Отчет по второй лабораторной работе

Этот модуль содержит функции для генерации комбинаций букв, создания массива объектов и сравнения времени поиска различных структур данных.

Функции

`generate_combinations()`

Эта функция генерирует все возможные комбинации из двух строчных букв английского алфавита.

Возвращает:

- `combinations` (list): Список всех возможных комбинаций из двух букв.

Алгоритм:

1. Получает строку из всех строчных букв английского алфавита.
2. Перебирает каждую букву дважды для создания комбинаций.
3. Добавляет каждую комбинацию в список.

`generate_objects_array(size)`

Эта функция генерирует массив объектов заданного размера, где каждый объект имеет случайный ключ и случайное значение.

Аргументы:

- `size` (int): Размер массива объектов.

Возвращает:

- `objects` (list): Массив объектов, где каждый объект имеет ключ и значение.

Алгоритм:

1. Генерирует все возможные комбинации из двух букв.
2. Создает массив объектов с случайным ключом и случайным значением.

Импортируемые модули

Структуры данных и функции:

- `BinarySearchTree` из `algos.bin_tree`
- `HashTable` из `algos.hash`
- `RedBlackTree` из `algos.red_black_tree`
- `MultiMap` из `algos.multimap`
- `generate_objects_array` и `generate_combinations` из `data.gen`

Логирование и графика:

- `logger` из `loguru`
- `time` и `random` для измерения времени и генерации случайных данных
- `matplotlib.pyplot` для построения графиков

Сравнение времени поиска

`compare_search_time(sizes)`

Эта функция сравнивает время поиска для различных структур данных при различных размерах массивов.

Аргументы:

- `sizes` (list): Список размеров массивов.

Алгоритм:

1. Генерирует ключи и случайные данные для каждого размера массива.
2. Вставляет данные в структуры данных и измеряет время поиска для случайного ключа.
3. Логирует и сохраняет время выполнения для каждой структуры данных.
4. Строит график времени поиска в зависимости от размера массива.

Пример использования:

```
sizes = [100, 1000, 5000, 10000, 50000, 75000, 100000]
compare_search_time(sizes)
```

Графики

Для визуализации результатов построены графики:

1. **Время поиска**:
    - Отображает время поиска для каждого размера массива для различных структур данных: бинарное дерево, красно-черное дерево, хеш-таблица и мультимап.

```
plt.plot(sizes, binary_tree_time, label="Binary Tree")
plt.plot(sizes, red_black_tree_time, label="Red Black Tree")
plt.plot(sizes, hashtable_time, label="Hash Table")
plt.plot(sizes, multimap_time, label="Multimap Table")
plt.xlabel("Array Size")
plt.ylabel("Search Time")
plt.legend()
plt.savefig("plot.png")
plt.show()
```

2. **Коллизии в хеш-таблице**:
    - Отображает количество коллизий в хеш-таблице в зависимости от размера массива.

```
from algos.hash import count_collisions

sizes = [100, 1000, 5000, 10000, 50000, 75000, 100000]
collisions = []

for size in sizes:
    col = count_collisions(size)
    collisions.append(col)

plt.plot(sizes, collisions, label="Collisions")
plt.xlabel("Array Size")
plt.ylabel("Collisions")
plt.legend()
plt.savefig("test.png")
plt.show()
```

Заключение

В данном отчете представлен код для генерации комбинаций букв, создания массива объектов и сравнения времени поиска различных структур данных: бинарное дерево, красно-черное дерево, хеш-таблица и мультимап. Было проведено измерение времени поиска для различных размеров массивов и визуализация результатов на графиках.