

# School of IT & Business Technologies Bachelor of Business Information Management Cover Sheet and Student Declaration

This sheet must be signed by the student and attached to the submitted assessment.

|  |  |  |  |
| --- | --- | --- | --- |
| **Course Title:** | **BBIM502 Introduction to Programming** | **Course code:** | BBIM502 |
| **Student Name:** |  | **Student ID:** | 764706455 |
| **Assessment No & Type:** | Assessment 2 Project | **Cohort:** | BBIM7123C |
| **Due Date:** | 31/10/2023 | **Date**  **Submitted:** | 31/10/2023 |
| **Tutor’s Name:** | Jatinder Singh |  |  |
| **Assessment** **Weighting** | 40% |  |  |
| **Total Marks** | 100 |  |  |

**Student Declaration:**

I declare that:

* I have read the New Zealand School of Education Ltd policies and regulations on assessments and understand what plagiarism is.
* I am aware of the penalties for cheating and plagiarism as laid down by the New Zealand School of Education Ltd.
* This is an original assessment and is entirely my own work.
* Where I have quoted or made use of the ideas of other writers, I have acknowledged the source.
* This assessment has been prepared exclusively for this course and has not been or will not be submitted as assessed work in any other course.
* It has been explained to me that this assessment may be used by NZSE Ltd, for internal and/or external moderation.
* If I am late in handing in this assessment without prior approval (see student regulations in handbook), marks will be deducted, to a maximum of 50%.

**Student signature:**

**Date:**

|  |  |  |
| --- | --- | --- |
| **Tutor only to complete** |  |  |
| **Assessment result:** | **Mark /100** | **Grade** |

© NZSE - BBIM502 Introduction to Programming Assessment 2 [Project] v1.1

Contents

[School of IT & Business Technologies Bachelor of Business Information Management Cover Sheet and Student Declaration 1](#_Toc149831098)

[Introduction 4](#_Toc149831099)

[Pseudocode 4](#_Toc149831100)

[Task 2 (Errors & Exception Handling) 9](#_Toc149831101)

[TASK 3 (Documentation) 12](#_Toc149831102)

[GitHub Repository Evidence 17](#_Toc149831103)

[Self-Reflection 18](#_Toc149831104)

## Introduction

This report covers the development of a console-based application for the Langham Hotels chain, a high-end lodging option in major New Zealand cities. The Langham Hotels are a subsidiary of the Elite Hotels Company Limited, with its headquarters located on Queen Street Auckland. The development of a console-based program that performs input/output (I/O) tasks, including exceptional handling, is the main objective of this research project. It acts as a pilot project; if it proves successful, it will eventually be merged with the database system. To display the program's capabilities, such as room allocation, deallocation, room status display, and other functionality, we will be utilizing our Python programming skills. The report will be supported by concise code documentation, a thorough user guide, error types, and self-reflection.

## Pseudocode

READ options .

Step 1: WHILE choice is not equal to 0:

DISPLAY the menu options.

Step 2 DISPLAY message “Enter Your Choice Number Here (0-9):”

Step 3 READ choice.

Step 4 IF choice is 0.

Exit the program.

Step 5 ELIF choice is 1.

Add Room module will run.

Step 5.1 DISPLAY Add Room choice.

Step 5.2 DISPLAY “Please enter the total number of rooms in the Hotel:”

Step 5.3 IF length of list of rooms is greater than 0.

Variable y is equal to length of list of rooms.

Step 5.4 ELSE

Variable y is equal to 0.

Step 5.5 FOR in range of number of rooms.

room is equal to Room.

list of rooms.APPEND room.

Step 5.6 FOR in range of y, y plus number of rooms

object room is equal to list of rooms[i]

Step 5.7 DISPLAY “Room Allocation {i + 1}:”

Step 5.8 DISPLAY “Please enter room number {i + 1}:”

Step 5.9 READ object room.Room Number

object room.is allocated is equal to FALSE.

list of rooms[i] is equal to object room.

Step 5.10 IF i > 0:

Step 5.11 FOR j in range of i:

Step 5.12 WHILE list of rooms[i].Room Number is equal to the previous input of list of rooms[j].Room Number:

Step 5.13 DISPLAY “Same room number already exists”.

