

**SDV502 Application Testing**

**Assessment 1**

**Kenneth-John Williams-Stockdale**

NMIT Student ID: 13477293

National Student Number: 133006701

Contents

[Introduction 3](#_Toc81560290)

[Public Function Adult\_Before\_5 3](#_Toc81560291)

[Public Function Adult\_After\_5 4](#_Toc81560292)

[Public Function Adult\_Tuesday 5](#_Toc81560293)

[Public Function Child\_Under\_16 6](#_Toc81560294)

[Public Function Senior 7](#_Toc81560295)

[Public Function Student 8](#_Toc81560296)

[Public Function Family\_Pass 9](#_Toc81560297)

[Public Function Chick\_Flick\_Thursday 10](#_Toc81560298)

[Public Function Kids\_Careers 11](#_Toc81560299)

[Summary 12](#_Toc81560300)

# Introduction

The case study for this assessment concerns the calculation of the cinema ticket prices charged by local State Cinemas in Nelson. These test cases test multiple cases of different functions ranging from days, time of days, quantity, and would be classes of people buying tickets. The test cases along with estimated results test if the predicted outcome is correct/ passes the test or fails if the test user case and expected results are different.

# Public Function Adult\_Before\_5

Input – int quantity, string person, string day, decimal time   
Output – decimal total-price

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Equivalence Partitioning & Boundaries** | | | | |
| **Status** | **quantitiy** | **person** | **day** | **time** |
| **Acceptable** | >0 | adult | Mon,  Wed,  Thu,  Fri,  Sat,  sun | <=1700 |
| **Unacceptable** | <=0 | Student,  Family,  Senior,  child | tue | >1700 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Test schedule** | | | | | | |
| **Use Case** | | **quantity** | **person** | **day** | **time** | **Expected** | **Result** |
| 1. One adult on Monday | | 1 | adult | monday | 4 | 14.50 | Pass |
| 2. Two adults on Monday | | 2 | adult | monday | 4 | 29.00 | Pass |
| 3. Zero adults on Monday | | 0 | senior | monday | 4 | -1 | Pass |
| 4. Four adults on Monday | | 4 | adult | monday | 6 | -1 | Pass |
| 5. Four adults on Monday | | 4 | adult | monday | 4 | 58.00 | Pass |

# Public Function Adult\_After\_5

Input – int quantity, string person, string day, decimal time   
Output – decimal total-price

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Equivalence Partitioning & Boundaries** | | | | |
| **Status** | **quantitiy** | **person** | **day** | **time** |
| **Acceptable** | >0 | adult | Mon,  Wed,  Thu,  Fri,  Sat,  sun | <=1700 |
| **Unacceptable** | <=0 | Student,  Family,  Senior,  child | tue | >1700 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Test schedule** | | | | | | |
| **Use Case** | | **quantity** | **person** | **day** | **time** | **Expected** | **Result** |
| 1. One adult on Tuesday | | 1 | adult | tuesday | 4 | -1 | Pass |
| 2. Two adults on Sunday | | 2 | adult | sunday | 4 | -1 | Pass |
| 3. One adults on Monday | | 1 | adult | monday | 6 | 17.50 | Pass |
| 4. Four adults on Monday | | 4 | adult | monday | 5 | 70.00 | Pass |
| 5. Three adults on Wedensday | | 3 | adult | wednesday | 7 | 52.50 | Pass |

# Public Function Adult\_Tuesday

Input – int quantity, string person, string day  
Output – decimal total-price

|  |  |  |  |
| --- | --- | --- | --- |
| **Equivalence Partitioning & Boundaries** | | | |
| **Status** | **quantitiy** | **person** | **day** |
| **Acceptable** | >0 | adult | Tue |
| **Unacceptable** | <=0 | Student,  Family,  Senior,  child | Mon,  Wed,  Thu,  Fri,  Sat,  sun |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Test schedule** | | | | | |
| **Use Case** | | **quantity** | **person** | **day** | **Expected** | **Result** |
| 1. One adult on Monday | | 1 | adult | monday | -1 | Pass |
| 2. One adult on Tuesday | | 1 | adult | tuesday | 13.00 | Pass |
| 3. One senior on Tuesday | | 1 | senior | tuesday | -1 | Pass |
| 4. Four adults on Tuesday | | 4 | adult | tuesday | 52.00 | Pass |
| 5. Six adults on Tuesday | | 6 | adult | tuesday | 78.00 | Pass |

# Public Function Child\_Under\_16

Input – int quantity, string person  
Output – decimal total-price

|  |  |  |
| --- | --- | --- |
| **Equivalence Partitioning & Boundaries** | | |
| **Status** | **quantitiy** | **person** |
| **Acceptable** | >0 | Child |
| **Unacceptable** | <=0 | Student,  Family,  Senior,  Adult |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Test schedule** | | | | |
| **Use Case** | | **quantity** | **person** | **Expected** | **Result** |
| 1. One child under 16 | | 1 | child | 12.00 | Pass |
| 2. Four children under 16 | | 4 | child | 48.00 | Pass |
| 3. One adult | | 1 | adult | -1 | Pass |
| 4. Four students | | 4 | student | -1 | Pass |
| 5. Six children under 16 | | 6 | child | 72.00 | Pass |

