Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по программе «**Практическое занятие 9.2**»

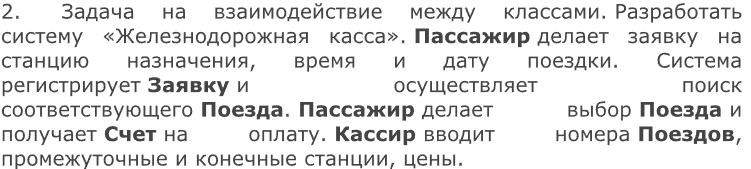
Выполнил: Сагутдинова Илона Ильнаровна

Группа: ПР-21

Преподаватель: Мирошниченко Г.В

2023

**Задание 1:**

****

**Входные данные:**

passengerName – имя пассажира, строка, string.

passengerAge – возраст пассажира, целое число, int.

trainNumber – номер поезда, целое число, int.

station – станция назначения, строка, string.

Time – время отправки поезда, число, int.

data – дата отправки поезда, число, int.

price – цена билета, дробное число, double.

**Выходные данные:**

Ticket.Info() – функция из класса, информация о билете.

**Блок-схема:**

**Листинг программы:**

import Railway.\*  
import Railway.Passenger as NewPassenger  
import Railway.Child as NewChild  
import Railway.Cashier as NewCashier  
import Railway.Ticket as NewTicket  
import Railway.Train as NewTrain  
fun main (){  
 try {  
 *println*("Введите имя пассажира:")  
 val passengerName = *readLine*()!!.toString()  
 *println*("Введите возраст пассажира:")  
 val passengerAge = *readLine*()!!.*toInt*()  
  
 // Создание объекта пассажира  
 val passenger = if (passengerAge!! < 18) {  
 NewChild(passengerName, passengerAge)  
 } else {  
 NewPassenger(passengerName, passengerAge)  
 }  
  
 // Проверка возраста  
 if (!passenger.ValidAge()) {  
 return  
 }  
 //Вывод информации о билете  
 passenger.Info()  
  
 // Ввод данных билета  
 *println*("Введите номер поезда:")  
 val trainNumber = *readLine*()!!.*toInt*()  
 *println*("Введите станцию назначения:")  
 val station = *readLine*() ?: ""  
 *println*("Введите время отбытия:")  
 val time = *readLine*()!!.*toInt*()  
 *println*("Введите дату отбытия:")  
 val data = *readLine*()!!.*toInt*()  
 *println*("Введите базовую цену билета:")  
 val price = *readLine*()!!.*toDouble*()  
  
 //Создание объекта поезд  
 val train = NewTrain(trainNumber, station, price)  
 //Проверка цены  
 if (!train.ValidPrice(price))  
 {  
 return  
 }  
 //Проверка номера поезда  
 if (!train.ValidNumer(trainNumber))  
 {  
 return  
 }  
 // Создание объекта билета  
 val ticket = NewTicket(passengerName, passengerAge, trainNumber, station, time, data, price)  
  
 //Рассчет скидки для ребенка  
 var discount = 0.0  
 if (passenger is NewChild) {  
 discount = passenger.СalculateDiscount()  
 *println*("Скидка для ребенка: ${discount \* 100}%")  
 }  
  
 // Расчет итоговой цены билета  
 val finalPrice = ticket.FinalPrice(discount)  
 ticket.price = finalPrice  
  
 // Ввод данных кассира  
 *println*("Введите имя кассира:")  
 val cashierName = *readLine*()!!.toString()  
  
 //Создание объекта кассира  
 val cashier = NewCashier(cashierName)  
  
 // Продажа билета  
 cashier.SellTicket(passenger, ticket)  
  
 // Изменение данных билета  
 *println*("Хотите изменить станцию назначения? (да/нет)")  
 val answer2 = *readLine*()!!.toString()  
 if (answer2 == "да") {  
 *println*("Введите новую станцию:")  
 val newStation = *readLine*() ?: ""  
 *println*("Введите новое время отбытия:")  
 val newTime = *readLine*()?.*toInt*()  
 *println*("Введите новую дату отбытия:")  
 val newData = *readLine*()?.*toInt*()  
 cashier.Сhange(ticket, newStation, newTime, newData)  
 }  
 else  
 {  
 *println*("Изменений нет")  
 }  
  
 // Вывод информации о билете  
 ticket.Info()  
  
 //Возврат билета  
 *println*("Хотите оформить возврат? (да/нет)")  
 val answer = *readLine*()!!.toString()  
 if (answer == "да")  
 {  
 cashier.ReturnTicket(ticket)  
 return  
 }  
 else  
 {  
 *println*("Возврат не оформен")  
 }  
 }  
 catch (e:Exception)  
 {  
 *println*("Неверный ввод")  
 }  
}

package Railway  
  
open class Train(  
 var numer: Int,  
 var endStation: String,  
 var price: Double  
) {  
 fun ValidPrice(price: Double?):Boolean{  
 if (price!! <0.0)  
 {  
 *println*("Цена не может быть меньше или равняться 0")  
 return false  
 }  
 return true  
 }  
 fun ValidNumer(numer: Int?):Boolean{  
 if (numer!! <0)  
 {  
 *println*("Номер поезда не может быть меньше или равняться 0")  
 return false  
 }  
 return true  
 }  
}

