|  |
| --- |
| Software Development Unit 3 Outcome 1Programming Folio School Assessed Coursework (SAC) Part 3 |

## **This SAC will consist of 4 Folio Tasks over an extended period of time.**

## U3 SAC 1:  Programming Folio (Contributes 10% of subject assessment)

## **Part 1:  T1W6 FRI 5 March (Double)**

## **Part 2:  T1W8 FRI 19 March (Double)**

## **Part 3:  T2W2 FRI 30 April (Double)**

## **Part 4:  T2W4 FRI 14 May (Double)**

**Outcome statement**

*On completion of this unit, the student should be able to interpret designs and apply a range of functions and techniques using a programming language to develop working modules.*

**Task Conditions**

1. **Allowed resources:** Teacher-provided solution designs, open book
2. **Time allocated to this task:** 2 periods (Double) Friday 30 April. **100 minutes**
3. **Marks allocated:** 34  
   **Location of files:** The zipped Visual Basic Project folder is to be submitted for marking Under:
4. **Submission of work: Learning Tasks:** “**Unit 3 SAC 1 Programming Folio 3**”.

**Task Outline**

Using the module requirements and provided designs, students are required to produce working software modules.

**Mr Mac’s Magnificent Motel**

|  |
| --- |
| **713 Vintage Neon Motel Sign Photos - Free & Royalty-Free Stock Photos from  DreamstimeMr Mac’s Magnificent Motel** is a newly established family business in the leafy suburbs of Macville. Mr Mac has requested that you add a user interface form that allows for the management of guests’ room bookings. Rooms are numbered from 1 to 25. |

**Functional Requirements**

|  |  |
| --- | --- |
| **FR** | **Description** |
| **FR01** | Can display a list of room numbers and their associated guest Names (booked rooms) |
| **FR02** | Can display a list of available hotel rooms (rooms not booked) |
| **FR03** | Can assign a room to a guest (make a booking) |
| **FR04** | Can unassign a guest to a room (make a room available) |
| **FR05** | Can search the bookings and state whether a particular room is Available or Not Available. |
| **FR06** | Can search the bookings and state whether a particular guest is currently booked into a room |
| **FR07** | Can search the master Customer File to see whether a particular guest exists |

**Non-Functional Requirements**

|  |  |
| --- | --- |
| **FR** | **Description** |
| **NFR01** | The application form size must fit the dimensions of a laptop |
| **NFR02** | The user interface must be intuitive for the user |
| **NFR03** | Font styles and colours must present a professional appearance |

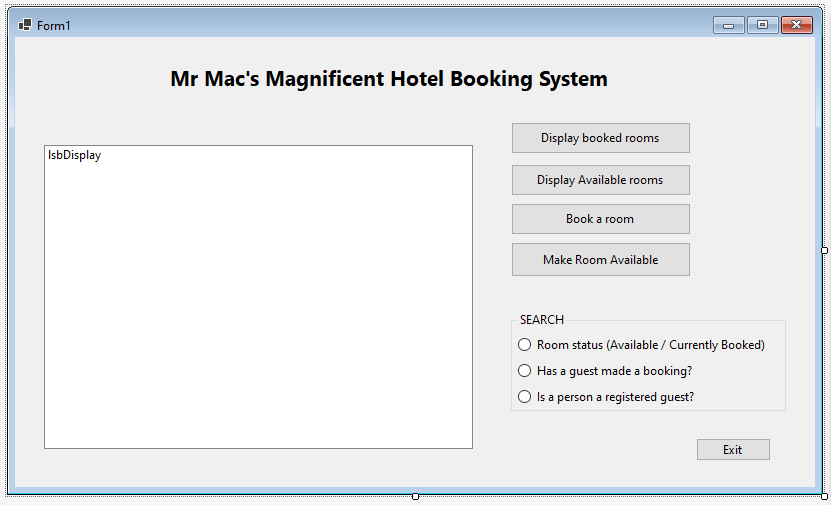
**Constraints**

|  |  |
| --- | --- |
| **FR** | **Description** |
| **C01** | The solution must be produced on the School laptop |
| **C02** | You must include a linear or binary search |
| **C03** | You must use a user-created function to return whether a room is Available or Not Available |
| **C04** | You must use the Visual Basic .NET programming language |
| **C05** | You must complete the task within 100 minutes. |

