|---|
| other | Another <a href="#">AutoInfo</a> object to compare with |
|-------|---|

Возвращает

true if objects are equal, false otherwise

#### 4.1.3.5 operator>()

```
bool AutoInfo::operator> (
    const AutoInfo & other ) const
```

Greater than comparison operator.

Аргументы

|       |   |
|-------|---|
| other | Another <a href="#">AutoInfo</a> object to compare with |
|-------|---|

Возвращает

true if this object is greater than other, false otherwise

#### 4.1.3.6 operator>=()

```
bool AutoInfo::operator>= (
    const AutoInfo & other ) const
```

Greater than or equal comparison operator.

Аргументы

|       |   |
|-------|---|
| other | Another <a href="#">AutoInfo</a> object to compare with |
|-------|---|

Возвращает

true if this object is greater than or equal to other, false otherwise

#### 4.1.4 Данные класса

##### 4.1.4.1 `carBrand`

`std::string AutoInfo::carBrand`

##### 4.1.4.2 `color`

`std::string AutoInfo::color`

##### 4.1.4.3 `licensePlate`

`std::string AutoInfo::licensePlate`

##### 4.1.4.4 `ownerName`

`std::string AutoInfo::ownerName`

##### 4.1.4.5 `productionYear`

`int AutoInfo::productionYear`

Объявления и описания членов классов находятся в файлах:

- `/mnt/hgfs/D/HSE/МП/Lab-01/include/AutoInfo.hpp`
- `/mnt/hgfs/D/HSE/МП/Lab-01/src/AutoInfo.cpp`

## 4.2 Класс MergeSort

Implementation of merge sort algorithm.

```
#include <SortMethods.hpp>
```

Граф наследования: MergeSort:

Граф связей класса MergeSort:

### Открытые члены

- [MergeSort](#) ()  
Constructor Initializes the name of the sorting method.
- void [sort](#) (std::vector< [AutoInfo](#) > &arr) override  
Sorts an array using merge sort algorithm.

### Закрытые члены

- void [merge](#) (std::vector< [AutoInfo](#) > &arr, size\_t left, size\_t mid, size\_t right)  
Merges two sorted subarrays.
- void [mergeSortHelper](#) (std::vector< [AutoInfo](#) > &arr, size\_t left, size\_t right)  
Recursive helper function for merge sort.

### Дополнительные унаследованные члены

#### 4.2.1 Подробное описание

Implementation of merge sort algorithm.

Merge sort works by dividing the array into two halves, recursively sorting them, and then merging the sorted halves. Time complexity:  $O(n \log n)$  in all cases.

#### 4.2.2 Конструктор(ы)

##### 4.2.2.1 MergeSort()

```
MergeSort::MergeSort ( ) [inline]
```

Constructor Initializes the name of the sorting method.

#### 4.2.3 Методы

##### 4.2.3.1 merge()

```
void MergeSort::merge (  
    std::vector< AutoInfo > & arr,  
    size_t left,  
    size_t mid,  
    size_t right ) [private]
```

Merges two sorted subarrays.



## Аргументы

|       |                                  |
|-------|----------------------------------|
| arr   | Vector containing the subarrays  |
| left  | Starting index of first subarray |
| mid   | Ending index of first subarray   |
| right | Ending index of second subarray  |

Creates temporary arrays to store the two subarrays, and then merges them back into the original array in sorted order.

## Аргументы

|       |                                  |
|-------|----------------------------------|
| arr   | Vector containing the subarrays  |
| left  | Starting index of first subarray |
| mid   | Ending index of first subarray   |
| right | Ending index of second subarray  |

## 4.2.3.2 mergeSortHelper()

```
void MergeSort::mergeSortHelper (
    std::vector< AutoInfo > & arr,
    size_t left,
    size_t right ) [private]
```

Recursive helper function for merge sort.

## Аргументы

|       |                     |
|-------|---------------------|
| arr   | Vector to be sorted |
| left  | Starting index      |
| right | Ending index        |

Recursively divides the array into two halves, sorts them, and then merges them back.

