Задание (общее для всех вариантов)

1. Спроектировать, реализовать на C++, протестировать и отладить первый из указанных необходимых классов отдельно от остальных;

2. Спроектировать, реализовать на C++ и протестировать полную версию программы;

3. Провести рефакторинг, превратив разработанную программу в библиотеку, пригодную для вызова из среды другого языка программирования;

4. Реализовать на языке Python или на другом высокоуровневом языке по выбору студента пользовательский интерфейс к библиотеке.

Общие пояснения ко всем вариантам задания

* Перечень классов и их методов не является исчерпывающим; можно вводить новые классы и методы;
* строковые данные можно считать состоящими только из символов ASCII, обрабатывать кодировки и многобайтовые представления символов не требуется;
* работа всех созданных классов, кроме тривиальных, должна проверяться встроенными тестами;
* приложение должно допускать вывод содержащейся в нём информации в человекочитаемом формате;
* классы-контейнеры должны работать равноправно с хранящимися в них данными различных типов, используя возможности наследования;
* все сложные участки кода должны сопровождаться комментариями, позволяющими разобраться в работе программы (число комментариев, их частота и т.д. никак не регламентируются);
* проектировать полноценный пользовательский интерфейс (консольный или иной) в частях задания 1-3 не требуется; можно ограничиться вызовом встроенных тестов, если они покрывают основные сценарии использования программы;
* пользовательский интерфейс в части 4 может быть по выбору студента консольным, графическим либо сетевым, но в любом случае должен давать пользователю исполнить все основные сценарии использования приложения. Все необходимые пояснения о том, как нужно организовать пользовательские интерфейсы каждого типа, будут приведены в лекциях;
* при желании студента возможно выполнение части 4 не на Python, а на другом языке программирования, который студент изучил или планирует изучить самостоятельно.

Данные об авиарейсах компании

**Описание приложения**

Приложение хранит данные о маршрутах авиарейсов, совершаемых авиакомпанией. Все рейсы выполняются регулярно каждую неделю по одним и тем же маршрутам. Каждый самолёт в одно и то же время может находиться только в одном рейсе.

**Необходимые классы**

Рейс между двумя пунктами: описывается названиями этих двух пунктов (строки), номерами дней недели начала и окончания рейсов, временем отбытия, временем прибытия, идентификатором самолёта. Время повсюду задаётся в едином часовом поясе, поэтому различия часовых поясов учитывать не надо. Должно поддерживаться автоматическое создание обратного рейса (между теми же пунктами, но в другом порядке) и сравнение двух рейсов по времени отбытия с помощью перегруженных операторов < и ==.

Циклический рейс (рейс по замкнутому циклу между набором пунктов): описывается массивом названий этих пунктов, днями недели начала и окончания рейсов, временем отбытия, временем прибытия в каждый пункт цикла, идентификатором самолёта.

Пункт маршрута: описывается названием, а также максимальным и минимальным временем в пути до связанных с ним маршрутами пунктов.

Расписание: упорядоченный однонаправленный список (лёгкий вариант) либо 2-3 дерево (сложный вариант). Содержит описания рейсов, упорядоченные по времени отлёта. При внесении рейсов расписание следует проверять на корректность; должна быть возможность получить время нахождения в воздухе для каждого самолёта. Необходимо обеспечить вывод недельного расписания для каждого из самолётов в человекочитаемом формате, а также подбор быстрейшего пути из одного пункта в другой для желаемого времени вылета из первого пункта. Сведения об ошибках в расписании должны сообщаться пользователю.