Документация за проект по Обектно Ориентирано Програмиране – 1 част

Проект 14: Билети

Да се напише програма, реализираща информационна система, която обслужва билетна каса. Програмата съхранява и обработва необходимите данни във файл. Представленията се играят в няколко зали, всяка от които има номер, брой редове и брой места на всеки ред. Залите са предварително зададени.

Изготвил : Владислав Красимиров Димитров

Факултетен номер : 23621677

Клас: Menu

Описание

Класът Menu управлява потребителския интерфейс на конзолна система за продажба на билети. Той отговаря за:

Регистриране на наличните команди (като book, unbook, buy, help и други)

Приемане и обработка на потребителския вход в безкраен цикъл

Изпълнение на съответната команда според въведения от потребителя текст

Класът следва принципа на отделяне на логиката на командите от интерфейса, използвайки шаблона (патърна) Command. Това позволява лесно разширяване на системата с нови команди без да се променя класът Menu.

Полета (променливи)

Map<String, Command> commands — Съхранява всички команди, асоциирани с техните имена (например "book" -> BookCommand).

Scanner scanner — Обект за четене на вход от конзолата.

TicketSystem system — Основната логика на системата за управление на билети. Командите работят с този обект.

Методи

public Menu()

Конструкторът създава обект Menu и регистрира всички команди в хеш-мапата commands. Командите се създават с препратка към един и същ обект TicketSystem, което осигурява целостта на данните.

public void run()

Основен метод, който стартира интерфейса. Върти се в безкраен цикъл, изчаква въвеждане на команда от потребителя, парсва текста и търси съответна команда в commands. Ако я намери — изпълнява execute(), ако не — извежда съобщение за непозната команда.

Обработва изключенията, за да не спира програмата при грешки.

Защо е използван този подход?

Шаблонът Command разделя командите в отделни класове, което подобрява поддръжката и разширяемостта.

Регистрацията на командите в HashMap позволява динамично търсене без големи if-else или switch конструкции.

Конзолният интерфейс е прост и бърз за реализиране, подходящ за прототип или малка система.

Възможни алтернативи

Switch-case логика: Вместо Command Pattern, можеше да се използва цикъл с switch върху командите. Това е по-прост подход, но затруднява добавянето на нови команди и увеличава риска от грешки.

Графичен потребителски интерфейс: Използване на JavaFX или Swing за визуално меню, което е по-приятно за потребителя, но изисква повече усилия.

Конфигурационни файлове за команди: Зареждане на командите динамично от конфигурация, вместо твърдо кодиране в конструктора.

Клас: TicketSystem

Описание

Класът TicketSystem представлява централизирана система за управление на зали, събития и билети. Той поддържа добавяне и управление на зали, събития и билетни кодове, предоставя функции за проверка на свободни места, регистриране и проверка на билетни кодове, както и записване и зареждане на състоянието на системата.

Полета

private final Map<String, Hall> halls

Съхранява залите в системата. Ключът е името на залата, стойността е обект Hall, който съдържа информация за редове и места.

private final Map<String, Map<String, Event>> events

Съхранява събитията, организирани по дата и име на събитието. Външният ключ е дата (като String), вътрешният ключ е име на събитието, а стойността е обект Event.

private final Map<String, String> codes

Съхранява билетни кодове и свързаната с тях информация за мястото.

Конструктор

TicketSystem()

Инициализира системата с три предварително дефинирани зали (Hall1, Hall2, Hall3) с различен брой редове и места.

Методи

Управление на зали

void addHall(String name, int rows, int seatsPerRow)

Добавя нова зала с посочено име, брой редове и места на ред.

Hall getHall(String name)

Връща обект Hall по зададено име или null ако залата не съществува.

Управление на събития

void addEvent(String date, String hallName, String name) throws Exception

Добавя ново събитие за конкретна дата и зала. Ако залата не съществува или е заета на същата дата, хвърля изключение.

