รายงาน

Library by OOP

เสนอ

อาจารย์ศักดิ์ระพี ไพศาลนันทน์

จัดทำโดย

นาย ธีรเดช สุขเกษม 630910864

นาย สุทธิพงษ์ เลิศปัญญาวุฒิกุล 630910880

นางสาว อณัศยา พวงทรัพย์ 630910881

รายงานนี้เป็นส่วนหนึ่งของวิชา OBJECT-ORIENTED SYSTEM DESIGN FOR ENGINEERS รหัสวิชา 618445 มหาวิทยาลัยศิลปากร ภาคเรียนที่2 ปีการศึกษา2566

คำนำ

รายงานเล่มนี้จัดทำขึ้นเพื่อเป็นส่วนหนึ่งของวิชา OBJECT-ORIENTED SYSTEM DESIGN FOR ENGINEERS โดยมีเนื้อหาเกี่ยวกับระบบจองที่นั่งในโรงภาพยน์โดยใช้ความรู้ด้าน OOP และ PyQt6 เพื่อให้ผู้ที่ต้องการศึกษาหาความรู้ในเรื่องการเขียนโค้ด ได้ศึกษาอย่างเข้าใจเพื่อเป็นประโยชน์กับการเรียน

ผู้จัดทำหวังว่ารายงานนี้จะมีข้อมูลที่เป็นประโยชน์กับผู้อ่านหรือนักเรียน นักศึกษาที่กำลังหาข้อมูลเรื่องนี้อยู่หากมีข้อแนะนำหรือข้อผิดพลาดประการใด ผู้จัดทำขอน้อมรับไว้และขออภัยมา ณ ที่นี้ด้วย

คณะผู้จัดทำ

สารบัญ

App.py

โปรแกรมเป็นแอปพลิเคชัน PyQt6 สำหรับการจองที่นั่งในโรงหนัง โค้ดประกอบด้วยคลาส CinemaBookingApp ที่เป็นหน้าต่างหลักของแอปพลิเคชัน และมีเมทอดต่าง ๆ สำหรับการแสดงผลและจัดการกับข้อมูลการจองที่นั่งในโรงหนัง

Import Libraries

```
import sys
from PyQt6.QtCore import QSize, Qt, QRectF
from PyQt6.QtGui import QPixmap, QIcon, QPainter, QFont, QPalette
from PyQt6.QtWidgets import QApplication, QWidget, QPushButton, QGridLayout, QMessageBox , QDialog ,QLabel ,QVBoxLayout ,QLineEdit
from Cinema_system import *
```

นำเข้าโมดูลที่จำเป็นต่าง ๆ ที่ใช้ในโปรแกรม เช่น sys, PyQt6, และ Cinema_system (ซึ่งเป็นโมดูลที่ไม่ได้ให้มาด้วยในตัวอย่าง แต่คาดว่าเป็นไฟล์ที่มีคลาส CinemaHall, VIPSeat, และ RegularSeat)

สร้างคลาส CinemaBookingApp

```
class CinemaBookingApp(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Booking Cinema")
        self.showFullScreen()
        self.__image_seat_regular = QPixmap("seat.png").scaled(50, 50)  # load รูปภาพ regular
        self.__image_seat_vip = QPixmap("vip.png").scaled(50, 50)  # load รูปภาพ vip
        palette = self.palette()
        palette.setColor(QPalette.ColorRole.Window, Qt.GlobalColor.white)
        self.setPalette(palette)
        self.__cinema = CinemaHall(50) # set ค่าที่นั่ง
        self.__main_layout = QGridLayout()  # สร้าง QGridLayout สำหรับจัดวางปุ่ม
        self.__main_layout.setContentsMargins(300,300,300,300)
        self.setLayout(self.__main_layout)
        self.display_seat(self.__main_layout)

        # สร้างปุ่มปิดโปรแกรม
        self.close_button = QPushButton("Close", self)
        self.close_button.clicked.connect(self.close_app)
        self.__main_layout.addWidget(self.close_button, 11, 0, 1, 10)  # เพิ่มปุ่มลงใน main_layout

    def close_app(self):
        self.close()
```

