***Achievement Standard 3.7*** *Version 1 AS91906 Credits: 6*

*Use complex programming techniques to develop a computer program*

**Marking guide**

|  |  |  |  |
| --- | --- | --- | --- |
| Program works as per specifications: |  |  |  |
| * It must allow the user to select the theatre |  |  |  |
| * It must allow the user to select the movie(s) |  |  |  |
| * It must ask for the quantity of tickets to be sold |  |  |  |
| * It must display an alert and block sales that exceed availability |  |  |  |
| * It must calculate the total cost of the order |  |  |  |
| * It must display the sale details |  |  |  |
| * It must enable the user to confirm or cancel the sale |  |  |  |
| The program must be set out clearly and documented with comments and docstrings |  |  |  |
| The program must use variables of at least two different data types |  |  |  |
| Program includes variables and lists |  |  |  |
| Program must contain input and have output |  |  |  |
| Have conditional statements such as IF |  |  |  |
| Have at least one loop such as WHILE or FOR |  |  |  |
| It must have user-defined functions |  |  |  |
| Uses 2 complex skills:   * Uses classes and objects * Includes a working GUI using Tkinter * Reads to/from a file |  |  |  |
| Video capture of program working is submitted |  |  |  |
| The program has been tested for expected cases |  |  |  |
| Evidence is provided that the program has been developed iteratively |  |  |  |
| **Teacher guidance required / worked independently throughout** |  |  |  |
| Enables the user to add movies |  |  |  |
| Enables the user to delete movies |  |  |  |
| Have followed the style guide |  |  |  |
| Variables, function, list and class names accurately reflect their function and behaviour |  |  |  |
| Parameters are passed into functions |  |  |  |
| Uses well-chosen functions, classes, GUI elements and event handling mechanisms. |  |  |  |
| Have sufficient comments in the program to describe its function and what each section does |  |  |  |
| Program works with expected and relevant boundary cases |  |  |  |
| Evidence of organised testing, showing program works for expected and relevant boundary cases |  |  |  |
| Program must be well structured and be laid out logically |  |  |  |
| Logical decomposition of the task |  |  |  |
| Have functions where they should reasonably be used |  |  |  |
| Use variables, constants and derived variables effectively and robustly |  |  |  |
| The program must be concise – be efficient with the amount of code required |  |  |  |
| Have comments that explain what each step is doing and justify why |  |  |  |
| Program works for expected, relevant boundary and invalid inputs |  |  |  |
| Evidence is provided to demonstrate that testing has been comprehensive |  |  |  |
| GitHub commit history demonstrates well-organised development process |  |  |  |

Teacher comments:

Program allows users to buy more tickets than are available. You need to add an error catch that prevents that from happening.

Docstring at the start of the program should describe what THIS program does, and docstrings in each class should accurately describe it.

Every function needs to have a docstring describing what it does. And every function needs comments within it explaining what the code is doing throughout the function.

Final grade: N A M E