**Задание 14**. Автомобильный транспорт

Сущности:

1. Грузовик (грузоподъемность кг – целое)

2. Автобус (пассажировместимость – короткое целое)

3. Легковой автомобиль (максимальная скорость – короткое целое)

Общие характеристики:

1. Емкость топливного бака в литрах (целое)

2. Расход топлива на 100 км в литрах (действительное)

Функция:

Максимальное расстояние, которое может пройти автомобиль в км (действительное число)

**Задание 22**. Переместить в конец контейнера те элементы, для которых значение, полученное с использованием функции, общей для всех альтернатив, больше, чем среднее арифметическое для всех элементов контейнера, полученное с использованием этой же функции. Остальные элементы сдвинуть к началу без изменения их порядка.

число интерфейсных модулей: 6

число модулей реализации: 5

общий размер исходных текстов: 46,7 Кб

**./program -n 10 tests/out\_randomn10.txt tests/out\_randomn10\_sorted.txt**

**Время работы = 0m0.089s –** 10 транспортных средств (т.с.)

**./program -n 30 tests/ out\_randomn30.txt tests/out\_randomn30\_sorted.txt**

**Время работы = 0m0.110s –** 30 т.с.

**./program -n 50 tests/ out\_randomn50.txt tests/out\_randomn50\_sorted.txt**

**Время работы = 0m0.041s –** 50 т.с.

**./program -n 500 tests/ out\_randomn500.txt tests/out\_randomn500\_sorted.txt**

**Время работы = 0m0.130s –** 50 т.с.

**./program -n 1000 tests/ out\_randomn1000.txt tests/out\_randomn1000\_sorted.txt**

**Время работы = 0m1.102s –** 1000 т.с.

**Test01.txt: Время работы = 0m0.050s**

**Test02.txt: Время работы = 0m0.047s**

**Test03.txt: Время работы = 0m0.065s**

**Test04.txt: Время работы = 0m0.066s**

**Test05.txt: Время работы = 0m0.066s**

**Сборка программы:**

g++ bus.cpp bus.h car.cpp car.h container.cpp container.h main.cpp rnd.h truck.h truck.cpp vehicle.cpp vehicle.h -o program

**Запуск программы:**

Исходные данные для тестирования содержатся в каталоге tests.

**Пример тестирования:**

./program -n 1000 tests/out\_randomn1000 tests/out\_randomn1000\_sorted.txt

**Описание ВС**

|  |  |
| --- | --- |
| **Тип** | **Размер в байтах** |
| **int** | **4** |
| **char** | **1** |
| **double** | **8** |
| **bool** | **1** |

|  |  |
| --- | --- |
| **Таблица классов**  **Container [80004 байт]** | |
| **Поля** | **Static: Отсустствуют  Local: int len[4 байта] Vehicle \*storage[8 \* 10000 байт]** |
| **Методы** | **Static: Отсутствует  Local: void In(FILE \*file); void InRnd(int size); void Out(FILE \*file);**  **double AverageDistance(); void MoveToEnd();** |
| **vehicle [0 байт]** | |
| **Поля** | **Static: Отсутствует  Local: Отсутствует** |
| **Методы** | **Static:**  **Vehicle \*StaticIn(FILE \*file); Vehicle \*StaticInRnd();  Local: languages(); void in(FILE \*file); void inRnd(); void out(FILE \*file); double Quotinent();** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Программная память** | | **Container** | | **Vehicle** | **Производные классы** |
| **Статические** | | **-** | | **Vehicle \*StaticIn(FILE \*file) - 12 байт Vehicle \*StaticInRnd() - 16 байт** | **-** |
| **Локальные** | | **void In(FILE \*file) - 8 байт void InRnd(int size) - 4 байта void Out(FILE \*file) - 8 байт void MoveToEnd() - 8 байт**  **double AverageDistance() -> 16 байт** | | **-** | **-** |
| **Виртуальные** | | **-** | | **Vehicle() - 8 байт void In(FILE \*file) - 8 байт void InRnd() - 8 байт void Out(FILE \*file) - 8 байт double Distance() -> 16 байт** | **Vehicle() - 8 байт void In(FILE \*file) - 24 или 25 байт void inRnd() - 24 или 25 байт void out(FILE \*file) -> 24 или 25 байт + Distance() размер double Distance() -> 24 байт** |
| **Куча** | |
| **Container \*storage -> 80000 байт** | |
| **+ объекты производных классов в зависимости от количества** | |

**Сравнительный анализ**

|  |  |  |  |
| --- | --- | --- | --- |
| **Показатель** | **С** | **С++** | **Данные для сравнения** |
| Время работы | -> | На небольшом количестве данных значений времени исполнения программы особо не отличаются, однако на большом объеме наблюдается значительное уменьшение времени | На 10 значениях: 0,052s/0,089s  На 30 значениях  0,046s/0,110s  На 1000 значениях  4,119s/1,102s |
| Размер исходных текстов | 70,0 Кб | 46,7 Кб | - |
| Агрегирование | Процедурное | ООП |  |

Преимущества и недостатки ОО перед процедурным агрегированием:

- ОО агрегирование предоставляет более устойчивую ассоциацию объекта, поддержанную непосредственным размещением процедур внутри классов,

- наличие на один аргумент меньше в методах класса против дополнительной передачи объекта в процедурном подходе, что сказывается на производительности,

- использование классов улучшает восприятие кода, так как при реализации структуры методы обработки и дальнейшей работы реализуются отдельно от структуры, что уменьшает читабельность кода,

- однако использование структур позволяет инкапсулировать методы обработки и скрыть ненужный для пользователя код;