**Практическая работа №3**

**Тема: «Хэш-таблицы».**

**Цель работы: изучить реализацию хэш-таблиц.**

Хеш-таблица — это структура данных, реализующая интерфейс ассоциативного массива, а именно, она позволяет хранить пары (ключ, значение) и выполнять три операции: операцию добавления новой пары, операцию поиска и операцию удаления пары по ключу.

Создадим хэш-таблицу с реализацией метода открытой адрессации для простейшего телефона справочника. Для этого определим структуру контакта, которая представлена на рис. 1.

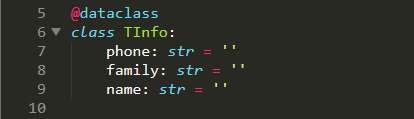


Рис. 1. Структура контакта.

Для одной ячейки таблицы определим следующую структуру, представленную на рис. 2.

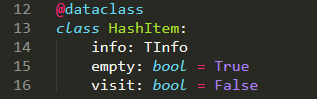


Рис. 2. Структура ячейки таблицы.

Где empty - флаг, указывающий, что ячейка свободна, в независимости от содержащихся там данных.

visit - флаг, указывающий, что ячейка просматривалась.

Для вычисления значения хэша будем использовать следующую функцию, представленную на рис. 3.

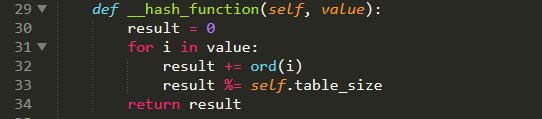


Рис. 3. Хэш-функция.

Диаграмма деятельности для этой функции представлена на рис. 4.

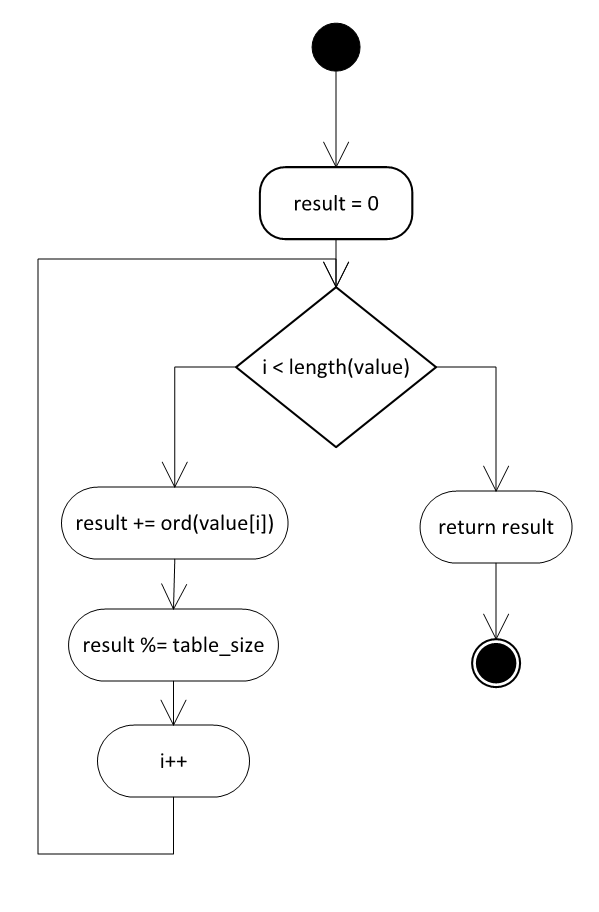


Рис. 4. Диаграмма деятельности для \_hash\_function.

Функция добавления элемента представлена на рис. 5.