Event getEvent(String date, String name)

Връща обект Event по дата и име или null ако не е намерено.

Информация за събития и места

void printFreeSeats(String date, String name)

Извежда списък с всички свободни места за дадено събитие (ред и място).

void printBookings(String date, String name)

Извежда информация за всички резервации за дадена дата и/или събитие. Ако параметрите са null, обхваща всички.

Управление на билетни кодове

void checkCode(String code)

Проверява дали даден билетен код е валиден и извежда информация за него.

void registerCode(String code, String info)

Регистрира нов билетен код с информация за мястото.

Запис и зареждане

void save()

Записва цялото състояние на системата в сериализиран файл save.dat.

void saveAs(String filename)

Записва състоянието в текстов файл с името, подадено като параметър.

void open(String filename)

Метод за зареждане от файл (в момента само заглушка, трябва да се имплементира).

void close()

Изчиства всички събития и кодове, ефективно рестартирайки системата.

Отчети

void generateReport(String from, String to, String hall)

Генерира отчет за всички събития в даден период и (опционално) по зала, като извежда броя на продадените билети.

Защо се използва този клас?

Централизира цялата логика за управление на зали, събития и билети в една система.

Осигурява удобен API за добавяне, търсене и управление на събития и зали.

Позволява съхранение и възстановяване на състоянието, както и проверка на валидност

на билетни кодове.

Използва вътрешни структури от данни за бърз достъп и проверка.

Клас: Ticket

Описание

Класът Ticket представлява модел на билет с информация за статуса на резервация, закупуване и допълнителна бележка. Той съдържа основни булеви флагове за статус и текстово поле за бележка, което позволява съхраняване на допълнителна информация, свързана с билета.

Полета

private boolean booked

Флаг, указващ дали билетът е резервиран.

private boolean bought

Флаг, указващ дали билетът е закупен.

private String note

Текстово поле, което може да съдържа бележка или коментар, свързан с билета.

Методи

Статус резервация

boolean isBooked()

Връща true, ако билетът е резервиран, в противен случай false.

void setBooked(boolean booked)

Задава статуса на резервация. Подава се true за резервиран билет, false за не резервиран.

Статус покупка

boolean isBought()

Връща true, ако билетът е закупен, иначе false.

void setBought(boolean bought)

Задава статуса на закупуване на билета. true ако е закупен, false ако не е.

Бележка

String getNote()

Връща бележката към билета като текст.

void setNote(String note)

Задава текстова бележка към билета.

Защо се използва този клас?

Представлява един билет в системата с основна информация за резервация и покупка.

Позволява лесно проследяване на статуса на билета.

Позволява добавяне на бележки, които могат да съдържат важна допълнителна информация (например специални условия, клиентски коментари и др.).

Клас: Hall

Описание

Класът Hall представлява модел на зала в системата за управление на билети и събития. Той съдържа основна информация за залата — нейното име, броя на редовете и броя на местата във всеки ред.

Полета

private final String name

Името на залата. Идентифицира конкретна зала в системата.

private final int rows

Броят на редовете в залата.

private final int seatsPerRow

Броят на местата във всеки ред.

Конструктор

Hall(String name, int rows, int seatsPerRow)

Създава нов обект залата с определено име, брой редове и места на ред. Тези параметри се задават веднъж при създаване и не могат да се променят след това (поради final полетата).

Методи

String getName()

Връща името на залата.

int getRows()

Връща броя на редовете в залата.

int getSeatsPerRow()

Връща броя на местата във всеки ред.

Защо се използва този клас?

Абстрахира и централизира информацията за зала, което позволява лесно управление и справяне със структурните характеристики на залата.

Служи като основа за разпределяне на билети и места на събития.

Позволява системата да управлява различни зали с различен капацитет.

Клас: Event

Описание

Класът Event представлява събитие, което се провежда на определена дата в конкретна зала. Той съдържа основна информация за събитието – име, дата, зала, както и управление на билетите, свързани с него.