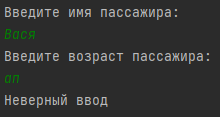
package Railway  
  
class Cashier(var name: String) {  
  
 //Функция для продажи билета  
 fun SellTicket(passenger: Passenger, ticket: Ticket) {  
 *println*("Кассир $name продал билет пассажиру ${passenger.names}")  
 ticket.Info()  
 }  
  
 //Функция для возврата билета  
 fun ReturnTicket(ticket: Ticket) {  
 *println*("Кассир $name оформил возврат билета на поезд №${ticket.numTrain}")  
 }  
  
 //Функция для изменения билета  
 fun Сhange(ticket: Ticket, newStation: String, newTime: Int?, newData: Int?) {  
 ticket.СhangeStation(newStation)  
 ticket.time = newTime  
 ticket.data = newData  
 *println*("Данные билета изменены.")  
 }  
}

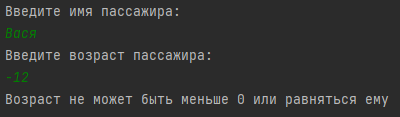
package Railway  
open class Passenger (var names:String,var age : Int?) {  
 //Функция для проверки возраста  
 fun ValidAge():Boolean{  
 if (age!!<=0)  
 {  
 *println*("Возраст не может быть меньше 0 или равняться ему")  
 return false  
 }  
 return true  
 }  
  
 //Функция для выводы информации пассажира  
 open fun Info(){  
 *println*("Имя: $names")  
 *println*("Возраст: $age")  
 }  
 fun ChangeName(newName: String) {  
 names = newName  
 *println*("Имя изменено на: $names")  
 }  
}

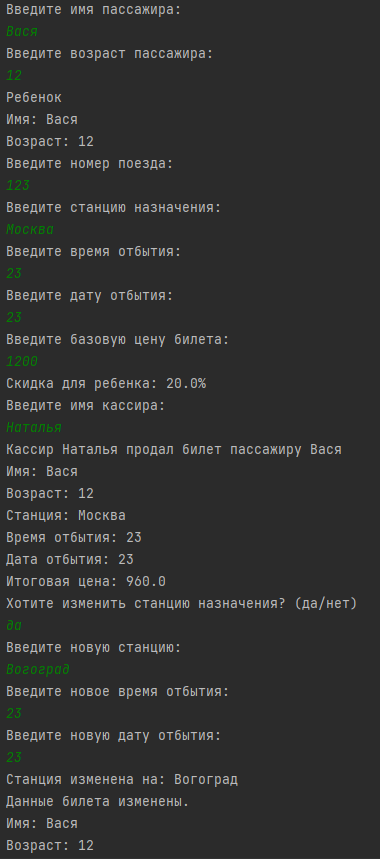
package Railway  
class Ticket(  
 var name:String,  
 age:Int,  
 var numTrain: Int?,  
 var station:String,  
 var time:Int?,  
 var data:Int?,  
 var price:Double?,):Passenger(name,age) {  
  
 //Функция для вывода информации о билете  
 override fun Info(){  
 *println*("Имя: $names")  
 *println*("Возраст: $age")  
 *println*("Станция: $station")  
 *println*("Время отбытия: $time")  
 *println*("Дата отбытия: $data")  
 *println*("Итоговая цена: $price")  
 }  
 //Рассчет цены билета  
 fun FinalPrice(discount: Double): Double {  
 return price!! \* (1 - discount)  
 }  
  
 //Вывод об измененной станции  
 fun СhangeStation(newStation: String) {  
 station = newStation  
 *println*("Станция изменена на: $station")  
 }  
}

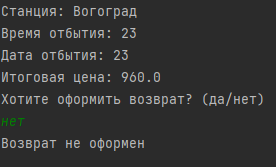
package Railway  
class Child(name:String,  
 age : Int):Passenger(name,age) {  
//Функция вывода информации о ребенке  
 override fun Info() {  
 *println*("Ребенок")  
 super.Info()  
 }  
//Рассчет скидки для ребенка  
 fun СalculateDiscount(): Double {  
 return if (age!! < 12) 0.5 else 0.2  
 }  
}

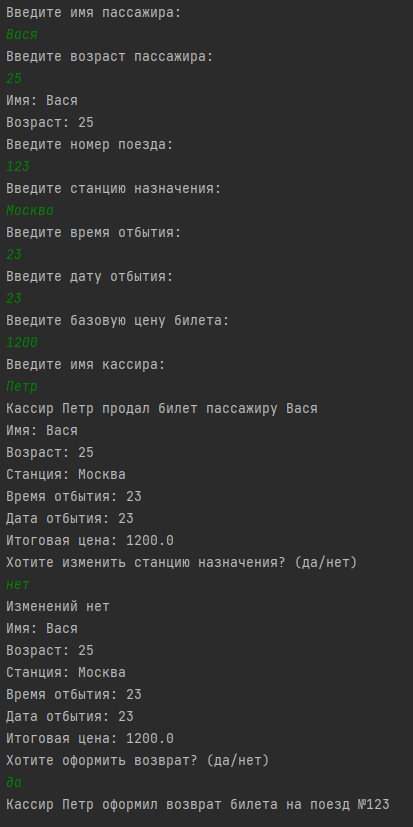
**Тестовые ситуации:**

****

****

****

****

****