**IPO Chart**

|  |  |  |
| --- | --- | --- |
| **Input** | **Processing** | **Output** |
| * The room number | * check to see if the room has an associated guest | * A message whether the room is available or not available |
| * The guest Name | * Check to see if the guest is associated with a room | * A message stating whether a particular guest is currently booked into a room |
| * The guest Name | * Search the Customer File to see if the guest exists in the file | * If the guest exists, the guest details are displayed in the listBox * If the guest is not in the Customer File then display a message that “[guest name] does not exist in the Customer File.” |

**User Interface Mock-ups**



**Testing Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test No.** | **Test Item** | **Test data** | **Expected Result** | **Actual Result** |
| 1 | Add a key/value pair | **(Combobox) Name:** Otto King  **(Combobox) Item:** Room 1 | **Listbox:** '… has been booked by Otto King" (when display pairs button used)    **MsgBox:** "Booking successful" | As expected |
| 2 | Remove a key/value pair | **(InputBox) Item:** | **Listbox:** Line removed | As expected |
| 3 | Display all key/value pairs | Key / Value Pairs Added (booked) via usage of Test 2 | **Listbox:** "(Item)… Has Been booked by (Name)"    (Refer to snip for evidence) | As expected |
| 4 | Display a specific key/value pair | Text File Data | **Listbox:** "====Listing Pairs===="  Names From Array on each line | As expected |
| 5 | Determine if an item has / does not have a pair    (Item paired) | **(InputBox) Item:**    Item already booked by Otto King | **Listbox:** "====Search Item Results====" "Item is booked by Otto King" | As expected |
| 6 | Determine if an item has / does not have a pair    (Item not paired) | **(Inputbox) Item:**    \*Item already booked by Otto King | **Listbox:** "====Search Item Results====" "Item is not currently booked" | As expected |
| 7 | Search (linear) a textile to state whether a name is in the file    (Item in file) | **(InputBox) Name:** Otto King    \*Name is in file | **Listbox:** "====Search Name Results====" "Otto King is a registered guest" | As expected |
| 8 | Search (linear) a textile to state whether a name is in the file    (Item not in file) | **InputBox) Name:** Jimbob Bobsmithson    \*Name is not in file | **Listbox:** "====Search Name Results====" "Jimbob Bobsmithson was not found" | As expected |

**'NOTE**: Room availability search and guest booking search function only AFTER a booking has been made

**'JUSTIFICATIONS:**

'--Visual Basic .NET framework was suitable given built-in features such as ability to easily read and save

'--data from textfiles via StreamReader, and GUI creation options.

'--Do...While Loop statements were suitable as it is a control flow statement that can execute a block of code repeatedly until

'--meeting a condition at the end of the block. This was used as part of adding names from the textbox into the combobox,

'--looping through until the last name had been added, and in other subroutines.

'--IF...Else statements were suitable as it allowed execution of code given specified criteria was met (TRUE), while

'--another set of code evaluated FALSE. IF statements were used in conjunction with boolean Yes/No to add key/value pairs when

'--borrowing items and in other subroutines.

'--Try...Catch statements were suitable as it allowed a block of code to be tested for errors, and for another block of code to

'--be executed if an error did occur in the try statement via the catch. This was used to check if the user input both a key and

'--a value into the comboboxes, and prompt a msgbox telling them they had not if an error occured.

'--Select Case statements suitable as allowed for a specific block of code to be executed out of seleceted radiobox options.

'--This was used for the 3 search options to execute different coded for each search.

'--For...Next statements were suitable as it is an iterative, incremental loop statement used to repeat sequence of

'--statements for specific number of circumstances. Was used for searches to check whether items were borrowed, requiring only

'--one line to be published to the listbox as an output rather than multiple, which could be specified by this statement. Also

'--used to read through 100 lines of names in textbox and added to them an array.

**'VALIDATION:**

'Validation for comboboxes was integrated into program code, as a boolean yes/no served as an existence check for the

'input key / value to be paired during borrowing.

