Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по программе «**Практическое занятие №8**»

Выполнила: Яковлева А.Н.

Группа: ПР-21

Преподаватель: Мирошниченко Г.В

2025

**Вариант 12**

**Задание:** Создать проект, в котором можно было бы вводить и получать информацию об определенных класса Кинотеатр.

**Входные и выходные данные основной программы**

n - количество внесенных фильмов, целое;

act - номер действия, целое;

move - номер фильма, целое.

**Описание класса Cinema:**

*Поля:*

name - название фильма, строковый;

session - время начала сеанса, строковый;

price - цена, целый беззнаковый;

hall - номер зала, целый беззнаковый;

spectators - количество мест для зрителей, целый беззнаковый.

*Методы:*

Filling() – запонение информации о фильме, процедура;

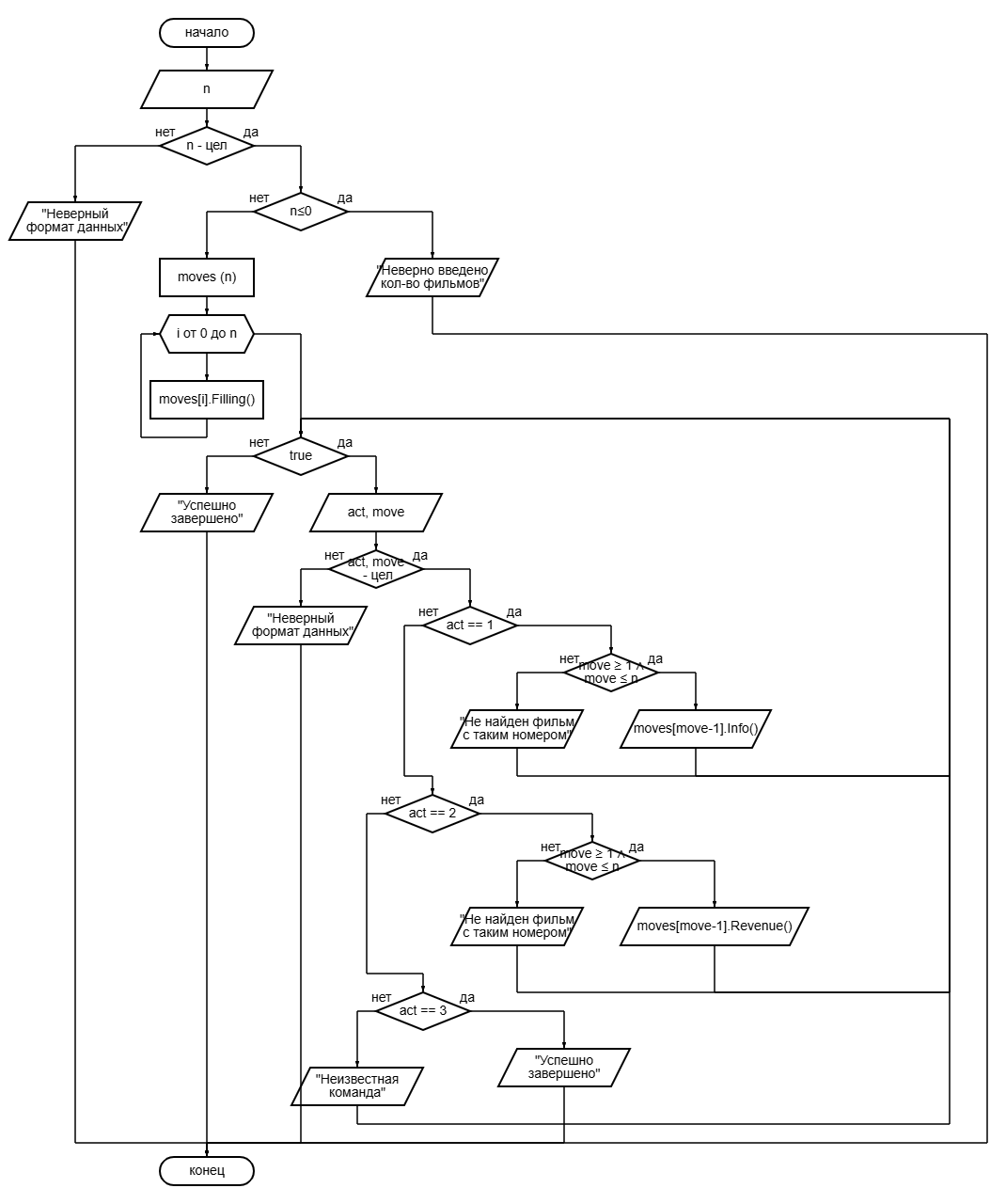
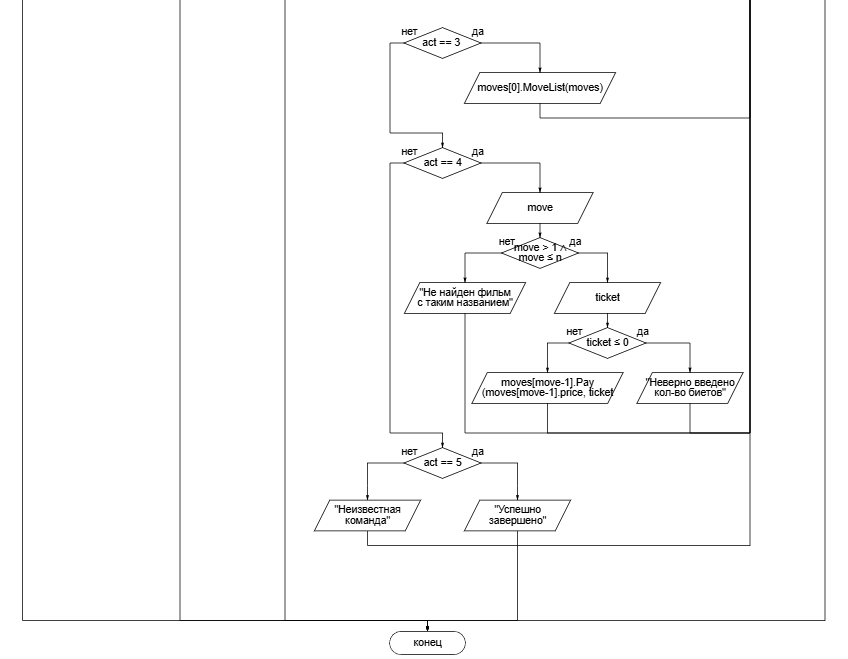
Info() – вывод информации о фильме, строка;