Полета

private final String name

Името на събитието.

private final String date

Датата на провеждане на събитието, представена като низ (например "2025-05-22").

private final Hall hall

Залата, в която се провежда събитието.

private final Map<String, Ticket> tickets

Колекция от билети за събитието, където ключът е код на мястото (напр. "R1S5" – ред 1, място 5), а стойността е обект Ticket.

Конструктор

Event(String name, String date, Hall hall)

Създава нов обект събитие с име, дата и посочена зала.

Методи

String getName()

Връща името на събитието.

String getDate()

Връща датата на събитието.

Hall getHall()

Връща обекта зала, в която се провежда събитието.

void addTicket(String seatCode, Ticket ticket)

Добавя билет към събитието, като го асоциира с код на мястото. Позволява по-късно проследяване и управление на билетите.

int countBoughtTickets()

Изчислява броя на всички билети, които са закупени (чрез проверка на статуса bought във всеки Ticket).

Защо се използва този клас?

Позволява логическа групировка на билети според събитие, дата и зала.

Улеснява управлението на билети — резервиране, закупуване и проследяване.

Предоставя методи за анализ на продажбите (напр. броя закупени билети).

Създава ясна структура за разширяване и интегриране със системата за билети и зали.

Интерфейс: Command

Описание

Интерфейсът Command дефинира команден шаблон (Command Pattern) за системата за билети. Той служи като контракт, който всички конкретни команди трябва да имплементират. По този начин се осигурява унифициран подход за изпълнение на операции в системата чрез метод, приемащ аргументи.

Цел и употреба

Позволява капсулиране на заявка или операция като обект.

Отделя клиента, който иска дадена операция, от реализацията ѝ.

Поддържа лесно добавяне на нови команди без промяна на съществуващия код.

Улеснява регистрирането и управлението на множество команди, например в колекция (като HashMap), за бърз достъп и изпълнение.

Метод

void execute(String[] args)

Изпълнява конкретната команда, като използва масив от аргументи, подадени от потребителя.

args — аргументите, които съдържат необходимата информация за изпълнението (например параметри, опции и др.).

Защо се използва

Позволява гъвкава и разширяема архитектура на приложението.

Отделя логиката на отделните команди в самостоятелни класове.

Лесно се интегрира с менюта, интерфейси и обработващи механизми.

Съдейства за поддръжка и тестване на отделните операции независимо.

Клас: AddEventCommand

Описание

Класът AddEventCommand имплементира интерфейса Command и реализира командата за добавяне на ново събитие в системата за билети. Той служи като обвивка, която обработва входните аргументи от потребителския интерфейс и предава заявката към бизнес логиката, дефинирана в класа TicketSystem.

Полета

private final TicketSystem system

Инстанция на системата за билети, с която командата работи. Използва се за извикване на методи за добавяне на събития и други операции.

Конструктори

public AddEventCommand(TicketSystem system)

Конструктор, който инициализира командата с обект от тип TicketSystem. Това позволява на командата да оперира върху конкретна инстанция на системата.

Методи

public void execute(String[] args)

Изпълнява логиката за добавяне на събитие.

Проверява дали са подадени минимум три аргумента: дата, зала и име на събитието.

Ако аргументите са недостатъчни, извежда помощно съобщение за правилния формат.

Извлича датата (date), името на залата (hall) и името на събитието (name), като събитийното име може да съдържа интервали и се събира чрез String.join.

Опитва да добави събитието в системата чрез system.addEvent().

Ако възникне изключение (например залата е заета или не съществува), отпечатва съобщението за грешка.

Защо се използва

Реализира конкретна команда според Command Pattern, което позволява лесно добавяне на нови операции без промяна в основния код.

Отделя обработката на потребителския вход и бизнес логиката на системата.

Позволява гъвкаво управление и разширение на командите в системата.

