**Задание**

Вариант – 270

Номер задания – 4

Номер функции – 20

**Задание:**

*Обобщенный артефакт, используемый в задании:*

* Объемная геометрическая фигура.

*Базовые альтернативы (уникальные параметры, задающие отличительные признаки альтернатив):*

1. Шар (целочисленный радиус)
2. Параллелепипед (три целочисленных ребра).
3. Правильный тетраэдр (длина ребра - целое).

*Общие для всех альтернатив переменные:*

* Плотность материала фигуры.

*Общие для всех альтернатив функции:*

* Вычисление площади поверхности.

**Функция:**

Удалить из контейнера те элементы, для которых значение,полученное с использованием функции, общей для всех альтернатив, больше,чем среднее арифметическое для всех элементов контейнера, полученное использованием этой же функции. Остальные элементы передвинуть вначало контейнера с сохранением порядка.

**Запуск программы**

Для запуска программы, используя текстовый документ с именем inputfileName необхожимо написать**./<имя програмы> -f <inputFileName> <outputFileName1> <outputFileName1>**, а для генерации n фигур необходимо написать **./<имя програмы> -n <n> <outputFileName1> <outputFileName1>.**

**Имя программы в папке – task.**

**Основные характеристики**

1. Модулей реализации – 7 штук
2. Общий размер исходных файлов – 32 кб
3. Строк кода - 564
4. Размер исполняемого файла – 36 кб
5. Размер тестов – 184 кб
6. Общий размер папки – 2.81 мб

**Тесты**

Тесты находятся в папке tests и имеют вид *testN.txt*, где N - номер теста.

Вывод элементов каждого теста находится в папке out:

* файл с элементами теста имеет вид *out\_N\_elements.txt*, где N - номер теста.
* файл с результатом теста и временем выполнения программы имеет вид *out\_N\_result.txt*, где N - номер теста.

**Результаты тестов**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Программа на уровне системы набора команд | Программа с динамической типизацией | Программа в объектно-ориентированном стиле |
| 5 элементов | 0.000001 s | 0.0029613 s | 0.000325 s |
| 100 элементов | 0.000003 s | 0.020966 s | 0.000516 s |
| 1000 элементов | 0.000021 s | 0.012965 s | 0.002021 s |
| 10000 элементов | 0.00022 s | 0.266969 s | 0.000325 s |
| 5000 элементов | 0.000099 s | 0.08299 s | 0.000121 s |

**Итоги**

Как можно увидеть по результатам тестов, программа на уровне систем команд работает значительно быстрее как программы с динамической типизацией, так и с программой в объектно-ориентированном стиле. Но как в программе с динамической типизацией мы жертвовали быстродействием программы, для того чтобы сам код был небольшим и читабельным, так в программе на уровне набора команд мы жертвуем читабельностью и размером кода ради огромной скорости выполнения программы.