Step 5.14 DISPLAY “Please enter a new room number {i + 1}:”

Step 5.15 READ object room.Room Number

room.is allocated is equal to FALSE.

list of rooms[i] is equal to object room.

Step 6 ELIF choice is 2.

Delete Room module will run.

Step 6.1 IF in list of rooms

Step 6.1 DISPLAY Delete Room choice.

Step 6.2 DISPLAY “Please enter the room number that you want to delete”.

Step 6.3 READ room number.

Step 6.4 FOR object room in list of rooms.

Step 6.5 IF object room is equal to room number.

List of rooms.remove object room.

Step 6.6 DISPLAY “Following rooms have been deleted”.

Step 6.7 ELSE

Step 6.8 DISPLAY “No rooms to delete. Please add rooms first”

Step 7 ELIF choice is 3.

Display Room Details module will run.

Step 7.1 IF in list of rooms.

Step 7.2 DISPLAY Display Room choice.

Step 7.3 FOR object room in list of rooms.

Step 7.4 DISPLAY “Room Number:{object room.Room Number}”

Step 7.5 ELSE

Step 7.6 DISPLAY “No rooms to display. Please add rooms first”.

Step 8 ELIF choice is 4.

Allocate Rooms module will run.

Step 8.1 DISPLAY “Allocate Rooms choice.”

Step 8.2 DISPLAY “How many rooms would you like to allocate?”

Step 8.3 READ allocate room.

Step 8.4 WHILE allocate room > length of list of rooms:

Step 8.5 DISPLAY “You cannot allocate more rooms than total number of rooms in the Hotel. Please enter number between 1- {length of the list of rooms)}:”

Step 8.6 DISPLAY “How many rooms would you like to allocate?”

Step 8.7 READ allocate room.

Step 8.8 DISPLAY “You are allocating {allocate room} room(s)”

i = 0

Step 8.9 WHILE I < allocate room:

room allocation is equal to Room Allocation.

customer is equal to Customer.

Step 8.10 DISPLAY “Room Allocation {i + 1}:”

Step 8.11 DISPLAY “Please search Room Number to allocate:”

Step 8.12 READ search room.

Step 8.13 FOR j in range of length of list of rooms:

Step 8.14 IF search room is matching with list of rooms[j].Room Number

Step 8.15 DISPLAY “Found matching room number to allocate.”

Step 8.16 IF list of rooms[j].is allocated is NOT allocated:

Step 8.17 DISPLAY “Room {list of rooms[j].Room Number} is empty”

Step 8.18 DISPLAY “Please enter Customer Number to allocate:”

Step 8.19 READ customer.Customer Number

Step 8.20 DISPLAY “Please enter Customer Name to allocate:”

Step 8.21 READ customer.Customer Name

list of rooms[j].is allocated is equal to TRUE.

Step 8.22 DISPLAY “Allocation has been done”.

room allocation.Allocated Room Number is equal to search room.

room allocation.Allocated Customer is equal to customer.

list of room allocations.APPEND room allocation

i += 1

Step 8.23 BREAK from the loop

Step 8.24 ELSE:

Step 8.25 DISPLAY “Room {list of rooms[j].RoomNo} is already occupied. Please enter another room to allocate".

i -= 1

Step 8.26 BREAK from the (WHILE) loop

Step 8.27 ELSE:

Step 8.28 WHILE j is matching the length of list of rooms – 1:

Step 8.29 DISPLAY “Could not find matching room number to allocate. Please enter correct room number or add room first”.

i -= 1

Step 8.30 BREAK

i += 1

Step 9 ELIF choice is 5.

Display Room Allocation Details module will run.

Step 9.1 IF in list of room allocations.

Step 9.2 DISPLAY Display Room Allocation Details choice.

Step 9.3 FOR object room in list of room allocations.

Step 9.4 DISPLAY “Allocated Room Number: {object room.Allocated Room Number}”

Step 9.5 DISPLAY “Customer Number: {object room.Allocated Customer.Customer Number}”

Step 9.6 DISPLAY “Customer Name: {object room.Allocated Customer.Customer Name}”

Step 9.7 ELSE:

Step 9.8 DISPLAY “No allocated rooms to display. Please allocate rooms first.”

Step 10 ELIF choice is 6.

Billing & Deallocation module will run.