Клас: BookCommand

Описание:

Класът BookCommand имплементира командата за резервиране на място за събитие в системата за билети. Той е част от командния шаблон (Command Pattern), който осигурява отделна логика за всяка команда в приложението.

Защо се използва:

Този клас инкапсулира логиката за обработка на командата „резервиране на място“ (book). Това разделяне на отговорностите улеснява поддръжката, разширяването и тестването на приложението, тъй като всяка команда има собствен клас с ясна и отделена функционалност.

Основни характеристики:

Съдържа референция към системата за билети (TicketSystem), чрез която извършва операции върху данните.

Приема аргументи от потребителя под формата на масив от низове (String[] args), като очаква най-малко 5 аргумента: ред, място, дата, име на събитие и бележка.

Проверява дали са подадени достатъчно аргументи и в противен случай извежда помощна информация за правилния формат на командата.

Засега методът execute не реализира самата логика за резервация, а само информира, че функцията все още не е имплементирана.

Как работи:

При извикване на метода execute се прави проверка за броя на аргументите. Ако аргументите са недостатъчни, се показва съобщение с инструкция за правилната употреба на командата. Ако аргументите са достатъчни, следващата стъпка би била извършване на реалната резервация чрез системата за билети, като се използват подадените параметри. В настоящата имплементация обаче тази функционалност все още не е реализирана.

Клас: BookingsCommand

Описание:

Класът BookingsCommand имплементира командата за извеждане на всички резервации в системата за билети. Тази команда позволява показване на резервации според дата или според дата и име на събитие.

Конструктор:

Приема инстанция на TicketSystem, която осигурява логиката за достъп и управление на резервациите.

Метод execute(String[] args):

Приема аргументи от главното меню в два възможни формата:

Само дата (<дата>) — изкарва всички резервации за тази дата.

Дата и име на събитие (<дата> <име на събитие>) — изкарва резервациите само за това конкретно събитие.

Извиква метод printBookings(date, name) от TicketSystem, който извежда съответната информация.

Клас: CheckCommand

Описание:

Класът CheckCommand имплементира командата за проверка на код, свързан с резервация или покупка на билет в системата за билети. Той е част от командния шаблон (Command Pattern), който предоставя отделен клас за всяка конкретна команда в приложението.

Защо се използва:

Този клас инкапсулира логиката за обработка на командата „провери код“ (check). По този начин се постига по-добра организация и модулност на кода, което улеснява поддръжката, разширяемостта и тестването на системата.

Основни характеристики:

Поддържа връзка с обект от типа TicketSystem, който извършва реалната проверка на подадения код.

Приема аргументи от потребителя във вид на масив от низове (String[] args), като очаква точно един аргумент – самия код.

Проверява дали е подаден необходимият брой аргументи. При липса на аргумент извежда указание за правилната употреба на командата.

Как работи:

При извикване на метода execute се проверява дали е подаден поне един аргумент. Ако не е, се показва съобщение с правилната синтактична структура на командата. Ако кодът е подаден коректно, методът делегира обработката му на системата за билети (TicketSystem), която извършва проверката и извежда резултата.

Клас: CloseCommand

Описание:Класът CloseCommand имплементира командата за затваряне на текущото състояние на системата за билети. Той е част от шаблона Command (Command Pattern), който осигурява разделение на логиката за всяка команда в отделен клас.

Защо се използва:

Този клас инкапсулира логиката за командата „затвори“ (close), която изчиства текущо заредените събития и билети. Това разделение подобрява четимостта и поддръжката на кода, като улеснява добавянето на нови команди без да се модифицира основната програма.

Основни характеристики:

Съдържа референция към основния обект TicketSystem, чрез който се извършва операцията по затваряне.

Не изисква аргументи от потребителя.

Извиква метода close() на системата, който изчиства всички текущо заредени събития и билети.