'The DropDownStyle property of the combobox was changed to DropDownList, which meant users could only input list items,

'not allowing any free user input, thereby circumventing the need for type/range validation code checks. Validation is not

'possible for built-in VB InputBoxes.

**Marking Scheme**

Each task will be assessed using the provided performance descriptors and table below.

|  |  |
| --- | --- |
| **Criteria / Skills** | **Marks Available** |
| Interpretation of designs to produce working modules | **3** |
| Data types and structures used | **3** |
| **Use** appropriate processing features of a programming language to develop working modules | **13** |
| **justify** appropriate processing features of a programming language to develop working modules | **4** |
| Develop and apply suitable validation techniques | **4** |
| Develop and apply suitable testing and debugging techniques using appropriate test data | **4** |
| document the functioning of modules and the use of processing features through internal documentation | **3** |
| **TOTAL** | **/34** |

VCE Applied Computing: Performance Descriptors

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOFTWARE DEVELOPMENT UNIT 3 OUTCOME 1 - SCHOOL-ASSESSED COURSEWORK** | | | | | |
| **Performance Descriptors - typical performance in each range** | | | | | |
| ***Unit 3***  ***Outcome 1***  ***Interpret teacher-provided solution requirements and designs, and apply a range of functions and techniques using a programming language to develop and test working software modules.*** | **Very low** | **Low** | **Medium** | **High** | **Very high** |
| Limited interpretation of solution requirements and designs to develop working modules. | Some interpretation of solution requirements and designs to develop working modules. | Sound interpretation of solution requirements and designs to develop working modules. | Most solution requirements and designs are interpreted accurately to developing working modules. | All solution requirements and designs are interpreted accurately to developing working modules. |
| Limited selection and use of data types and data structures. | Some selection and use of appropriate data types and data structures. | Sound selection and use of data types and data structures to develop working modules. | Detailed selection of relevant data types and data structures to develop working modules. | Comprehensive selection of relevant data types and data structures to develop working modules. |
| Limited selection and use of processing features of the programming language to develop some working modules. | Some selection and use of appropriate processing features of the programming language to develop some working modules. | Sound selection and use of appropriate processing features of the programming language to develop some working modules. | Most processing features of the programming language have been selected and used to develop all working modules. | Comprehensive selection and use of relevant processing features of the programming language to develop all working modules. |
| Limited explanation of how the selected processing features are used to develop working modules. | Some justification and explanation of how the selected processing features are used to develop working modules. | Sound justification and explanation of how the selection of appropriate processing features are used to develop working modules. | Detailed justification and explanation of how the selection of appropriate processing features of the programming language are used to develop working modules. | Comprehensive justification and explanation of how the selection of appropriate processing features of the programming language are used to develop working modules. |
| Limited data validation techniques are applied to check the reasonableness of some input data. | Some data validation techniques are effectively applied to check the reasonableness of some input data. | Sound use of data validation techniques are effectively applied to check the reasonableness of input data. | Detailed use of relevant data validation techniques are applied to efficiently and effectively check the reasonableness of all input data. | Comprehensive use of relevant data validation techniques are applied efficiently and effectively to check the reasonableness of all input data. |
| Limited range of test data is expressed in a testing table, with incomplete or missing results. | Some testing of test data is expressed in a testing table with actual output stated. | Sound range of testing of test data is expressed in a testing table, with both expected and actual output stated and some evidence of debugging. | Detailed use of test data is expressed in a testing table, with both expected and actual output stated with evidence of debugging. | Comprehensive use of test data is expressed in a testing table, with both expected and actual output stated, and showing detailed evidence of debugging. |
| Limited internal documentation with few comments regarding the use of the selected processing features. | Some internal documentation with comments regarding the functioning of modules and the use selected processing features. | Sound use of internal documentation with comments regarding the functioning of modules and the use of selected processing features. | Most software modules include detailed internal documentation regarding the functioning of modules and use of selected processing features. | All software modules include comprehensive internal documentation regarding the functioning of modules and use of selected processing features. |

KEY to marking scale based on the Outcome contributing 100 marks

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Very Low 1–20 | Low 21–40 | Medium 41–60 | High 61–80 | Very High 81–100 |