## Аргументы

|       |                     |
|-------|---------------------|
| arr   | Vector to be sorted |
| left  | Starting index      |
| right | Ending index        |

## 4.2.3.3 sort()

```
void MergeSort::sort (
    std::vector< AutoInfo > & arr ) [override], [virtual]
```

Sorts an array using merge sort algorithm.

Main merge sort function.

Аргументы

|     |   |
|-----|---|
| arr | Vector of <a href="#">AutoInfo</a> objects to be sorted |
|-----|---|

Initiates the merge sort algorithm on the entire array.

Аргументы

|     |   |
|-----|---|
| arr | Vector of <a href="#">AutoInfo</a> objects to be sorted |
|-----|---|

Замещает [SortMethod](#).

Объявления и описания членов классов находятся в файлах:

- `/mnt/hgfs/D/HSE/МП/Lab-01/include/SortMethods.hpp`
- `/mnt/hgfs/D/HSE/МП/Lab-01/src/SortMethods.cpp`

## 4.3 Класс QuickSort

Implementation of quick sort algorithm.

```
#include <SortMethods.hpp>
```

Граф наследования: QuickSort:

Граф связей класса QuickSort:

Открытые члены

- [QuickSort](#) ()  
Constructor Initializes the name of the sorting method.
- void [sort](#) (std::vector< [AutoInfo](#) > &arr) override  
Sorts an array using quick sort algorithm.

Закрытые члены

- size\_t [partition](#) (std::vector< [AutoInfo](#) > &arr, size\_t low, size\_t high)  
Partitions the array around a pivot element.
- void [quickSortHelper](#) (std::vector< [AutoInfo](#) > &arr, size\_t low, size\_t high)  
Recursive helper function for quick sort.

## Дополнительные унаследованные члены

### 4.3.1 Подробное описание

Implementation of quick sort algorithm.

Quick sort works by selecting a 'pivot' element and partitioning the array around the pivot. Time complexity:  $O(n \log n)$  average case,  $O(n^2)$  worst case.

### 4.3.2 Конструктор(ы)

#### 4.3.2.1 QuickSort()

QuickSort::QuickSort ( ) [inline]

Constructor Initializes the name of the sorting method.

### 4.3.3 Методы

#### 4.3.3.1 partition()

```
size_t QuickSort::partition (
    std::vector< AutoInfo > & arr,
    size_t low,
    size_t high ) [private]
```

Partitions the array around a pivot element.

Partitions the array around a pivot.

Аргументы

|      |                          |
|------|--------------------------|
| arr  | Vector to be partitioned |
| low  | Starting index           |
| high | Ending index             |

Возвращает

Position of the pivot element after partitioning

Selects the last element as pivot and partitions the array such that all elements less than the pivot come before it, and all elements greater than the pivot come after it.

## Аргументы

|      |                          |
|------|--------------------------|
| arr  | Vector to be partitioned |
| low  | Starting index           |
| high | Ending index (pivot)     |

## Возвращает

Position of the pivot after partitioning

## 4.3.3.2 quickSortHelper()

```
void QuickSort::quickSortHelper (
    std::vector< AutoInfo > & arr,
    size_t low,
    size_t high ) [private]
```

Recursive helper function for quick sort.

Recursive helper function for quicksort.

## Аргументы

|      |                     |
|------|---------------------|
| arr  | Vector to be sorted |
| low  | Starting index      |
| high | Ending index        |

Recursively sorts the array by partitioning and then sorting each partition.

## Аргументы

|      |                     |
|------|---------------------|
| arr  | Vector to be sorted |
| low  | Starting index      |
| high | Ending index        |

## 4.3.3.3 sort()

```
void QuickSort::sort (
    std::vector< AutoInfo > & arr ) [override], [virtual]
```

Sorts an array using quick sort algorithm.

Main quick sort function.