Step 10.1 DISPLAY De-allocate rooms choice.

Step 10.2 DISPLAY “Please enter Room Number for billing and de-allocation:”

Step 10.3 READ room number.

Step 10.4 FOR object room number IN list of room allocations:

Step 10.5 IF object room number. Allocated Room Number matches room number.

Step 10.6 DISPLAY “Please enter the total number of days stayed:”

Step 10.7 READ days stayed.

rate is equal to 150.

room charge is equal to days stayed \* rate.

Step 10.8 DISPLAY “Room has been Billed for a total of $, room charge”.

list of room allocations.remove object room number.

Step 10.9 DISPLAY “Room has been successfully de-allocated {room number}”

Step 10.10 BREAK

Step 10.11 ELSE:

Step 10.12 DISPLAY “Invalid input. There is no Allocated Room’’.

Step 11 ELIF choice is 7.

Save the Room Allocation to a File module will run.

Step 11.1 DISPLAY Save the Room Allocations to a File

Step 11.2 WITH opening file path

Assigning variable {now} with date and time

Step 11.3 FOR object room allocated in list of room allocations.

Step 11.4 DISPLAY “Allocated Room Number: {object room Allocated.Allocated Room Number}”

Step 11.5 DISPLAY “Customer Number: {object room Allocated.Allocated Customer.Customer Number}”

Step 11.6 DISPLAY “Customer Name: {object room Allocated.Allocated Customer.Customer Name}

Step 11.7 DISPLAY “Current Date and Time is {now}”

Step 11.8 WRITE the file.

Step 11.9 CLOSE the file.

Step 11.10 DISPLAY “File saved as ‘LHMS\_764706455.txt’ under Document folder”.

Step 12 ELIF choice is 8.

Show the Room Allocation from a File module will run.

Step 12.1 DISPLAY Show the Room Allocations from a File

Step 12.2 WITH opening file path

Step 12.3 READ lines in file.

Step 12.4 FOR line in lines

Step 12.5 DISPLAY without whitespace

Step 13 ELIF choice is 9.

Backup module will run.

Step 13.1 DISPLAY Backup choice.

Step 13.2 WHILE file path back up exists.

Step 13.3 DISPLAY “'LHMS\_backup.txt' file already exist. Existing file will be deleted”.

remove file path back up.

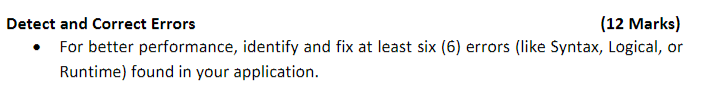
File path is renamed as file path back up.

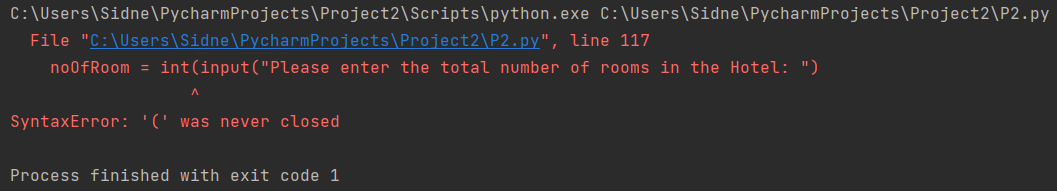
Step 14 ELSE:

Step 15 DISPLAY “Please choose (0-9)”

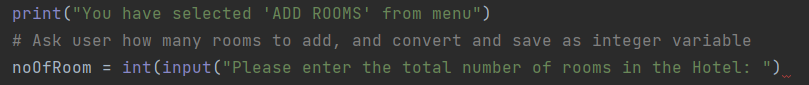
Step 16 END

## Task 2 (Errors & Exception Handling)

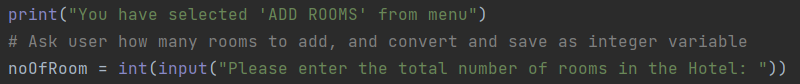


**Syntax Error**

Error missing a parenthesis.



Fixing error.



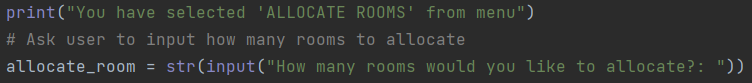
A Syntax Error occurs when there is an error in the syntax (structure) of the python code. To resolve this error, simply review the code to find underlined errors and input the correct syntax.