Как работи:

При извикване на метода execute се игнорират подадените аргументи (ако има такива), и директно се извиква методът close() на TicketSystem, който изчиства всички данни от текущата сесия. Командата е полезна при рестартиране или изход от програмата, когато потребителят иска да изчисти текущото състояние.

Клас: ExitCommand

Описание:

Класът ExitCommand реализира командата за изход от системата за билети. Той е част от шаблона Command (Command Pattern), като предоставя самостоятелна логика за прекратяване на работата на програмата.

Защо се използва:

Този клас инкапсулира поведението на командата „изход“ (exit), която служи за приключване на работата на приложението. Разделянето на логиката в отделен клас улеснява управлението на командите и добавянето на нови функционалности.

Основни характеристики:

Не изисква аргументи от потребителя.

Извежда съобщение при изпълнение, за да информира потребителя, че системата ще се затвори.

Прекратява изпълнението на програмата чрез извикване на System.exit(0).

Как работи:

При извикване на метода execute, командата игнорира подадените аргументи (ако има такива), показва съобщение „Exiting Ticket System.“ в конзолата и прекратява изпълнението на програмата. Това е крайната команда в системата, която се използва за затваряне на приложението.

Клас: FreeSeatsCommand

Описание:

Класът FreeSeatsCommand реализира командата за показване на свободните места за конкретно събитие в системата за билети. Тази команда е част от командния шаблон (Command Pattern), като отделя логиката за извеждане на свободните места в собствен клас.

Защо се използва:

Разделянето на командите на отделни класове позволява по-добра поддръжка и разширяемост на системата. FreeSeatsCommand инкапсулира конкретната операция за извличане и показване на свободните места за дадена дата и събитие.

Основни характеристики:

Съдържа препратка към TicketSystem, която държи информация за събитията и залите.

Приема аргументи под формата на масив от низове, като изисква поне два: дата и име на събитието.

Проверява дали са подадени достатъчно аргументи, и при липса изписва помощна информация.

Използва метода printFreeSeats на системата за билети, за да изведе свободните места.

Как работи:

При извикване на метода execute, първо се проверява дали потребителят е подал необходимите аргументи. Ако аргументите са по-малко от два, се извежда съобщение с инструкция за правилна употреба. Ако аргументите са валидни, се вземат датата и името на събитието (име, което може да съдържа няколко думи), след което се извиква методът printFreeSeats на TicketSystem, който отпечатва свободните места в залата за съответното събитие и дата.

Клас: HelpCommand

Описание:

Класът HelpCommand реализира командата за показване на помощна информация с наличните команди в системата за билети. Той служи като справочник за потребителя относно синтаксиса и възможностите на приложението.

Защо се използва:

Чрез отделна команда за помощ се улеснява ориентацията на потребителя в системата, като се предоставя ясен списък с всички поддържани команди и техните параметри. Това подобрява удобството и намалява грешките при използване на приложението.

Основни характеристики:

Не изисква аргументи за работа.

При изпълнение извежда на стандартния изход списък с всички поддържани команди и синтаксиса им.

Как работи:

Методът execute просто отпечатва предварително дефиниран списък от команди, като по този начин помага на потребителя да се ориентира как да използва системата. Аргументите, подадени към командата, се игнорират.

Клас: OpenCommand

Описание:

Класът OpenCommand реализира командата за отваряне на файл със запазени данни на системата за билети. Тази команда позволява зареждане на предишно запазено състояние на системата от файл.

Защо се използва:

Чрез отделна команда за отваряне на файл се осигурява възможност за възстановяване на данни и продължаване на работа от последния записан момент, като се поддържа състоянието на системата между сесиите.

Основни характеристики:

Приема като аргумент името на файла за зареждане.

Ако аргументът липсва или е невалиден, извежда помощно съобщение за правилен синтаксис.

При грешка по време на зареждане (например файлът не съществува или е повреден) извежда съобщение с описание на проблема.