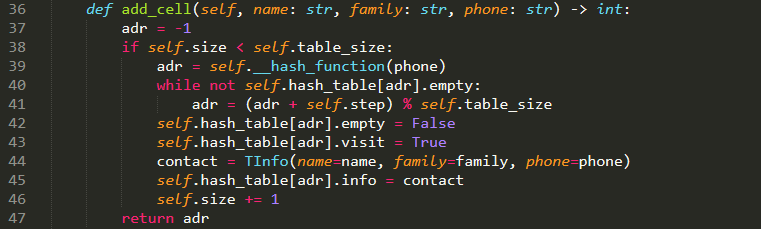


Рис. 5. Добавление элемента в хэш-таблицу.

Диаграмма деятельности для добавления элемента представлена на рис. 6.

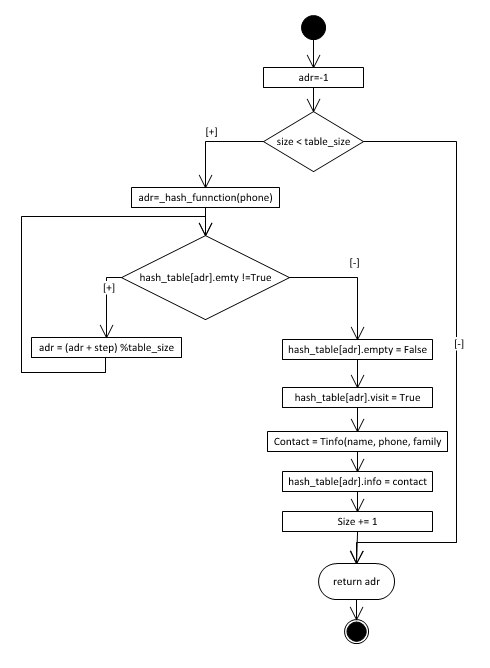


Рис. 6. Диаграма деятельности для добавления элемента, в таблицу методом открытой адрессации.

Для поиска элемента, надо убедиться, что флаги visit каждой ячейки сброшены к дефолтным значениям. Для этого мы используем функциию, код которой представлен на рис. 7.



Рис. 7. Сброс значений к дефолтным.

Диаграмма деятельности представлена для нее на рис. 8.

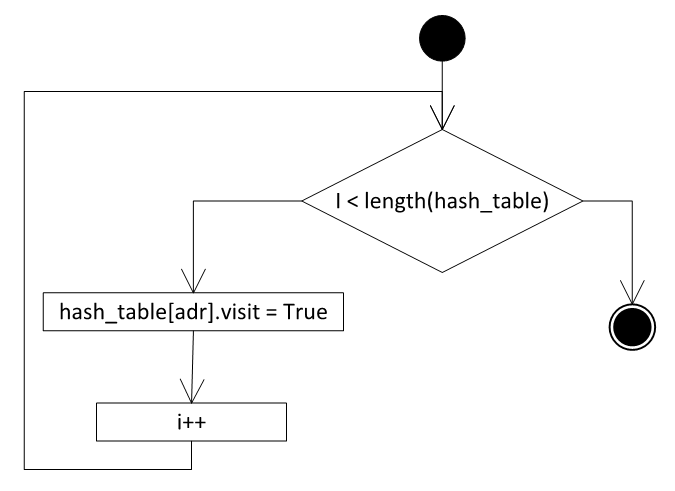


Рис. 8. Сброс флагов visit к дефолтным значениям.

Функция поиска значения в таблице представлена на рис. 9.

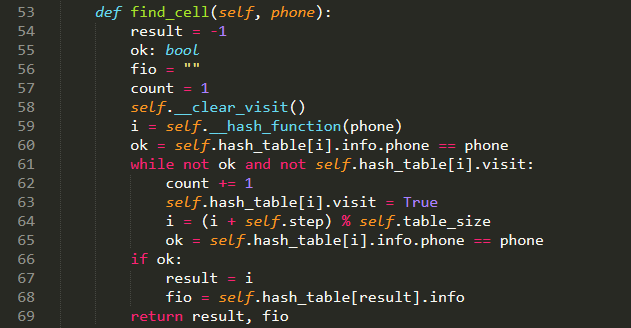


Рис. 9. Поиск элемента в таблице.