Аргументы

|     |   |
|-----|---|
| arr | Vector of <a href="#">AutoInfo</a> objects to be sorted |
|-----|---|

Initiates the quicksort algorithm on the entire array.

Аргументы

|     |   |
|-----|---|
| arr | Vector of <a href="#">AutoInfo</a> objects to be sorted |
|-----|---|

Замещает [SortMethod](#).

Объявления и описания членов классов находятся в файлах:

- `/mnt/hgfs/D/HSE/МП/Lab-01/include/SortMethods.hpp`
- `/mnt/hgfs/D/HSE/МП/Lab-01/src/SortMethods.cpp`

## 4.4 Класс SelectionSort

Implementation of selection sort algorithm.

```
#include <SortMethods.hpp>
```

Граф наследования: SelectionSort:

Граф связей класса SelectionSort:

Открытые члены

- [SelectionSort](#) ()  
Constructor Initializes the name of the sorting method.
- void [sort](#) (std::vector< [AutoInfo](#) > &arr) override  
Sorts an array using selection sort algorithm.

Дополнительные унаследованные члены

### 4.4.1 Подробное описание

Implementation of selection sort algorithm.

Selection sort works by repeatedly finding the minimum element from the unsorted part of the array and putting it at the beginning. Time complexity:  $O(n^2)$  in all cases.

### 4.4.2 Конструктор(ы)

#### 4.4.2.1 SelectionSort()

```
SelectionSort::SelectionSort ( ) [inline]
```

Constructor Initializes the name of the sorting method.

#### 4.4.3 Методы

##### 4.4.3.1 sort()

```
void SelectionSort::sort (
    std::vector< AutoInfo > & arr ) [override], [virtual]
```

Sorts an array using selection sort algorithm.

Implementation of selection sort algorithm.

Аргументы

|     |   |
|-----|---|
| arr | Vector of <a href="#">AutoInfo</a> objects to be sorted |
|-----|---|

This method sorts a vector of [AutoInfo](#) objects using selection sort. For each position, it finds the minimum element in the remaining unsorted portion and swaps it with the element at the current position.

Аргументы

|     |   |
|-----|---|
| arr | Vector of <a href="#">AutoInfo</a> objects to be sorted |
|-----|---|

Замещает [SortMethod](#).

Объявления и описания членов классов находятся в файлах:

- `/mnt/hgfs/D/HSE/МП/Lab-01/include/SortMethods.hpp`
- `/mnt/hgfs/D/HSE/МП/Lab-01/src/SortMethods.cpp`

## 4.5 Класс SortMethod

Base abstract class for sorting algorithms.

```
#include <SortMethods.hpp>
```

Граф наследования:SortMethod:

## Открытые члены

- `SortMethod` (`const std::string &method_name`)  
Constructor with algorithm name.
- virtual void `sort` (`std::vector< AutoInfo > &arr`)=0  
Pure virtual function for sorting.

## Открытые атрибуты

- `std::string name`

### 4.5.1 Подробное описание

Base abstract class for sorting algorithms.

Defines the interface for all sorting algorithm implementations. Each derived class must implement the sort method.

### 4.5.2 Конструктор(ы)

#### 4.5.2.1 SortMethod()

```
SortMethod::SortMethod (  
    const std::string & method_name )    [inline]
```

Constructor with algorithm name.

Аргументы

|                          |                               |
|--------------------------|-------------------------------|
| <code>method_name</code> | Name of the sorting algorithm |
|--------------------------|-------------------------------|

### 4.5.3 Методы

#### 4.5.3.1 sort()

```
virtual void SortMethod::sort (  
    std::vector< AutoInfo > & arr )    [pure virtual]
```

Pure virtual function for sorting.

Аргументы

|     |   |
|-----|---|
| arr | Vector of <a href="#">AutoInfo</a> objects to be sorted |
|-----|---|

Замещается в [SelectionSort](#), [QuickSort](#) и [MergeSort](#).