เป็นหน้าต่างหลักของแอปพลิเคชัน ซึ่งสืบทอดมาจาก QWidget __init__() methodที่เรียกใช้เมื่อมีการสร้างวินโดว์ของคลาส CinemaBookingApp โดยกำหนดค่าเริ่มต้นต่าง ๆ เช่น ชื่อหน้าต่าง และการแสดงเต็มจอ และเรียกmethod display_seat() เพื่อแสดงที่นั่งทั้งหมดในโรงหนัง

```
# แสดงเก้าอี้ทั้งหมด เริ่มต้น  ทั้ง vip และ regular
def display_seat(self , main_layout):
    # สร้างปุ่ม Regular
    for row in range(5):
        for col in range(10):
            seat_number = row * 10 + col
            if seat_number < 30:
                    s_VIP = VIPSeat(seat_number,500)
                    self.__cinema.add_seat(s_VIP , seat_number)
                    self.draw_seat(seat_number, main_layout, row, col)
            else:
                    s_Re = RegularSeat(seat_number,200)
                    self.__cinema.add_seat(s_Re , seat_number)
                    self.draw_seat_vip(seat_number, main_layout, row, col)


# update seat
def display_update(self):
    for row in range(5):
        for col in range(10):
            seat_number = row * 10 + col
            if seat_number < 30:
                    isBooked = self.__cinema.get_number_seat(seat_number).get_booked()
                    if not isBooked:
                        self.draw_seat(seat_number, self.__main_layout, row, col)
            else:
                    isBooked = self.__cinema.get_number_seat(seat_number).get_booked()
                    if not isBooked:
                        self.draw_seat_vip(seat_number, self.__main_layout, row, col)
```

- display_seat() methodที่ใช้สำหรับการสร้างและแสดงปุ่มที่แทนที่นั่งในโรงหนัง โดยมีการใช้คลาส CinemaHall ในการจัดการที่นั่ง
- display_update() methodที่ใช้สำหรับการอัปเดตสถานะของที่นั่งในโรงหนังและการวาดปุ่มใหม่ตามสถานะปัจจุบัน

```python
# วาดหน้าจอโรงหนัง
def paintEvent(self,event):
    screen = QRectF(1300 / 2, 30, 600, 100)
    margin = 0
    movie_title = QRectF(screen.x() + margin, screen.y() + margin, screen.width() - 2 * margin, screen.height() - 2 * margin)
    painter = QPainter(self)
    painter.setRenderHint(QPainter.RenderHint.Antialiasing)
    painter.setPen(Qt.GlobalColor.black)
    painter.setFont(QFont("Arial", 20))
    painter.drawText(movie_title, Qt.AlignmentFlag.AlignHCenter, "Conjuring")
    painter.drawRect(screen)

# วาดเก้าอี้ regular
def draw_seat(self, number, layout, row, col):
    icon_button = QPushButton(QIcon(self.__image_seat_regular), '', self)
    icon_button.setStyleSheet('border: none;')
    icon_button.setIconSize(QSize(50, 50))
    layout.addWidget(icon_button, row, col)
    icon_button.clicked.connect(lambda: self.booking_seat(number))
วาดเก้าอี้ vip
def draw_seat_vip(self, number, layout, row, col):
    icon_button = QPushButton(QIcon(self.__image_seat_vip), '', self)
    icon_button.setStyleSheet('border: none;')
    icon_button.setIconSize(QSize(50, 50))
    layout.addWidget(icon_button, row + 3, col)
    icon_button.clicked.connect(lambda: self.booking_seat(number))
```

- paintEvent() methodที่ใช้สำหรับการวาดข้อความและกรอบของหน้าจอโรงหนัง
- draw_seat() และ draw_seat_vip() methodสำหรับการวาดปุ่มที่แทนที่นั่งในโรงหนัง โดยรับพารามิเตอร์ที่ต้องการในการสร้างปุ่ม

```python
# ui เมื่อกดคลิกที่ปุ่ม
def booking_seat(self, number):
    if number >= 30:
        self.ui_booking('VIP' , number , 500)
    else:
        self.ui_booking('Regular', number , 200)

# ui เมื่อกดคลิกที่ปุ่ม เพื่อรับค่าการจอง
def ui_booking(self,seat_type , number , price):
    message = f'{seat_type} Seat'

    dialog = QDialog(self)
    dialog.setWindowTitle(message)

    layout = QVBoxLayout()

    name_label = QLabel("Name:")
    layout.addWidget(name_label)
    name_input = QLineEdit()
    layout.addWidget(name_input)

    price_label = QLabel("Price:")
    layout.addWidget(price_label)
    price_input = QLineEdit()
    price_input.setText(f'{price}')  # กำหนดค่า default เป็น "100"
    price_input.setReadOnly(True)  # ตั้งค่าเป็นค่าคงที่
    layout.addWidget(price_input)
```

```python
seat_label = QLabel("Seat_Number:")
layout.addWidget(seat_label)
seat_input = QLineEdit()
seat_input.setText(f'{number}')  # กำหนดค่า default เป็น "100"
seat_input.setReadOnly(True)  # ตั้งค่าเป็นค่าคงที่
layout.addWidget(seat_input)

btn_ok = QPushButton("OK")
btn_ok.clicked.connect(lambda: self.on_ok_booking(dialog, number , name_input.text()))
layout.addWidget(btn_ok)

dialog.setLayout(layout)
dialog.exec()
```