Диаграма деятельности для поиска элемента представлена на рисунке 10.

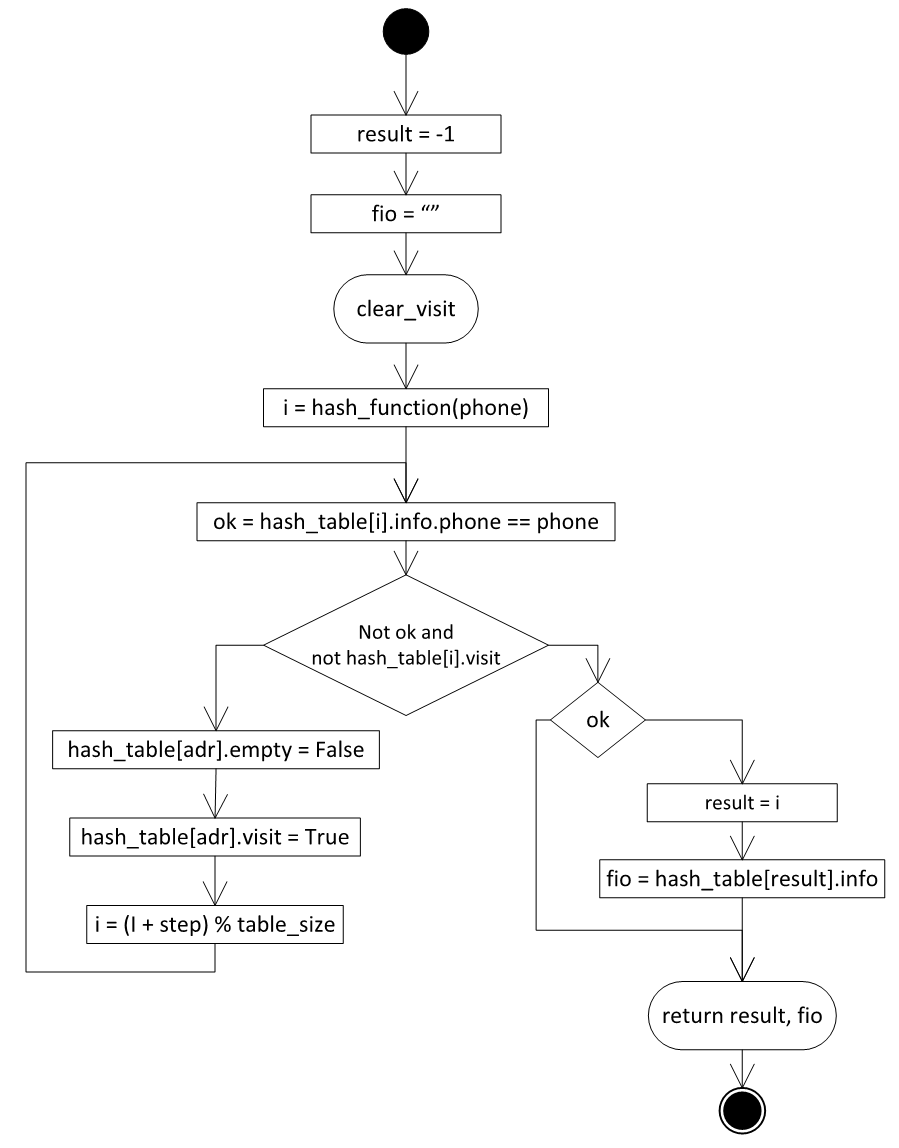


Рис. 10. Поиск элемента в хэш-таблице с открытой адресацией.

Для удаления элемента реализован метод, код которого представлен на рис. 11. Действие кода сводится к нахождению нужного элемента и выставление флага empty в позицию True.