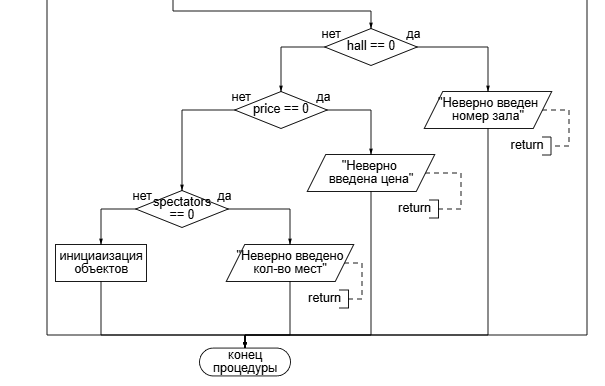
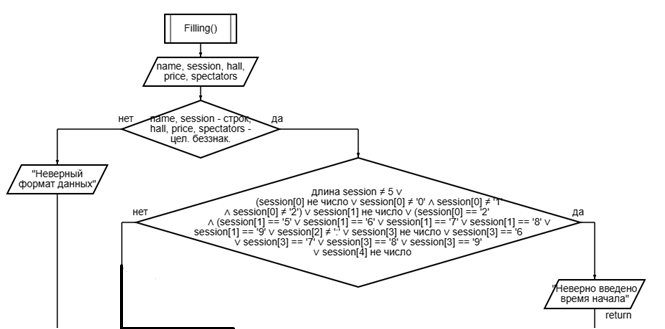
Revenue() – подсчет выручки с фильма, целый беззнаковый;

