|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Массив | Показатель | 1000 элементов | | |
|  |  | Insert | Select | Bubble |
| Упорядоченный | C | 499500 | 499500 | 999 |
| M | 0 | 0 | 0 |
| Обратно упорядоченный | C | 499500 | 499500 | 499500 |
| M | 499500 | 250000 | 499500 |
| Случайный | Mean(C) | 499500 | **499500** | 496365,4 |
| Mean(M) | 249939 | **5265,02** | 249789,1 |

Сравнительная таблица эффективности алгоритмов (Таблица 2)

1)

3) Полученные экспериментальные значения в таблице 2 с расчетными значениями приблизительно одинаковы.

4) Количество сравнений и перестановок, которые выполняет алгоритм вместе со средой, в которой выполняется код, являются ключевыми определяющими факторами производительности. Сортировка вставками наиболее эффективна, когда массив уже частично отсортирован и когда элементов массива не много. Если элементов меньше 10, то данный алгоритм является лучшим. Пузырьковая сортировка является неэффективной, поскольку она должна переставлять элементы до того, как станет известна их окончательная позиция. Эти “пустые” операции обмена затратны. Для случайных значений лучшим выбором сортировки является Select. (вывод из табл. 2), поскольку в выборе заранее известно, в какое место поставим элемент.