В этом проекте мы будем работать над базовой программой управления данными посетителей фитнес-центра. Он имеет три филиала (*Club Mercury, Club Neptune и Club Jupiter*). Также посетители делятся на две категории: посетители одного клуба (*Single Club Member*) и посетители нескольких клубов (*Multi Club Members*).

Посетителю одного клуба разрешен доступ только к своему клубу. С другой стороны, посетителям нескольких клубов доступны все три клуба. Размер членского взноса зависит от того, пользуется ли посетитель услугами одного или нескольких клубов. Для посетителей одного клуба размер оплаты также зависит от того, к какому клубу он имеет доступ.

***Класс Member***

Наконец, посетители нескольких клубов получают премиальные баллы за вступление в клуб. При регистрации они получают 100 баллов, которые могут использоваться для получения сувениров и напитков. Наша программа не будет реализовать процесс получения, а ограничится простым начислением 100 баллов на счет посетителя нескольких клубов.

Приложение использует CSV-файл для хранения информации о каждом посетителе. При каждом запуске приложения информация читается из CSV-файла и передается в LinkedList. Когда мы добавляем или удаляем посетителя из LinkedList, CSV-файл обновляется.

Приложение состоит из шести классов и одного интерфейса.

**Классы**

*Member*

*SingleClubMember extends Member*

*MultiClubMember extends Member*

*MembershipManagement*

*FileHandler*

*JavaProject*

**Интерфейс**

*Calculator*

***Member***

Начнем с класса Member. Этот класс содержит основную информацию о посетителях. Он служит родительским классом, от которого будут наследовать два других класса.

Поля

Класс содержит четыре приватных поля: memberType, memberID, name и fees с типом char, int, String и double соответственно.

Конструктор

Затем создадим конструктор для класса Member. Класс имеет один конструктор с четырьмя параметрами pMemberType(char), pMemberID (int), pName (String) и pFees (double).

Внутри конструктора четыре параметра присваиваются соответствующим полям.

*Метод toString() для класса*

Метод toString() — предварительно написанный метод класса Object, который возвращает строку с представлением объекта.

Тем не менее реализация toString() по умолчанию не очень информативна.

По этой причине этот метод обычно переопределяется в производных классах.

Желательно добавить аннотацию @Override — тем самым вы сообщаете компилятору, что переопределяете метод.

Метод решает только одну задачу: он возвращает строку с информацией о конкретном посетителе.

Например, метод может вернуть строку:

"S, 1, Yvonne, 950.0"

где S, 1, Yvonne и 950.0 — значения полей memberType, memberID, name и fees соответственно для данного посетителя.

Поля

private char memberType;

private int memberID;

private String name;

private double fees;

Конструктор

Member(char pMemberType, int pMemberID, String pName,

double pFees)

Методы

public void setMemberType(char pMemberType)

public void setMemberID(int pMemberID)

public void setName(String pName)

public void setFees(double pFees)

public char getMemberType()

public int getMemberID()

public String getName()

public double getFees()

public String toString()

*SingleClubMember*

Поля

Класс SingleClubMember содержит одно приватное поле типа int с именем club.

Конструктор

Класс имеет один конструктор с пятью параметрами: pMemberType (char), pMemberID (int), pName (String), pFees (double) и pClub (int). Внутри конструктора сначала ключевое слово super используется для вызова конструктора в родительском классе. Родительскому конструктору передаются значения pMemberType, pMemberID, pName и pFees. Затем параметр pClub присваивается полю club.

Методы

Метод toString() для этого класса. Этот метод объявлен открытым. Он похож на метод toString() в родительском классе, но выводит дополнительную информацию — клуб, в который ходит посетитель.

Поля

private int club

Конструктор

SingleClubMember(char pMemberType, int pMemberID, String pName, double pFees, int pClub)

Методы

public void setClub(int pClub)

public int getClub()

public String toString()

*MultiClubMember*

Поля

Класс MultiClubMember содержит одно поле — приватное поле типа int с именем membershipPoints.

Конструктор

Этот конструктор очень похож на конструктор SingleClubMember. Он тоже получает 5 параметров. Главное отличие заключается в том, что последним параметром конструктора является pMembershipPoints(int) вместо pClub.

Поля

private int membershipPoints

Конструктор

MultiClubMember(char pMemberType, int pMemberID,

String pName, double pFees, int pMembershipPoints)

Методы

public void setMembershipPoints(int pMembershipPoints)

public int getMembershipPoints()

public String toString()

*Calculator*

Интерфейс Calculator является функциональным, а следовательно, содержит только один абстрактный метод calculateFees(). Этот метод получает один параметр clubID и возвращает значение double. Попробуйте объявить этот метод самостоятельно. Вместе с методом будет готов и интерфейс.