#### 4.5.4 Данные класса

##### 4.5.4.1 name

`std::string SortMethod::name`

Name of the sorting algorithm

Объявления и описания членов класса находятся в файле:

- `/mnt/hgfs/D/HSE/МП/Lab-01/include/SortMethods.hpp`



## Глава 5

# Файлы

### 5.1 Файл /mnt/hgfs/D/HSE/МП/Lab-01/include/AutoInfo.hpp

Class for storing automobile information.

```
#include <string>
```

Граф включаемых заголовочных файлов для AutoInfo.hpp: Граф файлов, в которые включается этот файл:

#### Классы

- class [AutoInfo](#)  
Stores information about automobiles.

#### 5.1.1 Подробное описание

Class for storing automobile information.

Автор

Lab-01

Дата

2023

## 5.2 AutoInfo.hpp

См. документацию.

```

1 #pragma once
2 #include <string>
3
19 class AutoInfo {
20 public:
21     std::string ownerName;    // Owner's full name
22     std::string carBrand;    // Car brand/make
23     int productionYear;    // Production year
24     std::string licensePlate; // License plate number
25     std::string color;    // Car color
26
32     AutoInfo();
33
43     AutoInfo(const std::string& ownerName,
44             const std::string& carBrand,
45             int productionYear,
46             const std::string& licensePlate,
47             const std::string& color);
48
52     ~AutoInfo();
53
60     bool operator==(const AutoInfo& other) const;
61
68     bool operator!=(const AutoInfo& other) const;
69
79     bool operator<(const AutoInfo& other) const;
80
87     bool operator>(const AutoInfo& other) const;
88
95     bool operator<=(const AutoInfo& other) const;
96
103     bool operator>=(const AutoInfo& other) const;
104 };

```

## 5.3 Файл /mnt/hgfs/D/HSE/МП/Lab-01/include/SortMethods.hpp

Collection of sorting algorithm implementations.

```

#include "AutoInfo.hpp"
#include <vector>

```

Граф включаемых заголовочных файлов для SortMethods.hpp: Граф файлов, в которые включается этот файл:

### Классы

- class [SortMethod](#)  
Base abstract class for sorting algorithms.
- class [SelectionSort](#)  
Implementation of selection sort algorithm.
- class [QuickSort](#)  
Implementation of quick sort algorithm.
- class [MergeSort](#)  
Implementation of merge sort algorithm.

### 5.3.1 Подробное описание

Collection of sorting algorithm implementations.

Автор

Lab-01

Дата

2023

## 5.4 SortMethods.hpp

См. документацию.

```

1 #pragma once
2 #include "AutoInfo.hpp"
3 #include <vector>
4
19 class SortMethod {
20 public:
21     std::string name;
27     SortMethod(const std::string& method_name) : name(method_name) {}
28
33     virtual void sort(std::vector<AutoInfo>& arr) = 0;
34 };
35
44 class SelectionSort : public SortMethod {
45 public:
50     SelectionSort() : SortMethod("SelectionSort") {}
51
56     void sort(std::vector<AutoInfo>& arr) override;
57 };
58
67 class QuickSort : public SortMethod {
68 private:
76     size_t partition(std::vector<AutoInfo>& arr, size_t low, size_t high);
77
84     void quickSortHelper(std::vector<AutoInfo>& arr, size_t low, size_t high);
85
86 public:
91     QuickSort() : SortMethod("QuickSort") {}
92
97     void sort(std::vector<AutoInfo>& arr) override;
98 };
99
108 class MergeSort : public SortMethod {
109 private:
117     void merge(std::vector<AutoInfo>& arr, size_t left, size_t mid, size_t right);
118
125     void mergeSortHelper(std::vector<AutoInfo>& arr, size_t left, size_t right);
126
127 public:
132     MergeSort() : SortMethod("MergeSort") {}
133
138     void sort(std::vector<AutoInfo>& arr) override;
139 };

