Вариант 4, Задание 23

Реализовать контейнер для хранения альтернатив и их параметров.

Обобщённый артефакт: объемные трехмерные геометрические фигуры.

Базовые альтернативы:

· **Шар** - целочисленный радиус.

· **Параллелепипед** - целочисленная длина первого ребра, целочисленная длина второго ребра, целочисленная длина третьего ребра.

· **Правильный тетраэдр** – длина ребра.

Общая для всех альтернатив переменная – плотность материала фигуры – действительное число.

Общая для всех альтернатив функция: вычисление площади поверхности – действительное число.

Обработка данных в контейнере: переместить в начало 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.

Входные данные:

Форматы входной команды:

· «-f <файл с входными данными> <файл для вывода фигур в контейнере> <файл для вывода отсортированных согласно заданию 23 фигур в контейнере>»

Например, ‘-f input.txt output.txt outputsorted.txt’.

· «-n <количество фигур для генерации> < файл для вывода фигур в контейнере > < файл для вывода отсортированных согласно заданию 23 фигур в контейнере >»

Например, ‘-n 50 output.txt outputsorted.txt’.

Формат описания фигур в файле входных данных:

· **Шар:**

1 <плотность материала> <целочисленный радиус>

Например, 1 10.0 5 - шар плотностью 10.0 радиуса 5.

· **Параллелепипед:**

2 <плотность материала> <длина первого ребра> <длина второго ребра> < длина третьего ребра>

Например, 2 15.3 3 4 5 - параллелепипед плотностью 15.3 с ребрами длиной 3, 4 и 5.

· **Правильный тетраэдр:**

3 <плотность материала> <длина ребра>

Например, 3 7.5 5 – правильный тетраэдр плотностью 7.5 и длиной ребра 5.

Выходные данные:

Вывод информации о контейнере:

Container contains <количество фигур в контейнере> фигур:

- **для Шара**:

Ball: density = <плотность материала фигуры>, radius = <радиус шара>, surface area = <площадь поверхности>

- **для Параллелепипеда**:

Parallelepiped: density = <плотность материала фигуры>, first edge length = <длина первого ребра>, second edge length = <длина второго ребра>, third edge length = <длина третьего ребра>, surface area = <площадь поверхности>

- **для Тетраэдра**:

Tetrahedron: density = <плотность материала фигуры>, edge length = <длина ребра>, surface area = <площадь поверхности>

Вывод среднего арифметического площадей поверхностей:

Average surface area = <среднее арифметическое площадей поверхностей фигур в контейнере>

Метрики

Состав: 6 файлов реализации: ball.py; container.py; figure.py; main.py; parallelepiped.py; tetrahedron.py.

Размер файлов: **10 КБ**

Время исполнения:

20 фигур

Аргументы:

-n 20 tests/randomtest/output.txt tests/randomtest/outputsorted.txt

Время: **0.00125 с.**

100 фигур

Аргументы:

-n 100 tests/randomtest/output.txt tests/randomtest/outputsorted.txt

Время: **0.00344 с.**

500 фигур

Команда:

-n 500 tests/randomtest/output.txt tests/randomtest/outputsorted.txt

Время: **0.0139 с.**

2000 фигур

Аргументы:

-n 2000 tests/randomtest/output.txt tests/randomtest/outputsorted.txt

Время: **0.0689 с.**

10000 фигур

Аргументы:

-n 10000 tests/randomtest/output.txt tests/randomtest/outputsorted.txt

Время: **0.316 с.**

Сравнение:

Размер файлов с использованием ООП увеличился относительно процедурного подхода, но программа стала выполняться быстрее.

Отображение на память содержимого

Память программы	Таблица имён	Память данных	
main.py	container	list	[…]
	input_file_name	file	fileName
	input_string	string	"…"
	data_array	string	"…"
	container	module	container.py
	output_file	file	fileName
	output_sorted_file	file	fileName
	number_of_figures	int	<number>
container.py			
def file_input	ball	module	ball.py
	parallelepiped	module	parallelepiped.py
	tetrahedron	module	tetrahedron.py
	container	list	[…]
	data	string	"…"
	i	int	<number>
	ball	list	[…]
	parallelepiped	list	[…]
	tetrahedron	list	[…]
	figure_type	int	<number>
def random_input	ball	module	ball.py
	parallelepiped	module	parallelepiped.py
	tetrahedron	module	tetrahedron.py
	container	list	[…]
	i	int	<number>
	number_of_figures	int	<number>
	ball	list	[…]
	parallelepiped	list	[…]
	tetrahedron	list	[…]
	figure_type	int	<number>
def write_in_file	output	file	fileName
	container	list	[…]
def sort	container	list	[…]
def average_surface_area	avg_surface_area	float	<number>
	container	list	[…]
ball.py			
def get_figure	ball	list	[float, int]
	data	string	"…"
	i	int	<number>
def get_random_figure	ball	list	[float, int]
def surface_area	ball	list	[float, int]
def write_figure_in_file	output	file	fileName
	ball	list	[float, int]
parallelepiped.py			
def get_figure	parallelepiped	list	[float, int, int, int]
	data	string	"…"
	i	int	<number>
def get_random_figure	parallelepiped	list	[float, int, int, int]

def surface_area	parallelepiped	list	[float, int, int, int]
def write_figure_in_file	output	file	fileName
	parallelepiped	list	[float, int, int, int]
tetrahedron.py			
def get_figure	tetrahedron	list	[float, int]
	data	string	“…”
	i	int	<number>
def get_random_figure	tetrahedron	list	[float, int]
def surface_area	tetrahedron	list	[float, int]
def write_figure_in_file	output	file	fileName
	tetrahedron	list	[float, int]