**Type Error**

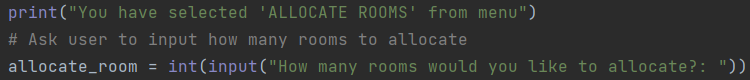
A screen shot of a computer

Description automatically generated

Type Error of string instead of integer.



Fixing error.



A Type Error is when an operator is supplied with a value of incorrect datatype, such as a string instead of a integer.

**Name Error**

A screenshot of a computer program

Description automatically generated

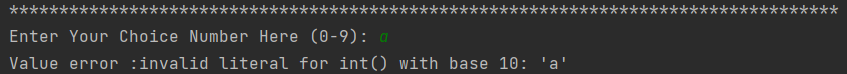
A Name Error is when a local or global variable name is not defined.

**Value Error**

A screen shot of a computer

Description automatically generated

Encountering error for wrong input of value, string instead of integer.



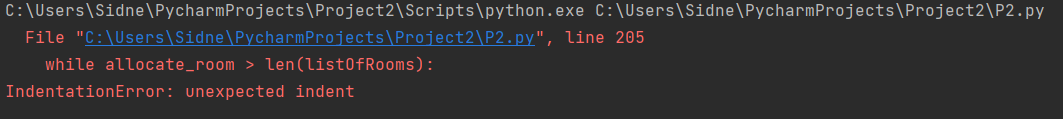
A Value Error is when a built-in method or operation receives an argument that has the right data type but mismatched or inappropriate values.

A screen shot of a computer error

Description automatically generated

To resolve the Value, Type and Syntax Errors, Try and Except blocks have been utilised to address these errors.

**Indentation Error**

****

Multiple spaces were placed ahead.

**A screenshot of a computer screen

Description automatically generated**

Fixing the code.

**A screen shot of a computer

Description automatically generated**

An indentation error occurs when the spaces or tabs are not placed properly.

**Recursion Error**

**A screen shot of a computer

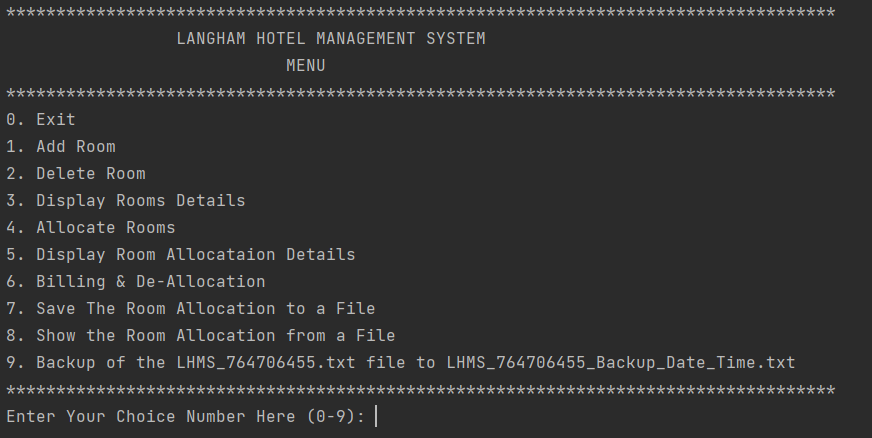
Description automatically generated**

A Recursion Error is caused by a function calling itself repeatedly. Python has a limit on the number of times a function can call itself recursively. This is to make sure that the function does not execute continuously. If this limit is gone beyond by a recursive function, a Recursion Error is raised. This can be resolved by using try and catch blocks to catch the recursion error.

## TASK 3 (Documentation)

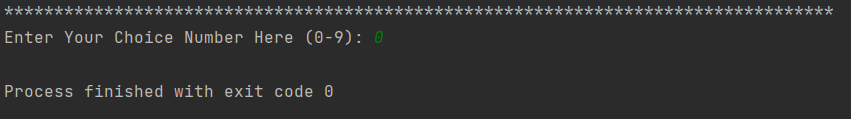


User guide with screenshots and documentation:



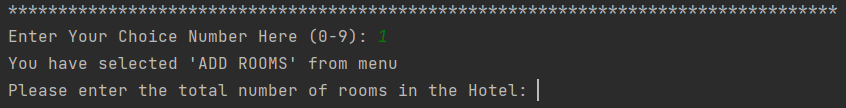
This is the Langham Hotel Management System. Enter a number through numbers (0-9) to select an option of your choice.

