Задачи II семестра

1 блок

1. Переписать консольное приложение по обработке изображений различными фильтрами на объектно-ориентированном языке (C#, Java, Kotlin). Нельзя использовать класс System.Drawing.Bitmap или аналогичный в языках на JVM.

2 блок

2. Реализовать иерархию классов для одной из игр и написать для неё ботов, реализующих как минимум 3 разные стратегии на усмотрение автора и имеющих некоторую стартовую сумму денег. Показать, сколько в среднем денег остаётся у каждого бота после 40 ставок. К решению приложить диаграмму классов UML.

- Блэкджек с базовым вариантом правил и 8 замешанными колодами (https://en.wikipedia.org/wiki/Blackjack) .
- Баккара в варинате Punto banco и 8 замешанными колодами (https://en.wikipedia.org/wiki/Baccarat_(card_game)).
- Рулетка с 1 зеро и ставками на цвет, четность, дюжину и конкретное число (https://en.wikipedia.org/wiki/Roulette).

Вынести базовый класс/интерфейс и реализации ботов в отдельные библиотеки. В программе, работающей с ботами, выводить в консоль требуемую информацию через методы или свойства базового класса/интерфейса.

3. Написать с использованием дженериков класс, реализующий одну из перечисленных коллекций с возможностью добавления, удаления и поиска по ней. Возможно, в реализации потребуется сделать более, чем один класс.

1) Двусвязный список.
2) Хэш-таблица.
3) Динамический массив.

Номер класса определяется как единица плюс остаток от деления длины имени на 4.

3 блок

4. Модифицировать программу для работы с ботами из задачи 2 таким образом, чтобы она искала по заданному пути библиотеки с ботами, загружала их и играла ботами из них.

5. Написать приложение для отображения текущей погоды в Санкт-Петербурге. Источниками данных являются открытые веб-сервисы, например:

1) https://www.weather.gov/documentation/services-web-api
2) https://www.tomorrow.io/
3) https://stormglass.io/
4) https://openweathermap.org/api

В программе необходимо показать данные не менее, чем от двух источников. Номер первого источника определяется как единица плюс остаток от деления длины имени на 4. Номер второго источника определяется как единица плюс остаток от деления длины фамилии на 4. В случае равенства полученных значений в качестве второго используется следующий номер (с учётом единицы плюс остатка от деления на 4).

Требуется отобразить текущую температуру (в градусах Цельсия и Фаренгейта), облачность, влажность, осадки, направление и скорость ветра. В случае, если соответствующих данных нет, отображать надпись «Данных нет». В данной задаче достаточно вывода в консоль. Предусмотреть возможность корректного выхода и обновления данных по нажатию клавиши. Предусмотреть корректную обработку ошибок в случае, если сервисы по какой-либо причине недоступны.

Классы, инкапсулирующие логику работы с веб-сервисом, должны обладать одним и тем же интерфейсом.

6. Настроить IoC-контейнер по своему выбору, с его помощью инициализировать классы 7 задачи и запустить её. Продемонстрировать возможность отключения того или иного сервиса на момент запуска программы.

7. Написать два приложения, использующее две технологии пользовательского интерфейса (например, WinForms, WPF, Swing, JavaFX), для отображения погодных данных в задачах 7-8. Общая функциональность по работе с веб-сервисами должна быть разделяема между программами.

4 блок

8. bash – это одна из наиболее популярных командных оболочек под UNIX. Необходимо реализовать свою версию этой программы с урезанной функциональностью. Она должна поддерживать следующие команды:

echo – вывести на экран аргумент(-ы);

exit – выйти из интерпретатора;

pwd – вывести на экран текущий рабочий каталог (название и список файлов);

cat [FILENAME] – показать на экране содержимое файла

wc [FILENAME] – показать на экране количество строк, слов и байт в файле

Команда, не распознанная как одна из приведённых выше, приводит к попытке запуска механизмами операционной системы (вроде Process.Start() в .NET).

Программа также должна поддерживать следующие возможности:

оператор $ - присваивание и использование локальных переменных сессии (например, $PATH, $a=4)

оператор | - конвейерная обработка команд. Результат выполнения одной команды становится входом для другой.