ОПИСАНИЕ ЗАДАНИЯ (задание 2, дополнительная функция 20):

Разработать программный продукт с использованием *динамической проверки типов во время выполнения (стиль написания – произвольный)*. Программа должна содержать следующие структуры:

Обобщенный артефакт, используемый в задании	**Базовые альтернативы**
Плоская геометрическая фигура, размещаемые в координатной сетке	1. Круг (целочисленные координата центра окружности, радиус) 2. Прямоугольник (целочисленные координаты левого верхнего и правого нижнего углов) 3. Треугольник (целочисленные координаты трех углов)

Для всех альтернатив общей переменной является **цвет** (перечислимый тип). Он может принимать значения: красный, оранжевый, желтый, зеленый, голубой, синий, фиолетовый. Общей функция всех альтернатив выступает вычисление периметра фигуры (действительное число). В качестве дополнительной функции контейнера необходимо удалять из него те фигуры, периметр которых больше, чем среднее арифметическое периметров всех фигур контейнера (остальные фигуры передвигать к началу контейнера с сохранением порядка).

Также нужно: разработать тестовые входные данные и провести тестирование и отладку программы на этих данных (при необходимости, программа должна правильно обрабатывать переполнение по данным); описать структуру используемой ВС с наложением на нее обобщенной схемы разработанной программы; зафиксировать количество заголовочных файлов, программных файлов, общий размер исходных текстов, полученный размер исполняемого кода (если он формируется), время выполнения программы для различных тестовых наборов данных.

(*) В качестве доп. функционала реализован способ генерации тестовых файлов.

СТРУКТУРНАЯ СХЕМА АРХИТЕКТУРЫ ВС С ПРОГРАММОЙ:

Программа разработана в 64 битной системе (дистрибутив Ubuntu на ярде Linux). Разработка велась на языке Python версии 3.9 с использованием объектно-ориентированной парадигмы программирования (с исключением для main.py). Рассмотрим все классы разработанной программы и опишем 5 функций, на примере которых отобразим графически архитектуру виртуальной Python машины с наложением на нее обобщенной схемы разработанной программы:

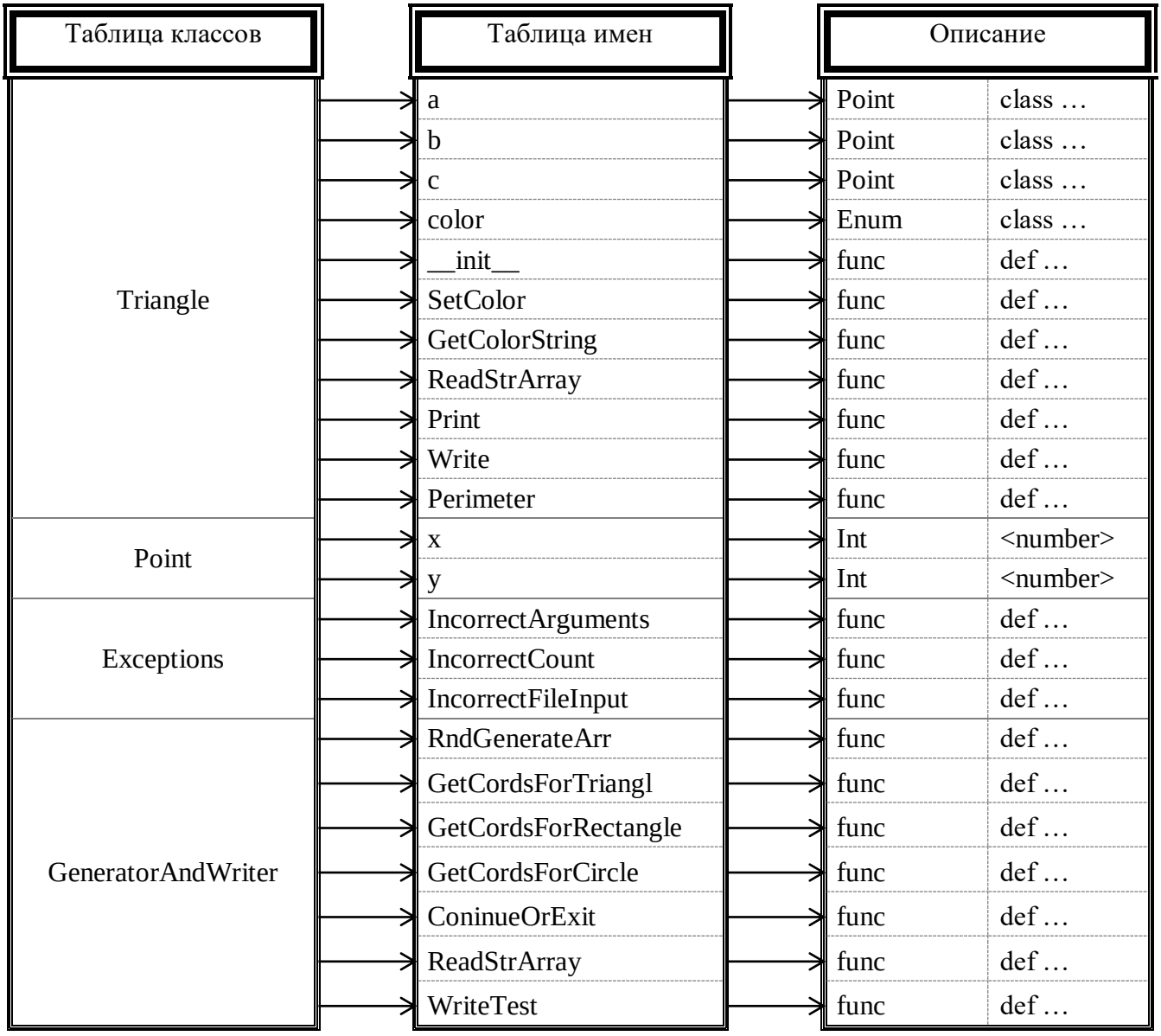
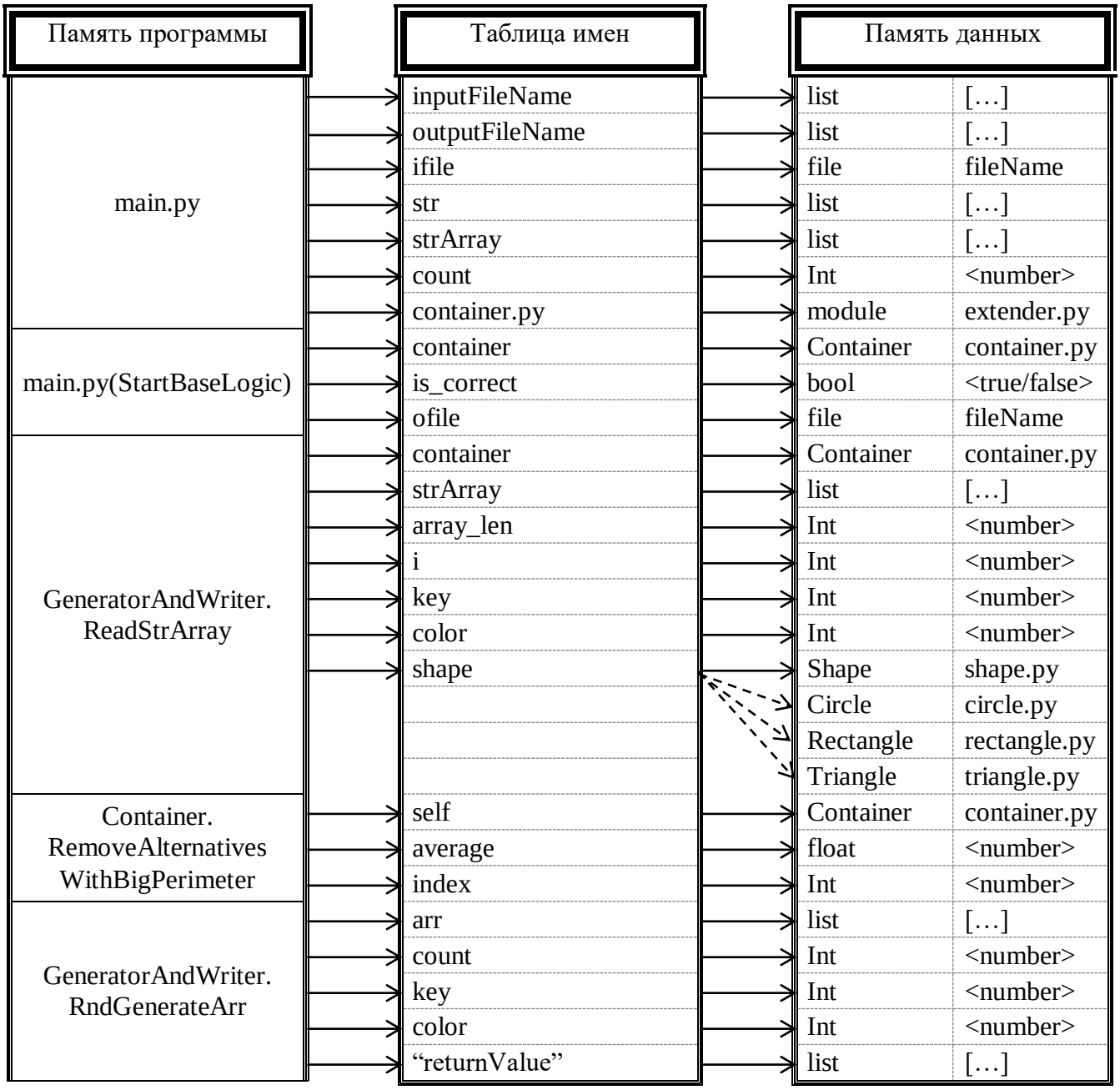


Таблица классов	Таблица имен	Описание	
Triangle	a	Point	class …
	b	Point	class …
	c	Point	class …
	color	Enum	class …
	__init__	func	def …
	SetColor	func	def …
	GetColorString	func	def …
	ReadStrArray	func	def …
	Print	func	def …
	Write	func	def …
	Perimeter	func	def …
Point	x	Int	<number>
	y	Int	<number>
Exceptions	IncorrectArguments	func	def …
	IncorrectCount	func	def …
	IncorrectFileInput	func	def …
GeneratorAndWriter	RndGenerateArr	func	def …
	GetCordsForTriangl	func	def …
	GetCordsForRectangle	func	def …
	GetCordsForCircle	func	def …
	ConinueOrExit	func	def …
	ReadStrArray	func	def …
	WriteTest	func	def …

ОТРАЖЕНИЕ НА ПАМЯТЬ МЕТОДОВ КЛАССОВ:



ОСНОВНЫЕ ХАРАКТЕРИСТИКИ ПРОГРАММЫ:

- Число заголовочных файлов – 1 (extender.py)
- Число модулей реализации – 9
- Общий размер исходных текстов – 535 строк кода вместе с комментариями (21.2 кб).
- Размер исполняемого файла – 0 (файл не генерируется, так как питон – скриптовый язык).
- Время выполнения программы для различных тестовых прогонов:

Номер теста	Время выполнения в секундах
Тест1 (3 корректных фигуры)	0.000999
Тест2 (пустой файл)	0.001000
Тест3 (некорректный ввод)	0.009978
Тест4 (некорректный цвет)	0.001000
Тест5 (некорректные аргументы)	0.009897
Тест6 (10000 элементов)	0. 577981

СРАВНЕНИЕ С ПРЕДЫДУЩИМИ ВЕРСИЯМИ ПРОГРАММЫ:

Предыдущее задание было посвящено разработки аналогичного консольного приложения, но с использованием языка со статической типизацией. Использование динамического связывания и виртуальной машины Python, вывод данных в консоль привели к ряду отличий данной версии программы, а именно:

- Программа стала работать существенно медленнее предыдущих 2-х версий. Примерно в 5–10 раз по сравнению с аналогичной программой, разработанной на C++, и программой на C (процедурный подход). Для примера приведем таблицу с временными результатами тестовых наборов предыдущего задания:

Номер теста	Время выполнения в секундах
Тест1 (3 корректных фигуры)	0.000272
Тест2 (пустой файл)	0.000251
Тест3 (некорректный ввод)	0.000211
Тест4 (некорректный цвет)	0.000116
Тест5 (некорректные аргументы)	0.000234
Тест6 (10000 элементов)	0.066709

  Связано это с особенностями работы виртуальной машины Python (в особенности, наличие дополнительного слоя – интерпретатор), а также со спецификой языка с динамической типизацией (требуется время на связывание).
- В программе появились новые опции – вывод содержимого в консоль (раньше вывод осуществлялся только в файл); отпала необходимость явно задавать выходной файл (если выходной файл не задан, то программа сгенерирует его сама); информация об ошибках стала более информативной.
- Вывод в файл теперь осуществляется в один файл, а не в 2, как это было раньше. Это помогает немного экономить время на очистки потока и запросе файла у системы.
- Пропала возможность измерять размер типов. В Python нет возможности получить размер типа (только объекта какого-то типа), поэтому провести разумный анализ затрачиваемой памяти не представляется возможным.
- Несмотря на расширение функционала, количество строк кода уменьшилось в 2 раза, что связано с облегченным синтаксисом питона, снятие с программиста обязанностей работы с указателями, динамической память и очищать эту самую память.