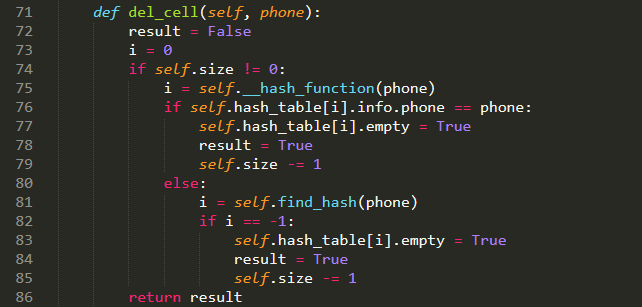


Рис. 11. Удаление элемента.

Диаграмма деятельности для удаления элемента представлена на рис. 12.

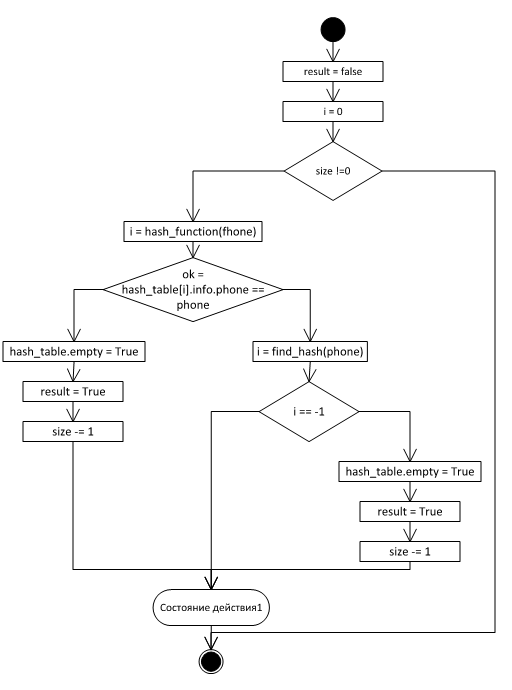


Рис. 12. Удаление элемента из хэш-таблицы.

Так же реализуем хэш-таблиц по методу цепочек. Для этого определим классы данных, как на рис. 13.



Рис. 13. Классы данных для метода цепочек.

Реализацию функции для хэширования оставим без измененений.

Изменим функцию добавления нового значения (рис. 14) и ее диаграмма деятельности представлена на рис. 15.

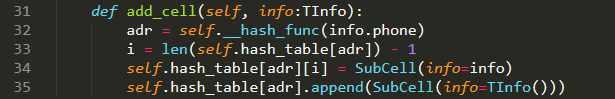


Рис. 14. Функция добавления новой записи в таблицу.

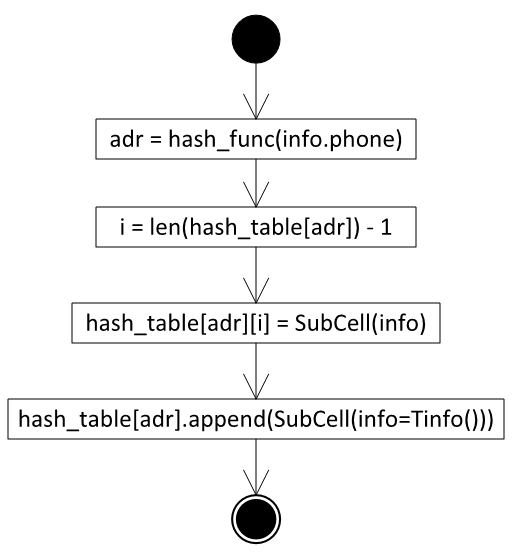


Рис. 15. Добавление нового элемента в таблицу.

Функция удаления элемента представлена на рис. 16. Диаграмма деятельности для нее представлена на рис. 17.

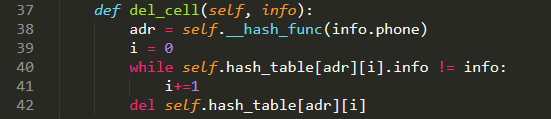


Рис. 16. Удаление элемента.

