State Cinema Ticket Price Controller Testing

Mark Christison

Nelson Marlborough Institute of Technology, New Zealand

Contents

Introduction

This document contains the Equivalence partition and boundary tables as well as test schedules for the ticket price controller of the state cinema ticket booking system.

Tests were carried out using NUnit, a unit-testing framework for .NET languages. Tests were written using JetBrains Rider.

Below is the current Pricing from the state cinema website. I have excluded some of the prices such as online booking fees as these are not represented in the test schedule or the price controller.

PRICING AND POLICY

Ticket Prices

	2D
Adult before 5pm	$14.50
Adult after 5pm	$17.50
Adult Tuesday (all day)	$13.00
Child (under 16) (all day)	$12.00
Senior (65+) (all day)	$12.50
Student (Current ID required)	$14.00
Family Pass (2 Adults/2 Children) or (1 Adult/3 Children)*Family Pass (2 Adults/2 Children) or (1 Adult/3 Children)	$46.00
Red Carpet Special (Film & Drink or Large Popcorn)	$22.00
Kids and Carers (1st Wed every Month)	$14.50

*Family Pass tickets must be sold to the same film.

Adult Before 5

Inputs int prQuantity, string prPerson, string prDay, decimal prTime

Outputs – dec

Equivalence partitioning and boundaries				
Status	**Quantity**	**Person**	**Day**	**Time**
Acceptable	>0	Adult	Monday Wednesday Thursday Friday Saturday Sunday	>5 <12
Unacceptable	>=0	Student Family Senior Child	Tuesday	>=0 <5

Test Schedule						
Use Cases	**Quantity**	**Person**	**Day**	**Time**	**Expected**	**Result**
One Adult on Monday at 1.00	1	Adult	Monday	1.00	14.50	14.50
One Adult on Saturday at 2.00	1	Adult	Monday	2.00	14.50	14.50
Three Adults on Thursday at 1.30	3	Adult	Thursday	1.30	43.50	43.50
1 Adult on Tuesday at 2.45	1	Adult	Tuesday	2.45	-1	-1
1 Child on Wednesday at 2.45	1	Child	Wednesday	2.45	-1	-1
1 Student on Wednesday at 3.00	1	Student	Wednesday	3.00	-1	-1
1 Senior on Wednesday at 3.00	1	Senior	Wednesday	3.00	-1	-1

1 adult on Wednesday at 10am	1	Adult	Wednesday	10.00	14.5	-1
1 family on Wednesday at 3.00	1	Family	Wednesday	3.00	-1	-1

Adult After 5

Inputs – int prQuantity, string prPerson, string prDay, decimal prTime

Outputs - dec

Equivalence partitioning and boundaries				
Status	**Quantity**	**Person**	**Day**	**Time**
Acceptable	>0	Adult	Monday Wednesday Thursday Friday Saturday Sunday	>5 <12
Unacceptable	>=0	Student Family Senior C hild	Tuesday	<5 >12

Test Schedule						
Use Cases	**Quantity**	**Person**	**Day**	**Time**	**Expected**	**Result**
1 Adult Wednesday 6pm	1	Adult	Wednesday	6.00	17.5	17.5
2 Adults Thursday 7.40pm	2	Adult	Thursday	7.40	35	35
1 adult Saturday 8pm	1	Adult	Saturday	8.00	17.5	17.5
1 Student Wednesday 8pm	1	Student	Wednesday	8.00	-1	-1
1 Family Wednesday 8pm	1	Family	Wednesday	8.00	-1	-1
1 Senior Wednesday 8pm	1	Senior	Wednesday	8.00	-1	-1
1 ADULT Wednesday 8pm	1	ADULT	Wednesday	8.00	17.5	17.5

Adult Tuesday

Inputs – int prQuantity, string prPerson, string prDay

Outputs – dec

Equivalence partitioning and boundaries			
Status	**Quantity**	**Person**	**Day**
Acceptable	> 0	Adult	Tuesday
Unacceptable	<= 0	Student Family Senior Child	Monday Wednesday Thursday Friday Saturday Sunday

Test Cases					
Use Cases	**Quantity**	**Person**	**Day**	**Expected**	**Result**
1 Adult Tuesday	1	Adult	Tuesday	13	13
4 Adults Tuesday	1	Adult	Tuesday	52	52
1 Child Tuesday	1	Child	Tuesday	-1	-1
1 Student Tuesday	1	Student	Tuesday	-1	-1
1 Family Tuesday	1	Family	Tuesday	-1	-1
1 Adult Wednesday 6pm	1	Adult	Wednesday	-1	-1

Child Under 16

Inputs – int prQuantity, string prPerson

Outputs – dec

Equivalence partitioning and boundaries		
Status	**Quantity**	**Person**
Acceptable	> 0	Child
Unacceptable	<= 0	Senior Adult Student Family

