**Описание консольной игры-стратегии «Олимпийские игры»**

Цель игры: завоевать как можно больше медалей, то есть попасть на лидирующие позиции в топе команд из разных стран(желательно на первое место).

***Описание по классам***

class Game создает новую команду. Нужен для того, чтобы можно было создавать, например, 2 разных команды в одной игре (но мы будем создавать одну команду, просто вдруг мы что-то потом захотим изменить, так будет удобнее).

class Team предназначен для непосредственной работы пользователя с игрой. Пользователь будет обращаться именно к функциям этого класса. Так как спортсменов может быть несколько у одного вида спорта, они хранятся тут в векторе.

class AthleteFactory фабрика содержит виртуальные методы создания хоккеистов, фигуристов и биатлонистов

class NorwayAthleteFactory : public AthleteFactory отвечает за создание непосредственно Норвежских сортсменов

class RussianAthleteFactory : public AthleteFactory отвечает за создание непосредственно Русских спортсменов

class Biathlete определяет параметры и методы биатлонистов. Speed – скорость, energy – энергия и постоянная accuracy – точность.

Методы: стрелять, ускориться, замедлиться, показать информацию.

class Figure\_skater имеет параметры: очки, энергия, навыки, выносливость. Методы : выполнить упражнения а и б, показать информацию.

class Hockey\_player

Параметры: энергия, точность, выносливость.

Методы: забить гол (попытаться, не факт, что получится), показать информацию.

Далее все классы-насленики, такие как

class NorwayBiathlete : public Biathlete определяю конкретные детали реализации для каждого спорта каждой страны. В них переопределяются методы, которые должны иметь разное поведение для разных стран.

class Composite компоновщик. Содержит функции добавить элемент в набор скомпонованных, удалить и саму ту функцию, по отношению к которой мы хотим группировать

людей. В моем случае это взятие счета. То есть мы может брать счет как у отдельных людей, отдельно спортсменов разных видов спорта, но и даже набора людей, не зависимо от их вида спорта.

class Composite\_biathlete : public NorwayBiathlete, public RussianBiathlete компоновщик биатлонистов. Нужен, чтобы мог стрелять как один человек, так сразу несколько.

class Composite\_hockey : public NorwayHockey, public RussianHockey компоновщик хоккеистов. Нужен, чтобы мог пытаться забивать гол как один человек, так сразу несколько

class Decorator :public Figure\_skater декорирует фигуриста, делая его характеристики лучше чем обычные.

class OlympicChempion :public Decorator непосредственно содержит те функции, которые нужно задекорировать: прыжки фигуриста. Теперь, когда надо создать человека, наделенного такими качествами, мы будем создавать именно объект этого класса.

Class Command содержит различные команды. Это класс паттерн Команда. Создавая игру, мы думаем не только о том, что есть сейчас, но и о возможном будущем. Поэтому создается паттерн, каждая команда является подклассом данного, у них у всех одинаковая функция execute(). Тогда в возможном будущем можно было бы сначала дать много команд, запихать их в одну очередь, а потом по порядку брать и выполнять. В моем случае в этом нет необходимости, но именно в целях гибкости кода, лучше сделать так.

class Observer - наблюдатель. Создан для того, чтобы следить за командой. У него есть класс наследник Lived\_player : public Observer – смотрит, если у какой-то страны все спортсмены выдохлись и не могут больше принимать участие в соревновании, то пора заканчивать игру.

***Словесное описание:***

В начале игры пользователь выбирает страну, за которую он будет играть: Норвегия или Россия. Далее он должен сформировать свою команду. В состав команды могу входить только фигуристы, биатлонисты и хоккеисты (спортсмены других видов спорта не допущены из-за допинга). Чтобы сформировать команду, пользователю нужно будет ввести количество спортсменов по каждому виду спорта.

Фигуристы имеют счёт – их баллы за прокат. У каждого есть выносливость и энергия, вторая по мере выполнения упражнений будут уменьшаться. Если параметр обнулиться, спортсмен теряет сознание и на этом заканчивается его выступление.

Хоккеисты имеют энергию – изменяемый параметр, точность и выносливость – статические. Могут забивать шайбу и допускать возможность того, что соперник забьет им.

Биатлонисты могут стрелять, ускорятся и отдыхать во время езды на лыжах. Поэтому есть атрибут скорость, который изначально для разных стран разный. Результативность попаданий зависит от энергии и точности в данный момент времени.

Так как есть две страны, у которых одинаковые спортсмены с одинаковыми функциями, но разными параметрами, нам удобно использовать фабрику для производства этих спортсменов.  
Для каждого вида спортсменов создается свой класс. В нем определяются одинаковые атрибуты у всех спортсменов такого вида. Далее в классах-наследниках уже для каждой страны переопределяются те атрибуты, которые должны иметь разные значения для каждой страны. В классе AthleteFactory мы объявляем виртуальными функции создания спортсменов, и далее уже пишем фабрику, которая служит для создания продуктов всех основных типов, для каждой конкретной страны. Это удобно, так как мы создаем взаимосвязанные объекты, сам процесс порождения объектов скрывается. Например, в классе Game мы как раз создаем спортсменов. Но мы не пишем, что хотим создать Российского фигуриста, пишем просто «создать фигуриста» так как программа уже сама поняла, из какого государства будет спортсмен.