If 0 is selected:



This will Exit and finish the program.

If 1 is selected:



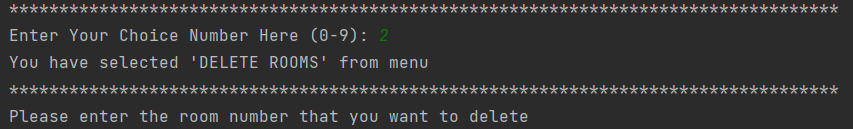
This is to add a room, where the user can enter the total number of rooms they wish to add.

A black rectangular frame with small blue and white figures

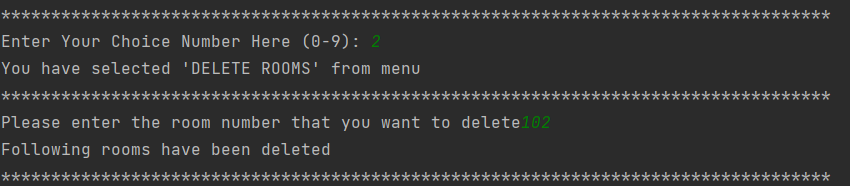
Description automatically generated

This is where the user will be asked to assign a room number to each given room.

If 2 is selected:



This is for deleting a room, where the user will enter the room number they wish to delete.



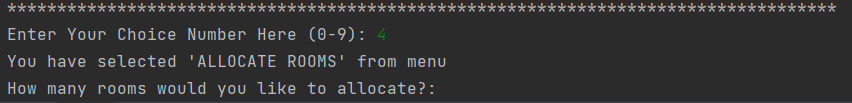
If 3 is selected:

A screen shot of a computer

Description automatically generated

This is to display the number of rooms added from option 1, in the beginning. As shown in the example of selection 2, deleting a room, room number 102 has been deleted.

If 4 is selected:



This is for allocating a room to a customer.

A screenshot of a computer program

Description automatically generated

After selecting how many rooms the user would like to allocate, the program will search to see if the room is empty or not. If the room is empty, the user will be asked to assign a customer number and customer name.

If 5 is selected:

A screen shot of a computer

Description automatically generated

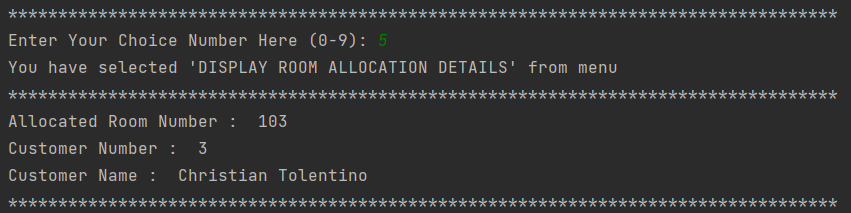
This is used to display the room allocation details, displaying the allocated room number, customer number, and customer name.

If 6 is selected:

A screen shot of a computer program

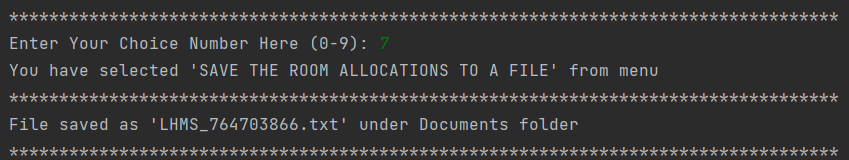
Description automatically generated

This is to deallocate and bill a room, charging the user per night.



The room has been deallocated.

If 7 is selected:



This is to save the room allocations to a file that will be found under the Documents folder.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The file containing the customer information.

If 8 is selected:

A screen shot of a computer

Description automatically generated

This is to show the room allocation from a file, within the program, displaying the customer information.

If 9 is selected:

A screenshot of a computer program

Description automatically generated

This is where the file be backed up under documents, and the original file will be deleted.

A screenshot of a computer

Description automatically generated

The backed-up file found under documents.

