# Звіт

Оцінки складності:

1. Обчислення хешу – O(k), k – довжина рядку. Так як згідно з умовою лабораторної роботи k <= 15, то обчислення хешу має часову складність O(1)
2. Пошук елементу в множині: O(1) в середньому випадку, O(n), n – кількість рядків в множині в найгіршому випадку (наприклад, при зламі). Така часова складність через імплементацію комірок таблиці як Linked List
3. Додавання елементу в множину: O(1) в середньому випадку, O(n), n – кількість рядків в множині в найгіршому випадку (наприклад, при зламі). Використовується пошук елементу в множині, далі рядок записується як перший елемент комірки
4. Видалення елементу в множині: O(1) в середньому випадку, O(n), n – кількість рядків в множині в найгіршому випадку (наприклад, при зламі). Використовується пошук елементу в множині, якщо елемент є в множині, то прибираємо його з відповідної комірки
5. Пошук паліндромів: Проходимо по всьому масиву комірок множини (в імплементації - 10^6 комірок) позначимо загальну кількість комірок – M, alpha \* M = n, де alpha – коефіцієнт наповненості. Для кожної непорожньої комірки запускаємо процедуру пошуку паліндромів:
   1. Знаходимо поліноміальні хеші рядку та оберненого йому рядку O(k)
   2. Почергово обираємо символи як центри паліндромів O(k)
      1. Бінарним пошуком по радіусу навколо символу знаходимо найдовший паліндром з центром в символі. Спочатку для паліндромів з непарною кількістю символів, потім з парною O(log(k))
      2. Записуємо в список паліндромів отримані паліндроми O(1)

Отже, часова складність пошуку паліндромів: O(n) \* (O(k) + O(k) \* O(log(k)) \* O(1) ) = O(n\*k\*log(k)). Так як k <= 15 за умовою, то маємо O(n\*15\*log(15)) = O(17.64 \* n) = O(n). Отже, алгоритм лінійний по кількості заповнених клітинок в таблиці. Це підтверджується наступним графіком:

