Дисциплина «Программирование»

Практическое задание №4

10 ноября – 16 ноября

«Овощной склад»

Процесс выполнения этого задания состоит из трёх частей:
1) реализация программы, согласно описанным в условии требованиям;
2) оценивание работ других студентов;
3) период «споров».

Период реализации программы:

После выдачи задания Вам необходимо выполнить его и загрузить архив (*.zip) с решением задачи (полностью заархивировать решение, созданное средой разработки) до крайнего срока. В работе строго запрещается указывать ФИО, а также любую другую информацию, которая может выдать авторство работы. В случае выявления факта деанонимизации работы, работа может быть аннулирована.

Период взаимного оценивания:

После окончания срока, отведенного на реализацию программы, начинается период взаимного оценивания. Вам будет необходимо проверить пять работ других студентов, также выполнявших данное задание, согласно критериям оценивания. Проверка осуществляется анонимно: Вы не знаете, чью работу Вы проверяете, также, как и человек, кому принадлежит решение, не знает, кем была проверена его работа. Помимо оценки Вам необходимо указать комментарий к каждому из критериев. В случае, если Вы снижаете балл, необходимо подробно описать, за что именно была снижена оценка. Также рекомендовано писать субъективные комментарии, связанные с тонкостями программной реализации, предлагать автору работу более оптимальные на ваш взгляд решения. Снимать баллы за субъективные особенности реализации запрещено.

Период споров:

По окончании периода взаимного оценивания, в случае если Вы не согласны с оценкой, выставленной одним из проверяющих, Вы можете вступить с этим студентом в анонимный диалог с целью уточнения причин выставления оценки по тому или иному критерию. В случае, если проверяющий не ответил Вам, или вы не пришли к обоюдному решению об изменении оценки, Вы можете поставить флаг, и работа будет рассмотрена одним из преподавателей.

Флаги, поставленные без предварительного обсуждения с проверяющим или после окончания периода выставления флагов, будут отклонены.

Также допустима перепроверка Вашей работы преподавателем. В таком случае оценки других проверявших работу не учитываются.

Дедлайн загрузки работы: 16 ноября 23:59
Дедлайн проверки: 19 ноября 23:59
Дедлайн обсуждения оценок: 21 ноября 23:59
Дедлайн выставления флагов: 22 ноября 23:59

Возможность поздней сдачи работы в этом задании предоставляться не будет.

Оценивание:
$О_{итог} = 0,8 * О_{задание} + 0,2 * О_{проверки}$, где $О_{задание}$ – неокруглённая десятибалльная оценка за решение задания выставленная проверяющими с учётом возможной перепроверки преподавателем, а $О_{проверки}$ неокруглённая десятибалльная оценка, выставленная студентами, чьи работы вы проверяли. Также в случае, если преподаватель обнаружит, что Вы проверили работы некачественно к Вам могут быть применены санкции в виде штрафа до 3 десятичных баллов от $О_{итог}$ (в таком случае оценка вычисляется по формуле $О_{итог} = О_{задание} – Штраф$).

Необходимо разработать консольное приложение – «Овощной склад».

В программе должны быть реализованы три вида сущностей:

1. Склад (есть ровно один в программе, его характеристики - кроме списка контейнеров - неизменны)
2. Контейнер с ящиками
3. Ящик с овощами

Далее указаны описания сущностей:

1. **Склад**

Склад является хранилищем контейнеров. Содержимое склада может меняться - туда могут поступать новые контейнеры и удаляться старые. При этом склад может содержать ограниченное число контейнеров (это число определяется пользователем) - если происходит добавление нового контейнера в заполненный склад, то он заменяет собой тот, что был добавлен не позднее всех остальных). Также склад взимает фиксированную плату за

хранение контейнера (также определяется пользователем, она одинакова для каждого контейнера).

Хранение контейнера может быть нерентабельно. При поступлении контейнера на склад рассчитывается степень его повреждения (случайная величина в диапазоне [0; 0.5)) - именно такую долю стоимости теряет каждый ящик внутри контейнера. Если суммарная стоимость содержимого контейнера после этого не превосходит стоимость хранения, то такой контейнер на склад не помещается.

2. **Контейнер**

Контейнер является хранилищем ящиков. Контейнер ограничивает максимальную суммарную массу хранимых ящиков, ограничение - случайное целое число в диапазоне [50, 1000].

При создании контейнера в него помещаются ящики с овощами. Количество и список помещаемых в контейнер ящиков указывается пользователем. При этом суммарная масса ящиков не может превосходить максимально допустимую массу контейнера - таким образом, если добавление ящика приводит к превышению максимально допустимой массы, то он не добавляется в контейнер.

3. **Ящик овощей**

Ящик овощей описывается двумя параметрами - массой в килограммах и ценой за килограмм. Оба параметра указываются при создании, при этом масса не может быть изменена позднее.

Ввод данных:

Действие 1: создаётся склад - он должен быть единственный. Пользователем указываются его характеристики - вместимость и цена хранения.

Следующие N действий: Пользователь вводит одну из двух команд:

1. Добавление контейнера на склад (после этого считываются данные о контейнере)
2. Удаление контейнера со склада (указывается идентификатор удаляемого контейнера - что является идентификатором, определяется разработчиком)

Необходимо реализовать 2 способа ввода данных:

Способ 1:

Все параметры вводятся пользователем через консоль (о складе, каждом действии, каждом контейнере и каждом ящике).

Способ 2: всё считывается из трех файлов:

1. Файл с описанием склада
2. Файл с описанием действий
3. Файл с описанием контейнеров

1. Описание склада - формат определяется разработчиком.
2. Описание действий - список из добавлений и удалений контейнеров со склада
3. Описание контейнера - вся информация о контейнере (включая список ящиков с овощами) в одной строке, контейнеры упорядочены в порядке создания.

В остальном формат входных данных формат определяется разработчиком.

В результате выполнения программы в консоль или в файл, указанный пользователем, выводится информация о складе и его содержимом - списке контейнеров с указанным содержимым. Формат определяется разработчиком.

Дополнительные требования (критерии оценивания):

1. Для проверки в систему PeerGrade должен быть загружен архив с решением. Тип проекта должен быть Console Application .Net Core 3.1. Ожидается, что проверка работы будет проводиться, в среде разработки Visual Studio 2019, поэтому в случае выполнения задания с использованием другой среды разработки настоятельно рекомендуется проверить возможность открытия и запуска проекта в этой среде разработки.
2. Текст программы должен быть отформатирован согласно кодстайлу языка C#[1]. Для автоматического форматирования в среде Visual Studio достаточно нажать Ctrl+K, D. Идентификаторы должны соответствовать соглашению о именовании C#[2]. Основным требованием, которое требуется соблюдать в данном задании – это написание имён локальный переменных с использованием camelCasing, а при именовании методов и типов PascalCasing.
3. Программа не должна завершаться аварийно (или уходить в бесконечный цикл) при любых входных данных. При некорректных входных данных программа должна выводить сообщение об ошибке и запрашивать ввод заново.

---

[1] https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions

[2] https://docs.microsoft.com/en-us/dotnet/standard/design-guidelines/naming-guidelines

4. Программа должна быть декомпозирована. Каждый класс должен быть описан в отдельном файле. Каждый из логических блоков должен быть выделен в отдельный метод. Не строго, но желательно, чтобы каждый метод по длине не превышал 40 строк.
5. Интерфейс программы должен быть понятен. Пользователю должны выводиться подсказки о возможных дальнейших действиях и иные необходимые сообщения. Предполагается, что для успешного использования программы не требуется обращения к исходному коду программы.
6. Должно быть реализовано повторение решения. То есть, после окончания работы приложения, пользователю предлагается завершить программу или начать работу снова.
7. Текст программы должен быть документирован. Необходимо писать, как комментарии перед методами, так и комментарии, поясняющие написанный внутри метода код. Названия переменных и методов должны быть на английском языке и отражать суть хранимых значений / выполняемых действий.
8. Для получения отличной оценки более 8 студенту необходимо реализовать больший функционал, чем описано в задании выше. В данном задании Вам необходимо самим определить что именно будет являться дополнительным функционалом, обратите внимание на то, что изменилось оценивание данного пункта.
9. Также оценивается общее впечатление, которое производит работа (как с точки зрения пользовательского интерфейса, так и с точки зрения написания кода программы). Эта часть оценки остаётся на усмотрение проверяющего.