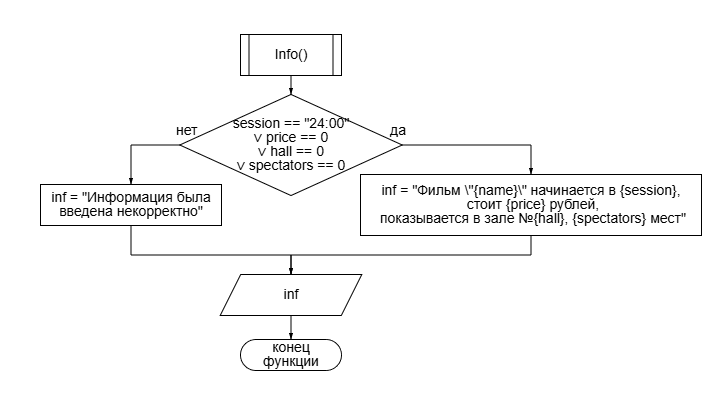
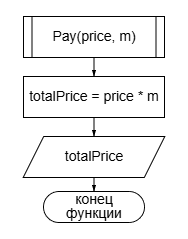
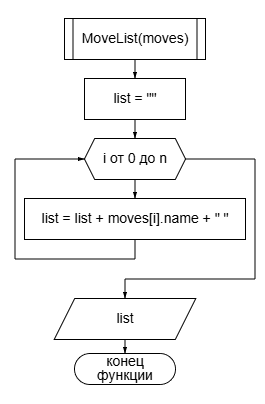
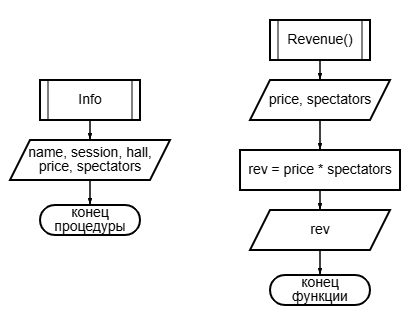
MoveList(массив объектов moves) – список названий фильмов, строка;

Pay(price, m) - подсчет стоимости для m билетов, целый беззнаковый.

**Блок-схема**

****





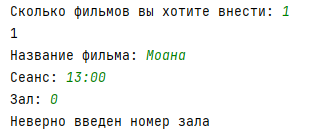
**Листинг программы**

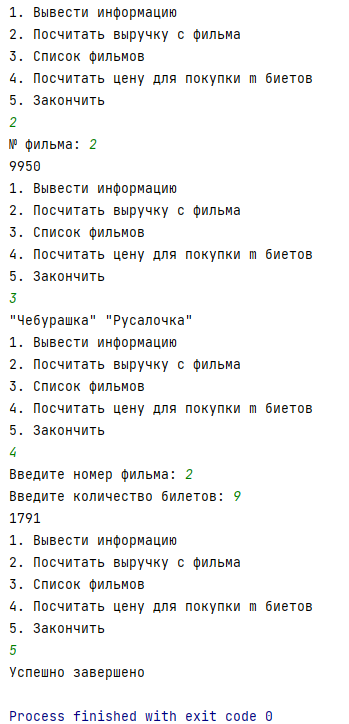
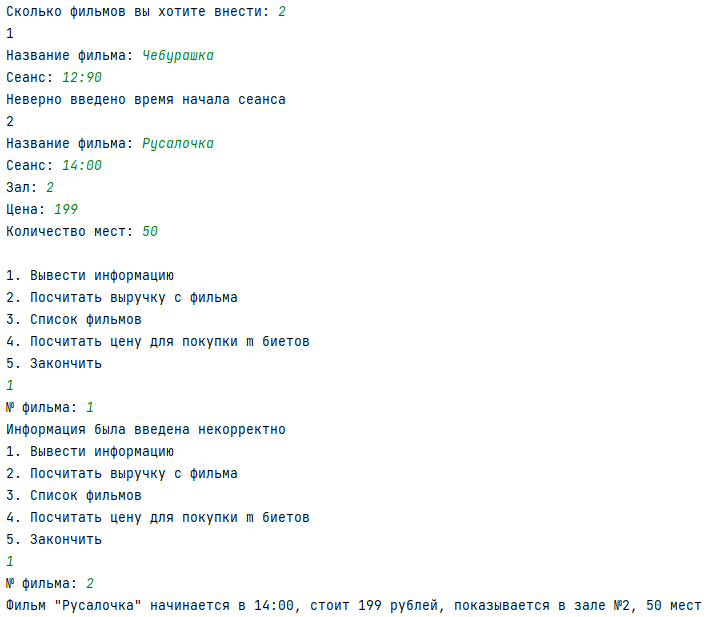
class Cinema constructor() {  
 var name: String = "name" *//название фильма* var session: String = "24:00" *//время начала сеанса* var price: UInt = 0u *//цена* var hall: UInt = 0u *//зал* var spectators: UInt = 0u *//количество мест для зрителей  
  