- booking_seat() methodที่เรียกเมื่อผู้ใช้คลิกที่ปุ่มที่นั่ง และเรียกmethod ui_booking() เพื่อแสดงหน้าต่างที่ให้กรอกข้อมูลการจอง
- ui_booking() methodที่ใช้สำหรับแสดงหน้าต่างที่ให้ผู้ใช้กรอกข้อมูลการจอง เช่น ชื่อและราคา

```python
# เมื่อกดจอง
def on_ok_booking(self, dialog, number, name):
    if not name:
        self.display_messagebox('Please enter your name.')
        return
    this_seat = self.__cinema.get_number_seat(number)
    this_seat.book()
    this_seat.set_name(name)
    # ลบทุกปุ่มที่อยู่บน main_layout
    for i in reversed(range(self.__main_layout.count())):
        widget = self.__main_layout.itemAt(i).widget()
        if widget is not None:
            widget.deleteLater()
    # วาดปุ่มใหม่ตามสถานะปัจจุบันของที่นั่ง
    self.display_update()
    dialog.accept()

# เอาไว้ สร้าง messagebox
def display_messagebox(self  , message):
    # สร้าง QMessageBox
    msg_box = QMessageBox()
    msg_box.setWindowTitle("Search result")
    msg_box.setText(message)
    msg_box.setInformativeText("Please try again")
    msg_box.setIcon(QMessageBox.Icon.Warning)  # เพิ่มไอคอนเตือน
    msg_box.setStandardButtons(QMessageBox.StandardButton.Yes)
    msg_box.setDefaultButton(QMessageBox.StandardButton.Yes)
    # แสดง Message Box และรอผลลัพธ์
    msg_box.exec()
```

- on_ok_booking() methodที่เรียกเมื่อผู้ใช้กดปุ่ม OK เพื่อยืนยันการจอง โดยจะทำการอัปเดตสถานะของที่นั่ง และแสดงผลใหม่บนหน้าจอ
- display_messagebox() methodที่ใช้สำหรับแสดง MessageBox เมื่อมีข้อผิดพลาดหรือข้อความแจ้งเตือนอื่น ๆ

การทำงานในส่วนอื่นๆ

```
if __name__ == '__main__':
    app = QApplication(sys.argv)
    window = CinemaBookingApp()
    sys.exit(app.exec())
```

ส่วนที่ทำหน้าที่เรียกใช้ QApplication เพื่อเริ่มการทำงานของโปรแกรม และสร้างวินโดว์ของคลาส CinemaBookingApp จากนั้นใช้ sys.exit(app.exec()) เพื่อให้โปรแกรมทำงานได้ต่อจนกว่าจะปิดโปรแกรมลง

Cinema_sytem.py

เป็นการสร้างโครงสร้างพื้นฐานสำหรับการจัดการที่นั่งในโรงภาพยนตร์ โดยมีคลาสและอินเทอร์เฟซต่าง ๆ ดังนี้

Import Libraries

```
from abc import ABC, abstractmethod
```

นำเข้าคลาส ABC และเมธอด abstractmethod จากโมดูล abc เพื่อใช้ในการสร้างอินเทอร์เฟซแบบ abstract