Test Cases				
Use Cases	**Quantity**	**Person**	**Expected**	**Result**
1 Child under 16	1	Child	12	12
3 Children under 16	3	Child	36	36
1 Student	1	Student	-1	-1
1 Family	1	Family	-1	-1
1 Adult	1	Adult	-1	-1

Senior

Inputs – int prQuantity, string prPerson

Outputs – dec

Equivalence partitioning and boundaries		
Status	**Quantity**	**Person**
Acceptable	> 0	Senior
Unacceptable	<=0	Adult Child Student Family

Test Schedule				
Use Cases	**Quantity**	**Person**	**Expected**	**Result**
1 Senior tickets	1	Senior	12.50	12.50
3 Seniors tickets	3	Senior	37.50	37.50
1 Student tickets	1	Student	-1	-1
1 Family tickets	1	Family	-1	-1
1 Adult tickets	1	Adult	-1	-1
1000 Senior tickets	1000	Senior	12500	12500
-1 Senior tickets	-1	Senior	-1	-1

Student

Input - int prQuantity, string prPerson

Output - dec

Equivalence partitioning and boundaries		
Status	**Quantity**	**Person**
Acceptable	> 0	Student
Unacceptable	<=0	Adult Child Student Family

Test Schedule				
Use Cases	**Quantity**	**Person**	**Expected**	**Result**
1 Student ticket	1	Student	14	14
7 Students tickets	7	Student	98	98
-10 Students tickets	-10	Student	-1	-1
1 Family ticket	1	Family	-1	-1
1 Adult ticket	1	Adult	-1	-1
1 Senior ticket	1	Senior	-1	-1

Family Pass

Input - int prQuantityTicket, int prQuantityAdult, int prQuantityChild

Output - dec

Equivalence partitioning and boundaries			
Status	**Quantity**	**Quantity Adult**	**Quantity Child**
Acceptable	1	1 2	2 3
Unacceptable	!= 1	<= 0 > 2	<= 0 > 3

Test Schedule					
Use Cases	**Quantity**	**Quantity Adult**	**Quantity Child**	**Expected**	**Result**
2 adults and 2 children 1 family pass	1	2	2	46.00	46.00
1 family pass 1 adult 3 children	1	1	3	46.00	46.00
1 family pass 3 adults 1 child	1	3	1	-1	-1
1 family pass 0 adults 4 children	1	0	4	-1	-1
1 Family Pass 4 Adults 0 children	1	4	0	-1	-1
10 family passes 20 Adults 20 Children	10	20	20	-1	-1
-1 Family Pass 2 Adults 2 Children	-1	2	2	-1	-1

Chick Flick Thursday

Input - int prQuantity, string prPerson, string prDay

Output - dec

Equivalence partitioning and boundaries			
Status	**Quantity**	**Person**	**Day**
Acceptable	> 0	Adult	Thursday
Unacceptable	< 1	Child Senior Student Family	Monday Tuesday Wednesday Friday Saturday Sunday

Test Schedule					
Use Cases	**Quantity**	**Person**	**Day**	**Expected**	**Result**
1 Adult on Thursday	1	Adult	Thursday	21.50M	21.50M
3 Adults on Thursday	3	Adult	Thursday	64.50M	64.50M
1 Adult on Wednesday	1	Adult	Wednesday	-1	-1
1 Child on Thursday	1	Child	Thursday	-1	-1
1 Student on Thursday	1	Student	Thursday	-1	-1
-2 Adults on Thursday	-2	Adult	Thursday	-1	-1
1 @du17 on Thursday	1	@du17	Thursday	-1	-1

Kids Careers

Input - int prQuantity, string prDay, bool prHoliday

Output - dec

Equivalence partitioning and boundaries			
Status	**Quantity**	**Day**	**Holiday**
Acceptable	> 0	Wednesday	True
Unacceptable	<= 0	Monday Tuesday Thursday Friday Saturday Sunday	False

Test Schedule					
Use Cases	**Quantity**	**Day**	**Holiday**	**Expected**	**Result**
1 Career Wednesday Holiday	1	Wednesday	True	12	12
4 Careers Wednesday Holiday	4	Wednesday	True	48	48
1 Career Tuesday Holiday	1	Tuesday	True	-1	-1
-1 Career Wednesday Holiday	-1	Wednesday	True	-1	-1
1 Career Wednesday Not Holiday	1	Wednesday	False	-1	-1
1 Career 'RandomString' Holiday	1	'RandomString'	True	-1	-1

Conclusion

I believe that the cinema functions could better reflect the current prices that are listed on the state cinema website. Moreover, adding in all the extra functionality of 3d movies, or clarifying the price controller to a further degree would be recommended.

Additionally, some parameters such as the time input, need to be slightly modified. Currently the functions work on a 12-hour clock. This could be improved such that the ticket price controller accepted 24-hour input and clarifying the time of day that the movie is being played.

It is arguable that the family pass should be acceptable to be order in bulk. However, in these type of cases I would assume that it would be larger than normal events and as such some extra coordination amongst the people who want the tickets and the movie theatre would need to occur.