Метод

double calculateFees(T clubID)

*FileHandler*

Класс содержит три открытых метода — readFile(), appendFile() и overWriteFile().

readFile()

Этот открытый метод не получает параметра и возвращает коллекцию LinkedList с объектами Member. Метод readFile() читает данные из CSV-файла с подробной информацией обо всех посетителях. Затем каждый посетитель добавляется в коллекцию LinkedList. Метод возвращает LinkedList. Данные в CSV-файле members.csv имеют следующий формат:

*Тип посетителя*, *идентификатор посетителя*, *имя посетителя*, *оплата*, *идентификатор клуба* для посетителей одного клуба и формат *Тип посетителя*, *идентификатор посетителя*, *имя посетителя*, *оплата*, *баллы* для посетителей многих клубов.

appendFile()

Метод appendFile() присоединяет новую строку к файлу members.csv каждый раз, когда добавляется новый посетитель. Он получает параметр String с именем mem и не возвращает никакого значения.

overwriteFile()

Метод overwriteFile() получает параметр LinkedList<Member> с именем m и ничего не возвращает. Этот метод вызывается каждый раз, когда вы хотите удалить посетителя из клуба. При удалении посетителя необходимо обновить CSV-файл. К сожалению, в Java несуществует метода для простого удаления строки из файла. В файл можно либо записывать данные, либо присоединять их, но возможность удаления не предусмотрена.

Следовательно, нужно создать временный файл.

Каждый раз, когда вы хотите исключить посетителя из клуба, он сначала удаляется из LinkedList. Затем объект LinkedList передается в аргументе метода overwriteFile().

В методе overwriteFile() создается временный файл members.temp, в который записываются все данные из LinkedList. Обратите внимание: данные не записываются прямо в файл members.csv. Это делается для того, чтобы предотвратить возможные ошибки из-за повреждения файла. Если все прошло нормально, исходный файл members.csv удаляется, а members.temp переименовывается в members.csv.

Методы

public LinkedList<Member> readFile()

public void appendFile(String mem)

public void overwriteFile(LinkedList<Member> m)

*MembershipManagement*

Класс MembershipManagement занимает центральное место в нашей программе. Этот класс реализует операции добавления и удаления посетителей. В нем также присутствует метод для вывода информации о посетителе.

Сначала мы объявим объект в классе объект Scanner и используем его для чтения пользовательского ввода.

Назовем этот объект reader и объявим его с ключевыми словами final и private: final private Scanner reader = new Scanner(System.in); Объект reader объявляется с ключевым словом final, потому что мы не будем присваивать ему новую ссылку позднее в коде. Также он объявляется с ключевым словом private, потому что он будет использоваться только в классе MembershipManagement.

getIntInput()

Первый метод называется getIntInput(). Этот метод вызывается каждый раз, когда любой метод в классе MembershipManagement использует вызов System.out.println() для того, чтобы запросить у пользователя значение int. Метод пытается прочитать введенное значение int. Если значение, введенное пользователем, не относится к типу int, метод продолжает запрашивать у пользователя данные, пока не получит допустимое значение. Он не получает параметров и возвращает значение int.

printClubOptions()

Метод printClubOptions(). Этот метод относительно тривиален. Он не получает параметров и не

возвращает значения. Метод просто использует серию вызовов System.out.println() для вывода следующего текста:

1) Club Mercury

2) Club Neptune

3) Club Jupiter

4) Multi Clubs

getChoice()

Далее необходимо запрограммировать еще четыре метода

этого класса. Все четыре метода являются открытыми.

Первый — открытый метод getChoice(). Он не получает

параметров и возвращает int.

Метод getChoice()определяет локальную переменную int с именем choice и использует серию команд System.out.println() и System.out.print() для вывода следующего текста:

WELCOME TO OZONE FITNESS CENTER

================================

1) Add Member

2) Remove Member

3) Display Member Information

Please select an option (or Enter -1 to quit):

Затем он вызывает метод getIntInput() для получения данных от пользователя ввода и присваивает ввод переменной choice.

addMembers()

Метод addMembers() получает коллекцию LinkedList объектов Member и добавляет в нее данные нового посетителя. После добавления данных в LinkedList он возвращает строку с информацией о добавленном посетителе.

Метод содержит 7 локальных переменных:

String name;

int club;

String mem;

double fees;

int memberID;

Member mbr;

Calculator<Integer> cal;

Следует отметить, что последняя переменная представляет собой ссылку на интерфейс Calculator.

.

Получение информации о доступности клубов

Сначала вызывается метод printClubOptions(). Затем пользователю предлагается ввести идентификатор клуба, доступного для посетителя. Наконец, вызов метода getIntInput() читает пользовательский ввод и присваивает значение локальной переменной club.