สร้างคลาส Seat

```
# กำหนด Seat เป็น interface class
class Seat(ABC):
    # method Seat
    def __init__(self, number):
        self.__number = number
        self.__is_booked = False
        self.__name = None

    def get_number(self):
        return self.__number

    def book(self):
        self.__is_booked = True

    def cancel_booking(self):
        self.__is_booked = False

    def get_name(self):
        return self.__name

    # abstract method
    @abstractmethod
    def set_name(self , name):
        self.__name = name

    @abstractmethod
    def get_booked(self):
        return self.__is_booked
```

เป็นคลาสหลักที่ใช้เป็นอินเทอร์เฟซในการกำหนดคุณสมบัติและพฤติกรรมของที่นั่งในโรงภาพยนตร์ มีเมธอด เช่น book() เพื่อจองที่นั่ง, cancel_booking() เพื่อยกเลิกการจอง, set_name() เพื่อกำหนดชื่อผู้จอง, และ get_booked() เพื่อตรวจสอบสถานะการจอง

สร้างคลาส RegularSeat และ VIPSeat

```
# inherit seat
class RegularSeat(Seat):
    # method RegularSeat
    def __init__(self, number ,price):
        super().__init__(number)
        self.__price = price  # Set price for regular seat

    def get_booked(self):
        return super().get_booked()

    def set_name(self, name):
        return super().set_name(name)

    def book(self):
        super().book()

# inherit seat
class VIPSeat(Seat):
    # method VIPSeat
    def __init__(self, number , price):
        super().__init__(number)
        self.__price = price  # Set price for VIP seat

    def get_booked(self):
        return super().get_booked()

    def set_name(self, name):
        return super().set_name(name)

    def book(self):
        super().book()
```

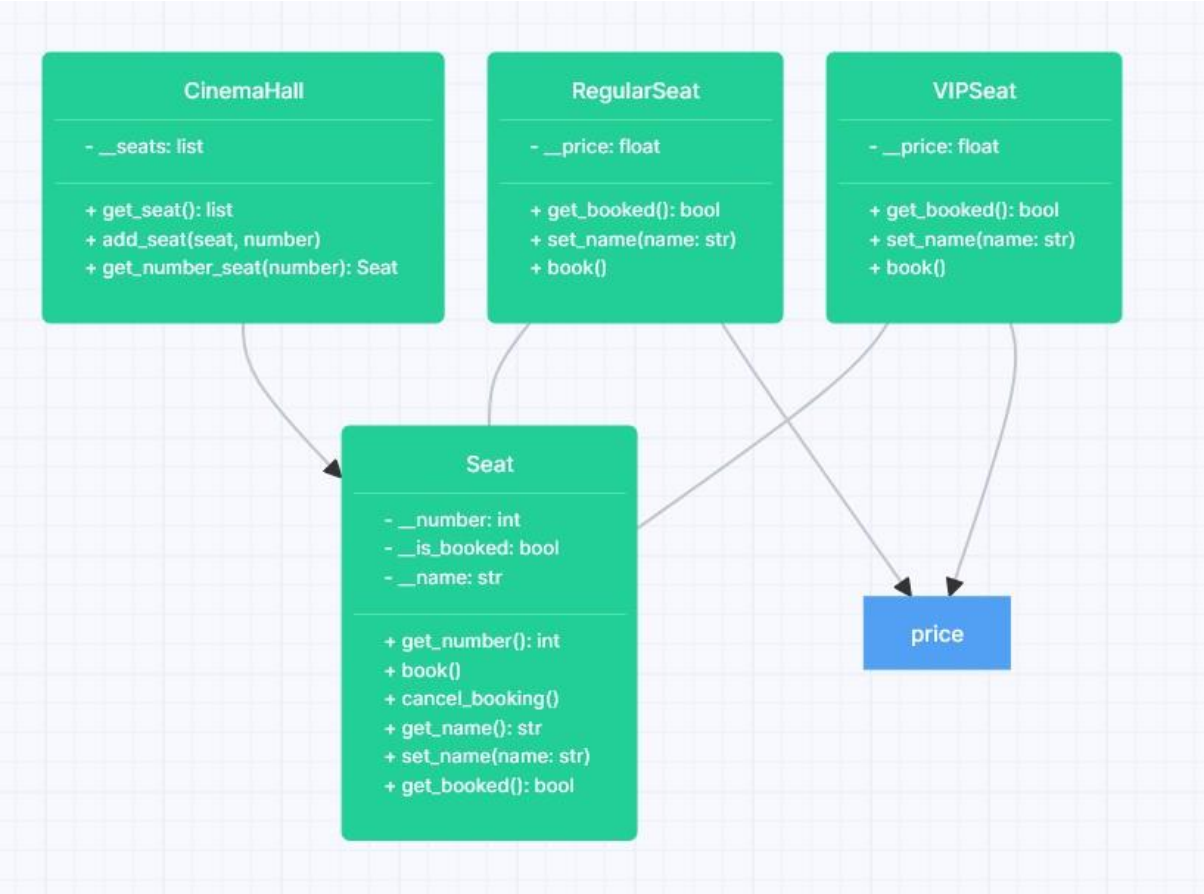
สองคลาสเหล่านี้เป็นคลาสที่สืบทอดมาจาก Seat โดยมีการโอเวอร์ไรด์เมธอด book() เพื่อเพิ่มพฤติกรรมเฉพาะของแต่ละประเภทของที่นั่ง และเมธอด set_name() เพื่อกำหนดชื่อผู้จอง ต่างจากคลาสหลัก โดยกำหนดราคาที่นั่งของแต่ละประเภทด้วย

