Ломтев Роман БПИ208 Вариант задания: 7; функция: 22

Описание задания:

Необходимо реализовать программу с использованием объектно-ориентированного подхода и динамической типизацией. В рамках конкретного задания предполагается использование следующих структур данных:

Обобщённый артефакт:

Фильм. Каждая из описанных ниже альтернатив (жанров) содержит название (строка) и год выхода (целочисленная переменная).

Общей функцией для всех альтернатив является нахождение частного от деления года выхода фильма на количество символов в его названии (числовая переменная действительного типа).

Базовые альтернативы (жанры):

— *Документальный фильм*. Характеризуется длительностью (целочисленная переменная).

— *Игровой фильм*. Содержит имя режиссёра (строка).

— *Мультфильм*. Характеризуется одним из следующих способовсоздания:

1) Рисованный
2) Кукольный
3) Пластилиновый

Способ создания хранится в переменной перечислимого типа.

Также необходимо обеспечить хранение и обработку данных с помощью спроектированного для этих целей контейнера. Помимо базового функционала необходимо реализовать упорядочивание посредством перемещения в конец контейнера тех элементов, 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.

Основные характеристики программы:

— Число интерфейсных модулей:

— Число модулей реализации:

— Общий размер исходных текстов:

— Размер исполняемого кода:

— Время работы программы на различных тестах:

Test 01: 0.001 c

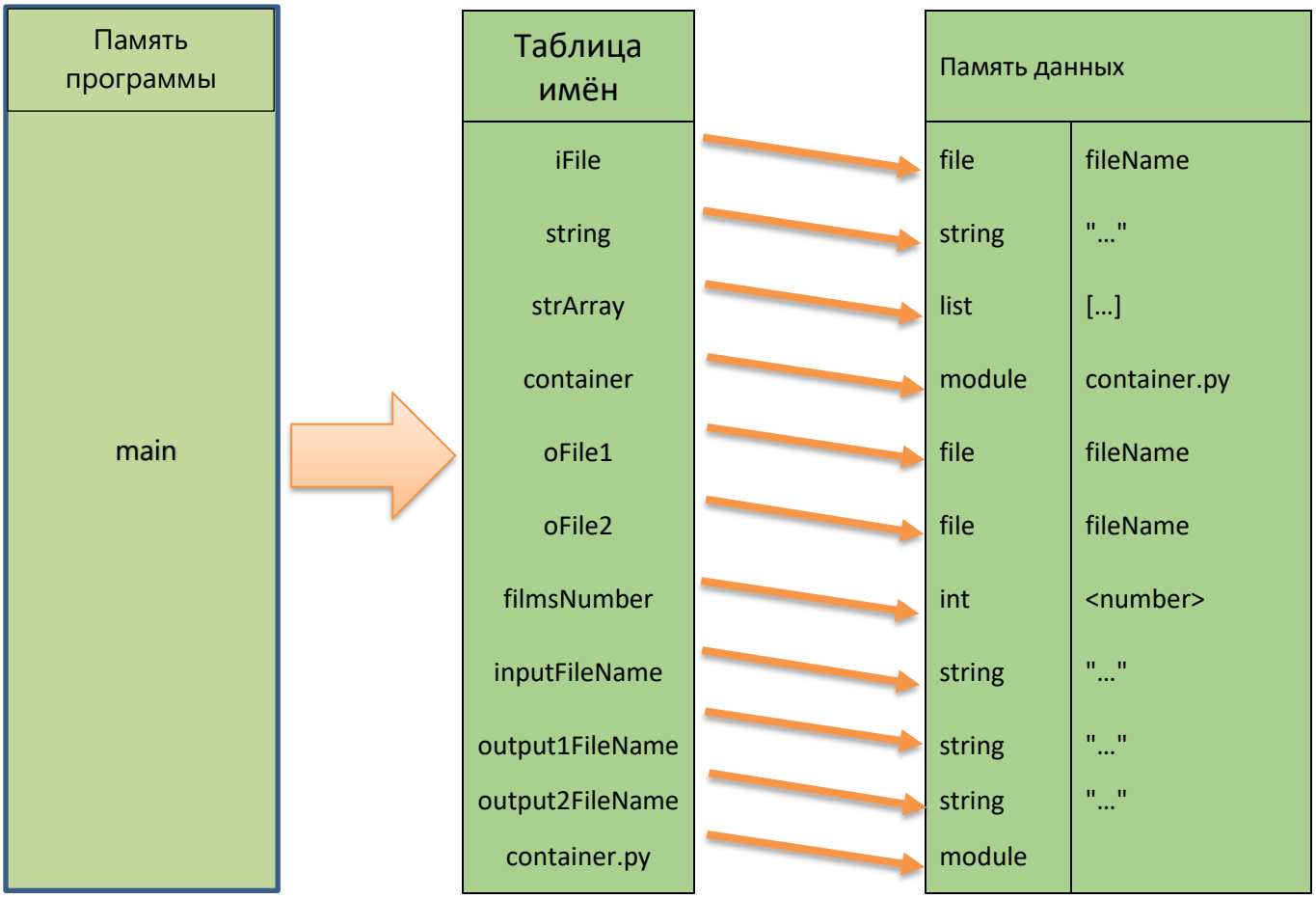
Test 02: 0.009 c

Test 03: 0.002 c

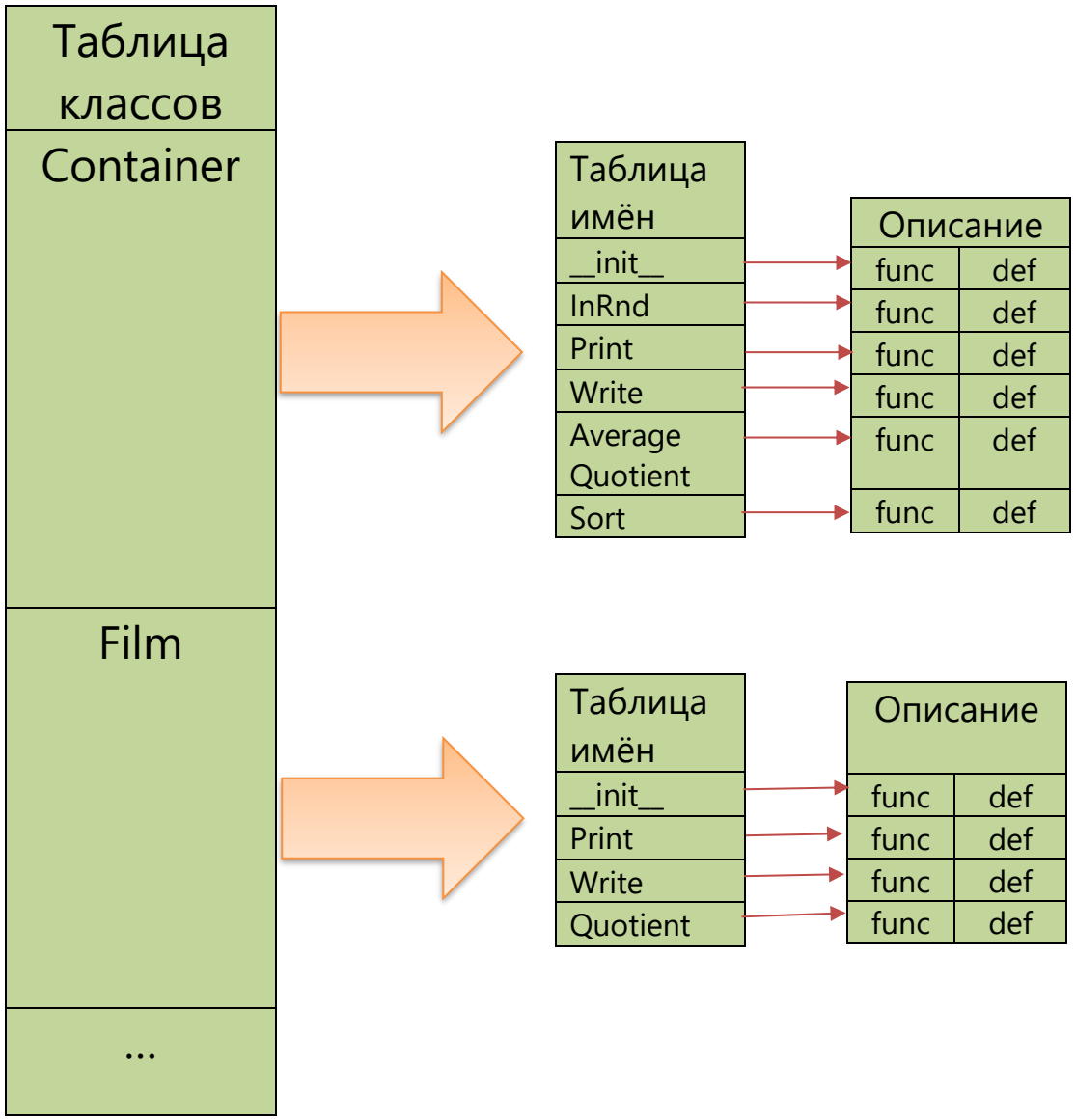
Test 04: 17.699 c

Test 05: 0.009 c

Отображение на память содержимого модуля main



Отображение содержимого классов Container и Film



Остальные классы (подклассы Film) описываются аналогично

Сравнительный анализ:

Выделим преимущества динамической типизации по сравнению со статической:

- Краткость кода. Объём получается намного меньше благодаря тому, что нет нужды расписывать каждое приведение типов
- Удобство описания обобщённых алгоритмов. В данном случае динамическая типизация выступает в качестве универсального инструмента, который освобождает от лишней возни с приведениями.

Недостатки динамической типизации:

- Повышенный риск ошибки во время исполнения. В некоторых условиях статическая типизация высвечивает потенциальные ошибки на этапе компиляции, чего нет у динамической типизации.
- Скорость выполнения. Поскольку постоянно нужно проверять корректность взаимодействия обобщённых экземпляров, это затрачивает намного больше времени, чем если бы мы это делали один раз – на этапе компиляции. Ярким примером этому послужит сравнение времени выполнения программ из задания №3 (данного) и задания №2 на тесте №4, где в контейнере хранится 10000 элементов:

  Test №4

  C++ (ООП подход, статическая типизация) – 0,398 с
  Python (ООП подход, динамическая типизация) – 17.699 с

  Если на маленьком объёме данных программы работают примерно одинаково, то на большом количестве уже видно колоссальную разницу по времени.