Вычисление идентификатора посетителя

Перейдем к вычислению идентификатора для нового

посетителя. Идентификатор — автоматически увеличивающееся число, которое присваивается каждому новому посетителю.

Метод getMemberID(), запрограммированный нами в классе Member, для получения поля memberID этого элемента.

Если коллекция LinkedList пуста, значение memberID равно 1. Это означает, что добавляемый посетитель является первым в LinkedList.

Добавление посетителя в LinkedList

При добавлении посетителя учитывается значение переменной club. Если значение club равно 1, 2 или 3, этот посетитель ходит в один клуб. Если значение равно 4, этот посетитель ходит в несколько клубов.

Добавление посетителя одного клуба

Сначала необходимо вычислить оплату для посетителя одного клуба. Для этого мы воспользуемся выражением для реализации метода calculateFees() в запрограммированном ранее интерфейсе Calculator. Метод получает единственный параметр — clubID.

Если идентификатор клуба равен 1, возвращается значение 900. Если он равен 2 или 3, возвращаются значения 950 и 1000 соответственно. Если оно не равно 1, 2 или 3, возвращается значение –1.

После добавления нового посетителя метод генерирует строку, представляющую нового посетителя, и присваивает ее локальной переменной

removeMember()

Этот метод не возвращает никакого значения. Он получает коллекцию LinkedList объектов Member и удаляет данные посетителя из LinkedList.

Внутри этого метода мы сначала объявим локальную переменную int с именем memberID.

Затем пользователю будет предложено ввести идентификатор посетителя, данные которого вы хотите удалить.

Затем метод getIntInput() читает введенное значение и присваивает его переменной memberID.

Выполнение метода должно быть завершено при обнаружении подходящего идентификатора посетителя.

printMemberInfo()

Этот метод очень похож на метод removeMember(). Он тоже получает коллекцию LinkedList объектов Member и не возвращает никакого значения.

Внутри метода все происходит почти так же, как для метода removeMember(), но вместо удаления посетителя методом remove() метод toString() используется для получения информации о конкретном посетителе.

MembershipManagement

Поля

final private Scanner reader

Методы

private int getIntInput()

private void printClubOptions()

public int getChoice()

public String addMembers(LinkedList<Member> m)

public void removeMember(LinkedList<Member> m)

public void printMemberInfo(LinkedList<Member> m)

Добавление посетителя нескольких клубов

Перейдем к блоку else, в котором в LinkedList должен быть добавлен посетитель нескольких клубов. Сначала необходимо написать лямбда-выражение, вычисляющее оплату для посетителя нескольких клубов. Выражение должно вернуть значение 1200, если переданный аргумент равен 4. В противном случае должно возвращаться значение –1. Написанное лямбда-выражение будет использоваться для вычисления оплаты посетителя нескольких клубов, которая затем будет присвоена переменной fees.

Затем необходимо создать объект MultiClubMember. Напомню, что конструктор MultiClubMember получает 5 параметров: char pMemberType, int pMemberID, String pName, double pFees и int pMembershipPoints. Для создания объекта MultiClubMember конструктору передаются символьное значение 'M', переменные memberID, name и fees, а также значение 100. Созданный объект присваивается переменной mbr, которая затем добавляется в LinkedList m. Далее метод должен сгенерировать строку для представления нового посетителя и присвоить ее mem. Наконец, на экран выводится сообщение об успешном добавлении нового посетителя нескольких клубов.

*JavaProject*

Класс JavaProject содержит единственный метод main().

В методе main() определяются пять переменных:

String mem;

MembershipManagement mm;

FileHandler fh;

LinkedList<Member> members;

int choice;

Объект mm используется для вызова метода getChoice() в классе MembershipManagement. Метод выводит список вариантов и возвращает вариант, выбранный пользователем, вызывающей стороне.

Доступны следующие варианты:

1) Add Member

2) Remove Member

3) Display Member Information

Также пользователь может ввести –1 для завершения программы.

Если выбран вариант 1, метод addMembers() класса MembershipManagement используется для добавления данных посетителя в LinkedList. Метод addMembers() предлагает пользователю ввести данные нового посетителя; эти данные используются для обновления переданной коллекции LinkedList. Кроме того, он возвращает строку с представлением добавленного посетителя. Эта строка присваивается локальной переменной mem. Затем метод appendFile()

в классе FileHandler используется для добавления посетителя в файл members.csv. Этот метод не возвращает никакого значения.

Если выбран вариант 2, метод removeMember() используется для удаления посетителя, а метод overwriteFile() обновляет CSV-файл.

Если выбран вариант 3, метод printMemberInfo() выводит информацию о посетителе.