สร้างคลาส CinemaHall

```
# class CinemaHall
class CinemaHall:
   #  method CinemaHall
    def __init__(self ,seat_all):
        self.__seats = [None] * seat_all

    def get_seat(self):
        return self.__seats

    def add_seat(self , seat , seat_number):
        self.__seats[seat_number] = seat

    def get_number_seat(self , number):
         return self.__seats[number]
```

เป็นคลาสที่ใช้เก็บข้อมูลเกี่ยวกับโรงภาพยนตร์ เช่น รายการที่นั่งทั้งหมดที่มีอยู่ในโรงภาพยนตร์ โดยมีเมธอดเพิ่มที่นั่ง add_seat() และเรียกดูข้อมูลของที่นั่งตามเลขที่นั่งที่กำหนด get_number_seat()

UML_diagram.md



โค้ดที่กำหนดมีแนวคิดเกี่ยวกับ Object-Oriented Programming (OOP) อย่างชัดเจน โดยใช้หลักการของ Encapsulation, Polymorphism, และ Inheritance ในการออกแบบและสร้างโค้ดของระบบการจัดการที่นั่งในโรงภาพยนต์ได้ดังนี้

Encapsulation (การซ่อนรายละเอียด)

- คลาส Seat มีการใช้งาน Encapsulation โดยใช้เมทอด getter/setter สำหรับการเข้าถึงและกำหนดค่าของตัวแปรเป็น private (__number, __is_booked, __name) เพื่อปกป้องข้อมูลและป้องกันการเข้าถึงโดยตรงจากภายนอกคลาส
- เมทอด book(), cancel_booking(), get_name(), set_name(name), get_booked() จะถูกเรียกใช้เพื่อทำงานกับข้อมูลในคลาส Seat โดยตรงผ่านเมทอดที่ถูกเข้ารหัสไว้

Polymorphism (การหลายรูปแบบ)

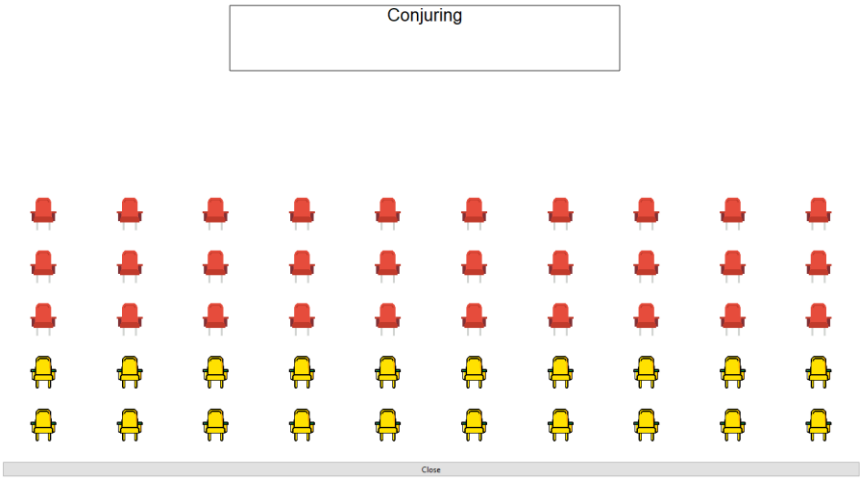
- โค้ดมีการใช้ Polymorphism ในเมทอด set_name(name) และ get_booked() ที่ถูกกำหนดเป็น abstract ใน Abstract Base Class Seat และถูกโอเวอร์ไรด์ (override) ในคลาส RegularSeat และ VIPSeat โดยใช้เมทอดเหล่านี้เพื่อกำหนดชื่อผู้จองและตรวจสอบสถานะการจองของที่นั่งตามลำดับ

