

# Пояснительная записка

| Алиева Рената Эдуардовна ФКН ПИ ВШЭ БПИ202.

| Вариант 42. Артефакт номер 14. Функция номер 3.

## Описание полученного задания:

Данная программа описывает вычисление максимального расстояния, которое может пройти автомобиль(условие задачи 14) а также выполняет сортировку контейнера пузырьком (Bubble Sort) .В качестве ключей для сортировки и других действий используются результаты функции, вычисляющей максимальное расстояние(обработка данных в контейнере под номером 3). Программа выполнена на низкоуровневом языке Assembly.В программе реализованы такие функции следующих сущностей, как автомобильный транспорт(car), грузовик(truck) – содержит грузоподъемность; емкость топливного бака; расход топлива, автобус(bus) – содержит пассажировместимость; емкость топливного бака; расход топлива, легковой автомобиль(automobile) – содержит максимальную скорость; емкость топливного бака; расход топлива.

## Работа программы:

Программа ожидает одну из команд:

- -f infile outfile01 outfile02 ,где infile – имя файла в котором хранятся входные данные, outfile01 – имя файла в котором будут выходные данные,outfile02 – имя файла в котором будут выходные данные, после выполнения программой функцией(В данном случае Bubble Sort).
- -n number outfile01 outfile02 – похожая команда, в которой number – кол-во артефактов, которые необходимо сгенерировать с помощью функции rnd().  
Остальные параметры такие же, как и в первой команде.

Ввод в программу в файле реализован следующим образом:

- Первый параметр – целое число от 1 до 3 обозначающее тип машины: 1 – грузовик,2 – автобус,3 - автомобиль.Первый параметр – расход топлива,Второй параметр - емкость топливного бака,Третий параметр - индивидуальный параметр(описан выше для каждого типа машины).







Программа протестирована на 7 файловых теста(расположены в папке input\_tests),результаты которых расположены в папке output\_tests(файл типа test01\_out.txt и test01\_sorted.txt означают файл с заполненными и сортированными элементами соответственно,а также out1\_rnd.txt и out1\_sorted.txt обозначающие выходные тесты выполненные после рандомной генерации и такой же отсортированный файл соответственно ).

### Основные характеристики:

- Кол-во единиц компиляции - 5
- Число макроопределений - 5
- Число подпрограмм - 5
- Общий размер текстов - 44.6 КБ
- Размер результатов тестов - 39.1КБ

## Метрики

### Время работы программы и сравнение с прошлой

 Тест	 Размер	 Assembler	 Python	 ООП	 Процедурный стиль
<u>test01</u>	8	0.000002082	0.0009973 seconds	real 0m0,006s user 0m0,001s sys 0m0,000s	real 0m0,012s user 0m0,000s sys 0m0,001s
<u>test02</u>	12	0.0000038	0.0019951 seconds	real 0m0,010s user 0m0,000s sys 0m0,003s	real 0m0,016s user 0m0,000s sys 0m0,004s
<u>test03</u>	1000	0.00048	1.2845616 seconds	real 0m0,012s user 0m0,007s sys 0m0,000s	real 0m0,030s user 0m0,005s sys 0m0,008s
<u>test04</u>	5000	0.001713803	19.7960272 seconds	real 0m0,027s user 0m0,010s sys 0m0,007s	real 0m0,031s user 0m0,010s sys 0m0,012s
<u>test05</u>	9999	0.010218400	92.182657 seconds	real 0m0,024s user 0m0,020s sys 0m0,011s	real 0m0,031s user 0m0,015s sys 0m0,027s



Анализируя время работы программы написанной в стиле ООП ,на динамически типизированном языке,с помощью процедурного стиля и на низкоуровневом языке можно заметить что данная программа работает гораздо быстрее чем та же программа на Python, немного быстрее,чем программы выполненные на C++. Причиной этому может быть то, что данная программа была написана на более низкоуровневом языке.

## Вывод

Программа, написанная на ООП позволяет написать код более эффективно и удобно, и лучше читаемо. Скорость работы программы, написанной в объектно-ориентированном стиле выше ,чем написанной в процедурном стиле. А также удобен в модульной разработке.

Использование модульного программирования позволяет упростить тестирование программы и обнаружение ошибок, так как структура и поведение подчиняются определённым правилам и модули работают независимо. Естественным на более высокоуровневом языке как Python программисту будет легче писать код, так как в данном случае не надо задумываться о том какой тип нужен той или иной переменной и понимать это сами. Но если проект весьма большой, это может понести ряд проблем, из-за которых будет трудно отследить какой тип может попасть в ту или иную переменную и решить эту проблему будет труднее.

Программирование на низкоуровневом языке является очень неудобным для программиста,так как содержит много строчек и команд,трудных для понимания.Естественно,программирование больших программ на низкоуровневом языке займет гораздо больше времени ,вероятность совершить ошибку увеличивается и могут появиться трудности в реализации.