# Public Function Senior

Input – int quantity, string person  
Output – decimal total-price

|  |  |  |
| --- | --- | --- |
| **Equivalence Partitioning & Boundaries** | | |
| **Status** | **quantitiy** | **person** |
| **Acceptable** | >0 | Senior |
| **Unacceptable** | <=0 | Student,  Family,  Adult,  child |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Test schedule** | | | | | |
| **Use Case** | | **quantity** | **person** | **Expected** | **Result** |
| 1. One child | | 2 | child | -1 | Pass |
| 2. Two seniors | | 2 | senior | 25.00 | Pass |
| 3. One adult | | 1 | adult | -1 | Pass |
| 4. Four seniors | | 4 | senior | 50.00 | Pass |
| 5. Two students | | 2 | student | -1 | Pass |

# Public Function Student

Input – int quantity, string person  
Output – decimal total-price

|  |  |  |
| --- | --- | --- |
| **Equivalence Partitioning & Boundaries** | | |
| **Status** | **quantitiy** | **person** |
| **Acceptable** | >0 | Student |
| **Unacceptable** | <=0 | Adult,  Family,  Senior,  child |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Test schedule** | | | | | |
| **Use Case** | | **quantity** | **person** | **Expected** | **Result** |
| 1. One student | | 1 | student | 14.00 | Pass |
| 2. Two seniors | | 2 | senior | -1 | Pass |
| 3. Three adults | | 3 | adult | -1 | Pass |
| 4. Four students | | 4 | student | 56.00 | Pass |
| 5. Two students | | 2 | student | 28.00 | Pass |

# Public Function Family\_Pass

Input – int quantity\_adult, int quantity\_child   
Output – decimal total-price

|  |  |  |  |
| --- | --- | --- | --- |
| **Equivalence Partitioning & Boundaries** | | | |
| **Status** | **quantity\_ticket** | **quantitiy\_adult** | **quantitiy\_child** |
| **Acceptable** | >0 | >0 | >0 |
| **Unacceptable** | <=0 | <=0 | <=0 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Test schedule** | | | | | |
| **Use Case** | | **quantity\_ticket** | **quantity\_adult** | **quanitity\_child** | **Expected** | **Result** |
| 1. One pass, two adults and two children | | 1 | 2 | 2 | 46.00 | Pass |
| 2. One pass, one adult and three children | | 1 | 1 | 3 | 46.00 | Pass |
| 3. Two passes, two adults and six childre | | 2 | 2 | 6 | 92.00 | Fail |
| 4. One pass, three adults and one child | | 1 | 3 | 1 | -1 | Pass |
| 5. Two passes, four adults and 4 children | | 2 | 4 | 4 | 92.00 | Fail |

***Note****: Use cases 3 and 5 in my opinion should be technically correct in quantity of tickets in theory. Using real world application or situations that may arise such as being able to process two families at the same time. It allows the families to have an option to be able pay all together at the same time if required and for ease of use.*

*In this case the use case returns a failure result due to the way the code is written and processing of family quantity.*

# Public Function Chick\_Flick\_Thursday

Input – int quantity, string person,  
Output – decimal total-price

|  |  |  |  |
| --- | --- | --- | --- |
| **Equivalence Partitioning & Boundaries** | | | |
| **Status** | **Quantitiy** | **Person** | **Day** |
| **Acceptable** | >0 | adult | Thu |
| **Unacceptable** | <=0 | Student,  Family,  Senior,  Child | Mon,  Tue,  Wed,  Fri,  Sat,  Sun |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Test schedule** | | | | | |
| **Use Case** | | **quantity** | **person** | **day** | **Expected** | **Result** |
| 1. One adult on Thursday | | 1 | adult | thursday | 21.50 | Pass |
| 2. One student on Thursday | | 1 | student | thursday | -1 | Pass |
| 3. Two adults on Monday | | 2 | adult | monday | -1 | Pass |
| 4. Two adults on Thursday | | 2 | adult | thursday | 43.00 | Pass |
| 5. Four adults on Thursday | | 4 | adult | thursday | 86.00 | Pass |

# Public Function Kids\_Careers

Input – int quantity, string day, boolean holiday  
Output – decimal total-price

|  |  |  |  |
| --- | --- | --- | --- |
| **Equivalence Partitioning & Boundaries** | | | |
| **Status** | **Quantitiy** | **Day** | **Holiday** |
| **Acceptable** | >0 | Mon,  Tue,  Thu,  Fri,  Sat,  Sun | false |
| **Unacceptable** | <=0 | Wed | true |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Test schedule** | | | | | | |
| **Use Case** | | **quantity** | **day** | **holiday** | **Expected** | **Result** |
| 1. One admission on Thursday, non public holiday | | 1 | thursday | false | -1 | Pass |
| 2. One admission on Wednesday, non public holiday | | 1 | wednesday | false | 12.00 | Pass |
| 3. Two admissions on Wednesday, public holiday | | 2 | wednesday | true | -1 | Pass |
| 4. Two admissions on Wednesday, non public holiday | | 2 | wednesday | false | 24.00 | Pass |
| 5. Five admissions on Wednesday, non public holiday | | 5 | wednesday | false | 60.00 | Pass |

# Summary

These test cases show the logic behind the functions being used by State Cinemas Nelson to process transactions for admission/tickets for various classes of customers. Per function tested there are five test cases tested against that test function for a total of 45 test cases overall between 9 test functions. The test cases have been designed and tested with variations of test schedules to show basic different real-world scenarios of purchases from customers that may happen. With these test cases it shows that the functionality and logic of state cinema nelson’s ticket purchasing methods are successful.

However, there is one function for the test cases that returns a failure result in a test schedule that I tested. The function being tested was the family pass function. The way the code was written, and logic processed it only allows for a certain size family to be processed one at a time. With a change in this code, it could allow multiple sized families to be processed at once. This could be helpful for one simple payment among many people allowing faster admission rates and less cue times at busy times of the day.