Inheritance (การสืบทอด)

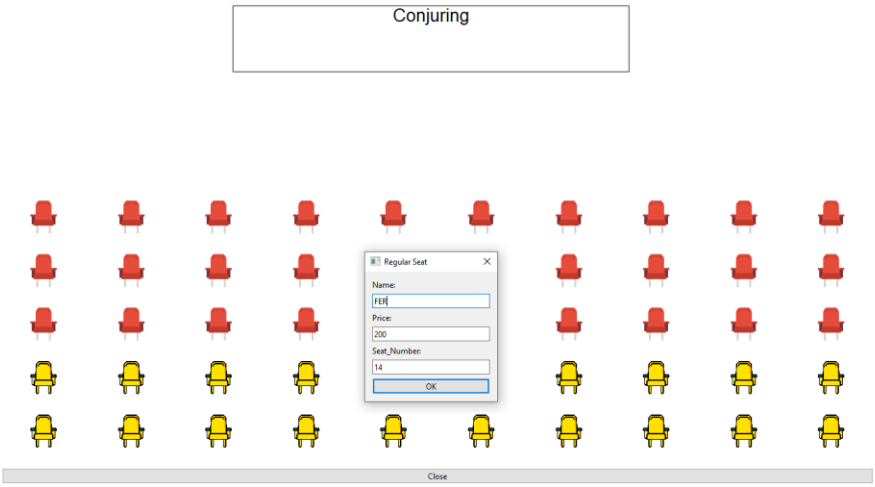
- คลาส RegularSeat และ VIPSeat ได้รับการสืบทอดทุกอย่างจากคลาส Seat ซึ่งทำให้สามารถใช้เมทอดและลักษณะของ Seat ได้โดยตรง และสามารถเพิ่มความสามารถเฉพาะของตัวเองได้ด้วยการเพิ่มเมทอดและลักษณะในคลาสของตนเอง

โดยทั้ง Encapsulation, Polymorphism, และ Inheritance เป็นหลักการสำคัญใน OOP ที่ช่วยให้โค้ดมีความยืดหยุ่นและสามารถนำไปใช้ซ้ำในส่วนต่าง ๆ ของโปรแกรมได้อย่างมีประสิทธิภาพและเป็นระเบียบ

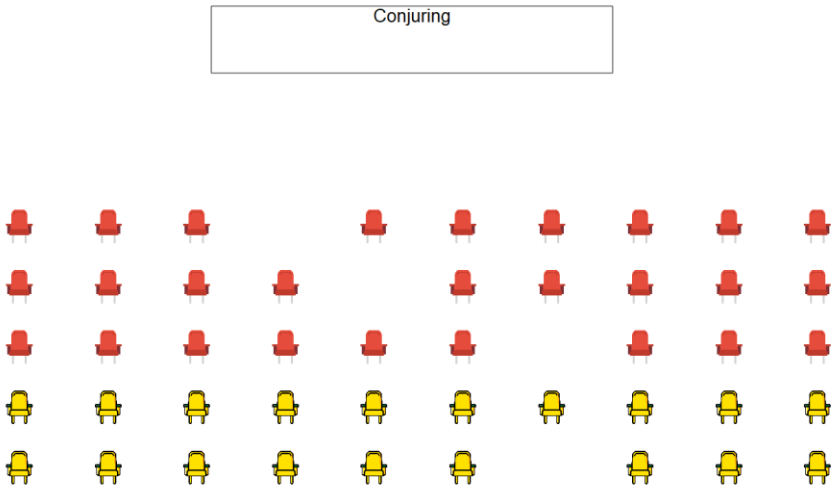
ผลการรัน



หน้าหลักของระบบการจองที่นั่งในโรงภาพยนตร์



ทำการคลิกที่นั่งที่ใดที่หนึ่งจากนั้นกรอกชื่อ แล้วกด OK



ที่นั่งที่ทำการจองแล้วจะหายไป