```

## 5.5 Файл /mnt/hgfs/D/HSE/МП/Lab-01/src/AutoInfo.cpp

Implementation of the [AutoInfo](#) class methods.

```
#include "AutoInfo.hpp"
```

Граф включаемых заголовочных файлов для AutoInfo.cpp:

## 5.6 Файл /mnt/hgfs/D/HSE/МП/Lab-01/src/main.cpp

Program, that implements sorting algorithms and compares their performance.

```

#include <iostream>
#include <fstream>
#include <string>
#include <chrono>
#include <vector>
#include <algorithm>
#include <iomanip>
#include "AutoInfo.hpp"
#include "SortMethods.hpp"

```

Граф включаемых заголовочных файлов для main.cpp:

## Функции

- `std::vector< std::string > * split` (`const std::string &str`, `const std::string &delimiter`)  
Splits a string into tokens based on a delimiter.
- `void sort_iteration` (`const std::vector< AutoInfo > *autos`, `SortMethod *current_sort`)  
Executes a sorting algorithm and measures its performance.
- `int main` (`int argc`, `char *argv[]`)  
Main function.

### 5.6.1 Подробное описание

Program, that implements sorting algorithms and compares their performance.

Автор

Vagin Anton

Дата

2025

### 5.6.2 Функции

#### 5.6.2.1 main()

```
int main (
    int argc,
    char * argv[] )
```

Main function.

Reads automobile data from a CSV file, applies different sorting algorithms, measures their performance, and writes the sorted data to an output file.

Аргументы

|                   |                                  |
|-------------------|----------------------------------|
| <code>argc</code> | Number of command-line arguments |
| <code>argv</code> | Array of command-line arguments  |

Возвращает

0 if successful, 1 if an error occurred

## 5.6.2.2 sort\_iteration()

```
void sort_iteration (
    const std::vector< AutoInfo > * autos,
    SortMethod * current_sort )
```

Executes a sorting algorithm and measures its performance.

Creates a copy of the input array, runs the specified sorting algorithm, measures the execution time, and outputs the results.

## Аргументы

|              |  |
|--------------|--|
| autos        | Pointer to the vector of <a href="#">AutoInfo</a> objects to be sorted |
| current_sort | Pointer to the sorting algorithm to be used                            |

## 5.6.2.3 split()

```
std::vector< std::string > * split (
    const std::string & str,
    const std::string & delimiter )
```

Splits a string into tokens based on a delimiter.

## Аргументы

|           |                              |
|-----------|------------------------------|
| str       | The input string to be split |
| delimiter | The delimiter string         |

## Возвращает

Vector of string tokens

## 5.7 Файл /mnt/hgfs/D/HSE/МП/Lab-01/src/SortMethods.cpp

Implementation of various sorting algorithms.

```
#include "SortMethods.hpp"
#include <algorithm>
```

Граф включаемых заголовочных файлов для SortMethods.cpp:

## 5.7.1 Подробное описание

Implementation of various sorting algorithms.



