Сортировка Шелла. Время работы (б/д)

ceagest

1 Сортировка Шелла

- Вспомним алгоритм сортировки вставками.
- Там мы искали для элемента позицию для вставки, просматривая элементы подряд, и выполняли обмены.
- К сожалению, один обмен может убрать не более 1 инверсии.
- Если увеличить шаг, то число возможных убираемых инверсий может быть, очевидно, больше, чем 1 (если повезет может быть много).
- Почему бы не попробовать начать с большого шага, а затем его уменьшать до 1?

1.1 Время работы сортировки Шелла и её реализация

- Шелл предлагал делать шаги размеров $\frac{n}{2}, \frac{n}{4}, ..., 1$. В итоге это давало сложность $\mathcal{O}(n^2)$.
- Кнут, в свою очередь, предложил последовательность, дающую сложность в среднем случае $\mathcal{O}(n^{\frac{4}{3}})$, в худшем $\mathcal{O}(n^{\frac{3}{2}})$.
- Также известно огромное множество других последовательностей.
- Собственно последовательность, предложенная Кнутом: $t_0 = 1, t_i = 3t_{i-1} + 1$.

Вариант реализации сортировки Шелла на Java

```java
public class ShellSort {
    public static void shellSort(int[] a, int n) {
        int step = 1;
        while (step <= n / 9) {step = step * 3 + 1;}
        for (; step > 0; step /= 3) {
            for (int i = step; i < n; i += step) {
                int j = i;
                int tmp = a[i];
                for (; j >= step && a[j - step] > tmp; j -= step) {
                    a[j] = a[j - step];
                }
                a[j] = tmp;
            }
        }
    }
}
```