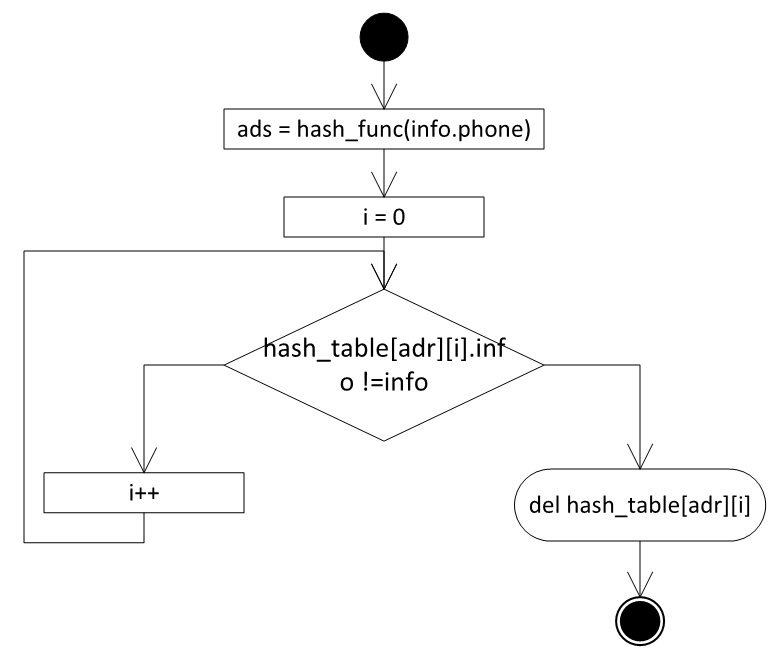


Рис. 17. Удаление элемента.

Функция поиска элемента представлена на рис. 18. Диаграмма деятельности на рис. 19.

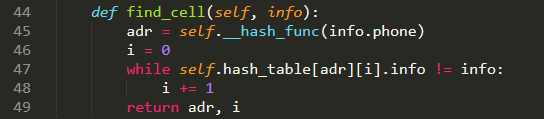


Рис. 18. Функция поиска элемента.

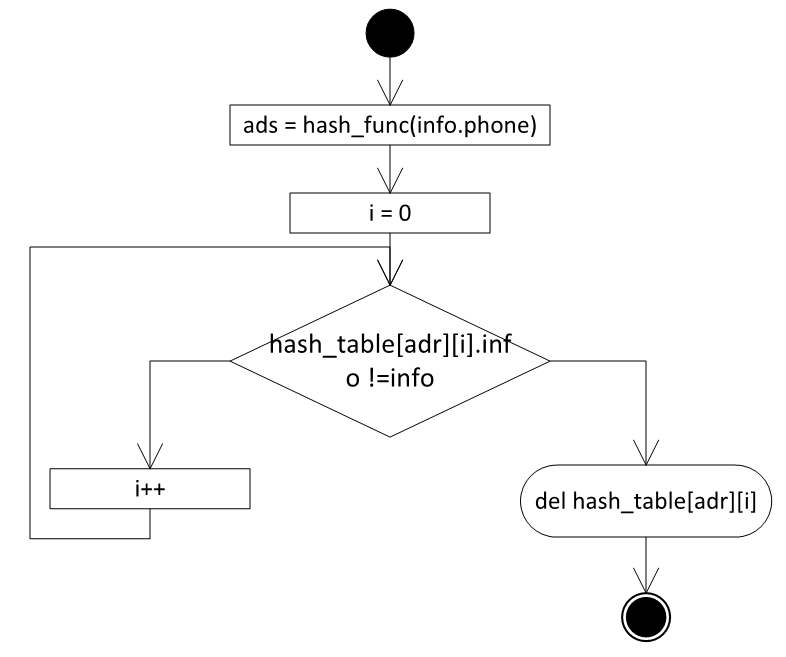


Рис. 19. Поиск элемента в хэш-таблице.

Вывод: в ходе выполнения практической работы были изучены хэш- таблицы и методы их реализации на языке Python.