 //Ввести информацию о фильме* fun Filling(){  
 try {  
 *print*("Название фильма: ")  
 name = *readLine*()!!.toString()  
 *print*("Сеанс: ")  
 session = *readLine*()!!.toString()  
 if (session.length != 5 ||  
 (!session[0].*isDigit*() || session[0] != '0' && session[0] != '1' && session[0] != '2')  
 || !session[1].*isDigit*() || (session[0] == '2'  
 && (session[1] == '5' || session[1] == '6' || session[1] == '7' || session[1] == '8' || session[1] == '9'))  
 || session[2] != ':'  
 || !session[3].*isDigit*() || session[3] == '6' || session[3] == '7' || session[3] == '8' || session[3] == '9'  
 || !session[4].*isDigit*()  
 ) {  
 *println*("Неверно введено время начала сеанса")  
 return  
 }  
 *print*("Зал: ")  
 hall = *readLine*()!!.*toUInt*()  
 if (hall == 0u) {  
 *println*("Неверно введен номер зала")  
 return  
 }  
 *print*("Цена: ")  
 price = *readLine*()!!.*toUInt*()  
 if (price == 0u) {  
 *println*("Неверно введена цена")  
 return  
 }  
 *print*("Количество мест: ")  
 spectators = *readLine*()!!.*toUInt*()  
 if (spectators == 0u){  
 *println*("Неверно введено коичество мест для зрителей")  
 return  
 }  
 *println*()  
 }  
 catch (e: Exception) {  
 *println*("Неверный формат данных")  
 }  
 }  
  
 *//Вывести информацию о фильме* fun Info(): String{  
 var inf: String = ""  
 if (session == "24:00" || price == 0u || hall == 0u || spectators == 0u){  
 inf = "Информация была введена некорректно"  
 }  
 else {  
 inf = "Фильм \"$name\" начинается в $session, стоит $price рублей, показывается в зале №$hall, $spectators мест"  
 }  
 return inf  
 }  
  
 *//Подсчет выручки* fun Revenue() :UInt{  
 val rev = price \* spectators  
 return rev  
 }  
  
 *//Список названий фильмов* fun MoveList(moves: Array<Cinema>): String{  
 var list: String = ""  
 for (i in 0 *until* moves.size)  
 {  
 list += "\"${moves[i].name}\" "  
 }  
 return list  
 }  
  
 *//Подсчет стоимости для m билетов* fun Pay(price: UInt, m: UInt):UInt{  
 val totalPrice = price \* m  
 return totalPrice  
 }  
}

fun main() {  
 try {  
 *print*("Сколько фильмов вы хотите внести: ")  
 val n = *readLine*()!!.*toInt*()  
 when  
 {  
 n <= 0 -> *println*("Неверно введено количество фильмов")  
 else -> {  
 val moves = Array(n) **{** Cinema() **}** *//Массив объектов(фильмов)* for (i in 0 *until* n)  
 {  
 *println*(i+1)  
 moves[i].Filling()  
 }  
 while (true) {  
 *println*("1. Вывести информацию")  
 *println*("2. Посчитать выручку с фильма")  
 *println*("3. Список фильмов")  
 *println*("4. Посчитать цену для покупки m биетов")  
 *println*("5. Закончить")  
 val act = *readLine*()!!.*toInt*()  
 when (act){  
 1 -> {  
 *print*("№ фильма: ")  
 val move = *readLine*()!!.*toInt*()  
 when{  
 (move in 1..n) -> *println*(moves[move-1].Info())  
 else -> *println*("Не найдем фильм с таким номером")  
 }  
 }  
 2 -> {  
 *print*("№ фильма: ")  
 val move = *readLine*()!!.*toInt*()  
 when{  
 (move in 1..n) -> *println*(moves[move-1].Revenue())  
 else -> *println*("Не найдем фильм с таким номером")  
 }  
 }  
 3 -> {  
 *println*(moves[0].MoveList(moves))  
 }  
 4 -> {  
 *print*("Введите номер фильма: ")  
 val move = *readLine*()!!.*toInt*()  
 when {  
 (move in 1..n) ->{  
 *print*("Введите количество билетов: ")  
 val ticket = *readLine*()!!.*toUInt*()  
 when {  
 (ticket == 0u) -> *println*("Неверно введено количество билетов")  
 else -> *println*(moves[move-1].Pay(moves[move-1].price, ticket))  
 }  
 }  
 else -> *println*("Не найдем фильм с таким номером")  
 }  
 }  
 5 -> break  
 else -> *println*("Неизвестная команда")  
 }  
 }  
 *println*("Успешно завершено")  
 }  
 }  
 }  
 catch (e: Exception)  
 {  
 *println*("Неверный формат данных")  
 }  
}

**Тестовые ситуации**





**Вывод:** Я научилась работать с классами в Kotlin.