# Предметный указатель

`/mnt/hgfs/D/HSE/МП/Lab-01/include/AutoInfo.hpp` [name](#)  
[21, 22](#) [SortMethod, 20](#)  
`/mnt/hgfs/D/HSE/МП/Lab-01/include/SortMethods.hpp`,  
[22, 23](#) [operator!=](#)  
`/mnt/hgfs/D/HSE/МП/Lab-01/src/AutoInfo.cpp`, [AutoInfo, 8](#)  
[23](#) [operator<](#)  
`/mnt/hgfs/D/HSE/МП/Lab-01/src/SortMethods.cpp`, [AutoInfo, 9](#)  
[25](#) [operator<=](#)  
`/mnt/hgfs/D/HSE/МП/Lab-01/src/main.cpp`, [23](#) [AutoInfo, 9](#)  
`~AutoInfo` [operator>](#)  
[AutoInfo, 8](#) [AutoInfo, 10](#)  
[operator>=](#)  
[AutoInfo, 10](#)  
[operator==](#)  
[AutoInfo, 10](#)  
[ownerName](#)  
[AutoInfo, 11](#)  
[partition](#)  
[QuickSort, 15](#)  
[productionYear](#)  
[AutoInfo, 11](#)  
[QuickSort, 14](#)  
[partition, 15](#)  
[QuickSort, 15](#)  
[quickSortHelper, 16](#)  
[sort, 16](#)  
[quickSortHelper](#)  
[QuickSort, 16](#)  
[SelectionSort, 17](#)  
[SelectionSort, 17](#)  
[sort, 18](#)  
[sort](#)  
[MergeSort, 13](#)  
[QuickSort, 16](#)  
[SelectionSort, 18](#)  
[SortMethod, 19](#)  
[sort\\_iteration](#)  
[main.cpp, 24](#)  
[SortMethod, 18](#)  
[name, 20](#)  
[sort, 19](#)  
[SortMethod, 19](#)  
[split](#)  
[main.cpp, 25](#)  
`/mnt/hgfs/D/HSE/МП/Lab-01/include/AutoInfo.hpp` [name](#)  
[21, 22](#) [SortMethod, 20](#)  
`/mnt/hgfs/D/HSE/МП/Lab-01/include/SortMethods.hpp`,  
[22, 23](#) [operator!=](#)  
`/mnt/hgfs/D/HSE/МП/Lab-01/src/AutoInfo.cpp`, [AutoInfo, 8](#)  
[23](#) [operator<](#)  
`/mnt/hgfs/D/HSE/МП/Lab-01/src/SortMethods.cpp`, [AutoInfo, 9](#)  
[25](#) [operator<=](#)  
`/mnt/hgfs/D/HSE/МП/Lab-01/src/main.cpp`, [23](#) [AutoInfo, 9](#)  
`~AutoInfo` [operator>](#)  
[AutoInfo, 8](#) [AutoInfo, 10](#)  
[operator>=](#)  
[AutoInfo, 10](#)  
[operator==](#)  
[AutoInfo, 10](#)  
[ownerName](#)  
[AutoInfo, 11](#)  
[partition](#)  
[QuickSort, 15](#)  
[productionYear](#)  
[AutoInfo, 11](#)  
[QuickSort, 14](#)  
[partition, 15](#)  
[QuickSort, 15](#)  
[quickSortHelper, 16](#)  
[sort, 16](#)  
[quickSortHelper](#)  
[QuickSort, 16](#)  
[SelectionSort, 17](#)  
[SelectionSort, 17](#)  
[sort, 18](#)  
[sort](#)  
[MergeSort, 13](#)  
[QuickSort, 16](#)  
[SelectionSort, 18](#)  
[SortMethod, 19](#)  
[sort\\_iteration](#)  
[main.cpp, 24](#)  
[SortMethod, 18](#)  
[name, 20](#)  
[sort, 19](#)  
[SortMethod, 19](#)  
[split](#)  
[main.cpp, 25](#)  
`AutoInfo, 7`  
`~AutoInfo, 8`  
`AutoInfo, 8`  
`carBrand, 11`  
`color, 11`  
`licensePlate, 11`  
`operator!=, 8`  
`operator<, 9`  
`operator<=, 9`  
`operator>, 10`  
`operator>=, 10`  
`operator==, 10`  
`ownerName, 11`  
`productionYear, 11`  
`carBrand`  
[AutoInfo, 11](#)  
`color`  
[AutoInfo, 11](#)  
`licensePlate`  
[AutoInfo, 11](#)  
`main`  
[main.cpp, 24](#)  
`main.cpp`  
[main, 24](#)  
[sort\\_iteration, 24](#)  
[split, 25](#)  
`merge`  
[MergeSort, 12](#)  
`MergeSort, 12`  
[merge, 12](#)  
[MergeSort, 12](#)  
[mergeSortHelper, 13](#)  
[sort, 13](#)  
`mergeSortHelper`  
[MergeSort, 13](#)