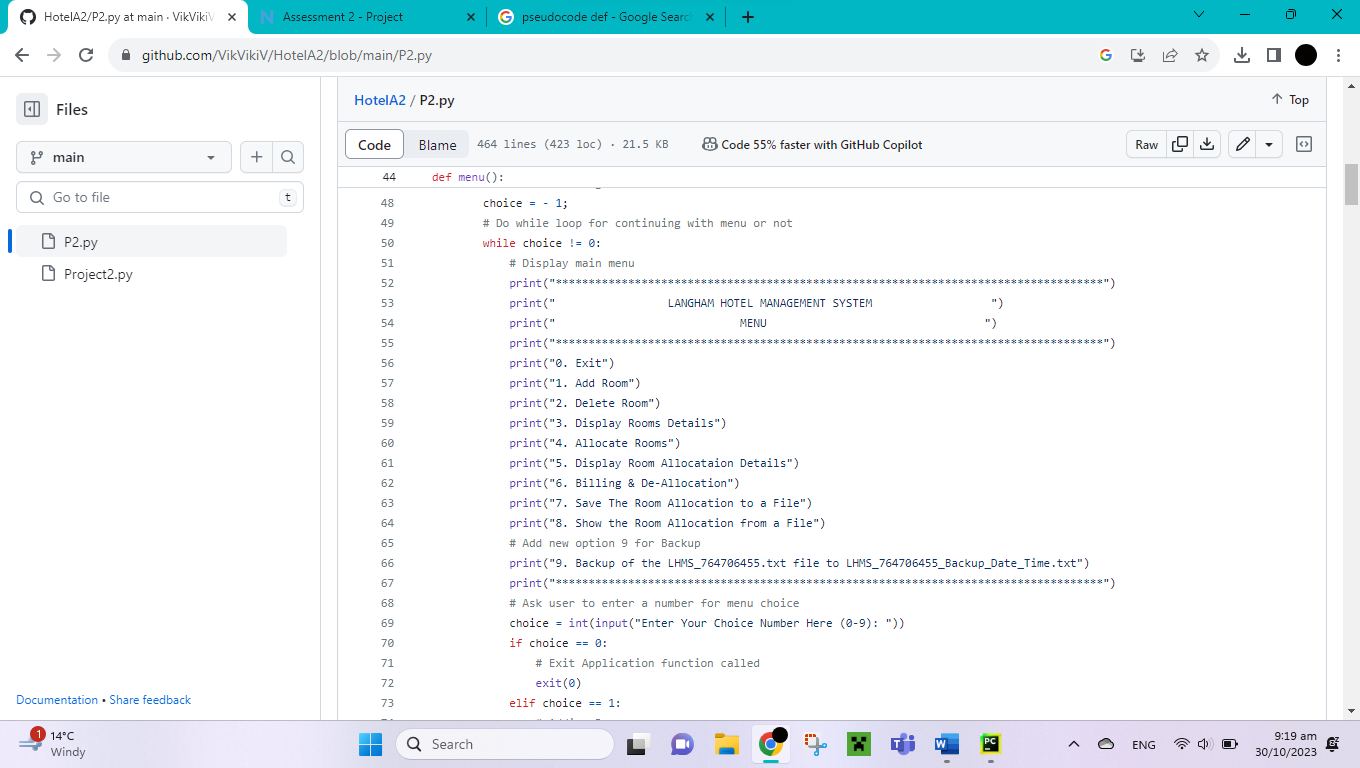
This will conclude the user guide for the Langham Hotel Management System.

## GitHub Repository Evidence



A screenshot of a computer

Description automatically generated



## Self-Reflection



In the process of constructing or converting the code from C# to Python, certain changes were made to make sense of how the code would function from a C# into a program written in Python code. In the given code of C#, the billing part of the program was missing, so it was needed to be implemented in the Python code. With the transfiguration of the code, most of the code was relatively easy to understand on how to change it into Python, but there were a few that were difficult to interpret, thus lending assistance from Sir Jatinder. Certain Errors were encountered and as a counter measure, Try and Except blocks were used to capture specific errors and solve them or provide an alternative solution. Overall, I learned to see how the code was broken into various phases for an easier understanding, breakdown, and implementation for a complete and functional Hotel Management program.