Как работи:

Методът execute проверява дали е подадено име на файл. Ако не, показва съобщение за правилна употреба. Ако аргументът е наличен, опитва да зареди състоянието на системата чрез метода open на TicketSystem. При възникнала грешка се уловя изключението и се отпечатва съобщение с причината.

Клас: ReportCommand

Описание:

Класът ReportCommand реализира командата за генериране на отчет за продадени билети в даден период от време. Командата може да ограничи отчета и по конкретна зала, ако е посочена.

Защо се използва:

Тази команда позволява да се извлече статистическа информация за продадените билети за избран период, което помага за анализ и управление на продажбите и за планиране на бъдещи събития.

Основни характеристики:

Приема като аргументи начална и крайна дата (задължителни).По избор може да се подаде и име на зала, за да се филтрират събитията само в нея.

Ако не са подадени поне начална и крайна дата, извежда помощна информация за правилния синтаксис.

Как работи:

Методът execute проверява броя на аргументите. Ако са по-малко от две, показва съобщение с инструкции за правилна употреба. Ако аргументите са валидни, извиква метода generateReport на системата за билети с подадените параметри — начална дата, крайна дата и по избор име на зала.

Клас: SaveAsCommand

Описание:

Класът SaveAsCommand реализира командата за записване на текущото състояние на системата за билети в нов файл. Тази команда позволява запазване на данните под различно име на файл.

Защо се използва:

Чрез този клас се осигурява възможност потребителят да съхрани текущото състояние на системата в отделен файл, различен от основния, което улеснява управлението на множество версии или архивиране.

Основни характеристики:

Приема като аргумент име на файл, в който да бъде записано състоянието.

Ако не е подадено име на файл, извежда помощна информация за правилната употреба.

При възникване на грешка при записването извежда съобщение с описание на проблема.

Как работи:

При извикване на метода execute се проверява дали е подадено име на файл. Ако не, се извежда съобщение с инструкция за правилна употреба. Ако аргументът е валиден, се извиква методът saveAs на системата за билети с подаденото име на файл. При евентуална грешка по време на записа, се показва съобщение с описанието ѝ.

Клас: SaveCommand

Описание:

Класът SaveCommand реализира командата за записване на текущото състояние на системата за билети в основния файл за съхранение. Това позволява съхраняване на всички направени промени по данните.

Защо се използва:

Чрез този клас се инкапсулира логиката за стандартното запазване на състоянието на системата. Така се осигурява удобен и централизирано контролиран начин за записване без нужда от указване на файл.

Основни характеристики:

Не приема аргументи при изпълнение.

При възникване на грешка по време на записване извежда съобщение с описание на проблема.

Как работи:

При извикване на метода execute се извиква методът save() на системата за билети. Ако възникне изключение, се хваща и се извежда съобщение с информация за грешката.

Клас: UnbookCommand

Описание:

Класът UnbookCommand реализира командата за освобождаване (отмяна на резервация) на билет за конкретно място, дата и събитие в системата за билети. Тази команда трябва да премахне направената резервация за зададеното място, като по този начин освобождава мястото за повторна продажба.

Защо се използва:

Командата се използва, за да позволи на потребителя да отмени направена резервация на билет, например при отказ или промяна на плановете. Тя осигурява удобен и централизирано контролиран начин за управление на резервациите в системата.

Основни характеристики:

Приема четири аргумента: ред, място, дата и име на събитие.

Проверява коректността на подадените аргументи и при липса на достатъчно такива извежда съобщение за правилна употреба.

В настоящата версия не реализира логиката за отмяна, а само информира, че функционалността не е имплементирана.

Как работи:

При извикване на метода execute се проверява дали са подадени достатъчно аргументи. Ако не са, се извежда помощно съобщение с правилния синтаксис на командата. Ако аргументите са коректни, в момента командата само извежда съобщение, че отмяната на резервация не е имплементирана. В бъдеще тук трябва да се добави логика, която да комуникира със системата за билети и да премахва резервацията.