**Module Code & Module Title:**

**CS4051NT Fundamentals of Computing**

**Assessment Weightage & Type:**

**60% Individual Coursework**

**Year and Semester:**

**2024 Spring**

**Student Name: Manish Kumar Chaudhary**

**London Met ID: 23049274**

**College ID: np05cp4a230084**

**Assignment Due Date: May 7, 2024**

**Word Count:**

Table of content

## Table of figures

## Table of tables

## 1. Introduction to Module

The course is called "Fundamentals of Computing" and it lasts for one semester. This topic covered the fundamentals of flowcharts, pseudocode, and algorithms. Python is one of the essentials computer programming language that we studied.

This module covers Python operators, iteration and branching, data types for collections, manipulating strings, Python functions, and object-oriented programming. We will be using the Python programming language to construct a program for the Land Rental System of Techno property Nepal as part of the coursework of this module.

## 2. Introduction to Python

Python is a high-level, interpreted, object-oriented language with dynamic semantics (jaro education, 2024). Its dynamic typing and dynamic binding, along with its high-level built-in data structures, make it an appealing language for Rapid Application Development and for usage as a scripting or glue language to join existing components (jaro education, 2024). Because of its straightforward, basic syntax, Python emphasises readability, which lowers programme maintenance costs (jaro education, 2024). Python's support for packages and modules promotes code reuse and programme modularity (jaro education, 2024). The large standard library and the Python interpreter are freely distributable and accessible for free on all major platforms in source or binary form. (jaro education, 2024)

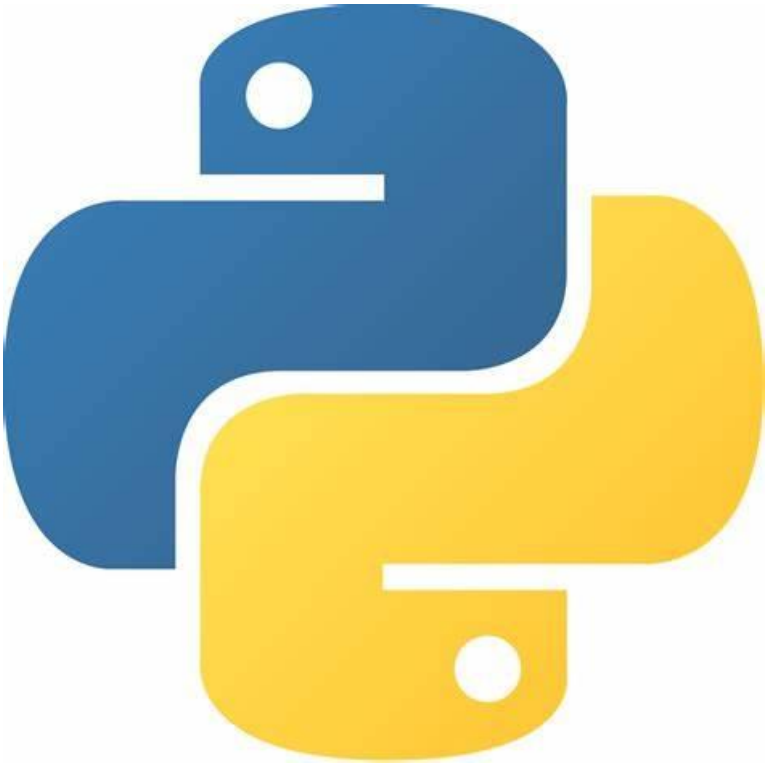

Figure 1 Logo of Python

## 3. About Coursework

In this coursework, I have created program for land renting system for Techno Nepal Property and make costumer to rent land of different available lands of Nepal from this program. I have created main.py python file in which all the code functionality are shown and also created txt file for storing the data of land. Some other python files are operation.py, read.py and write.py in which I have written all the required codes for a land rental system.

While using this coursework I have used different tools such as for code editing I have used IDLE code editor of python, I have used Ms word for documentation and draw.io for flowchart.


Figure 2 Tools that are used

## 4. Goals and objective

The objective of this program to rent available lands of different place of Nepal through this program easily and quickly. To save the time of costumer and main aim is to make student known about how to work on such program. The program's structure aims to facilitate land rental and return operations, maintaining records of transactions and updating land status accordingly. The use of file-based data storage allows for a simple yet effective way to manage and update land data.

## 5. Algorithm

An algorithm is a step-by-step procedure or set of rules for solving a specific problem or performing a task. Algorithms are fundamental in computer science and play a crucial role in various applications. Algorithm is helpful in breaking down the big problem into small steps to analyse quickly or helps in faster decision making. Below given the algorithm of my code:

**Step 1** - Start

- Print the welcome message for Techno Property Nepal

**Step 2** - Main Loop

- While True:
    - Print the main menu with options:
        - 1. Show all data
        - 2. Rent a Land
        - 3. Return a Land
        - 4. Exit

**Step 3** - Get User's Choice

- Get the user's choice (an integer from 1 to 4)

**Step 4** - Show All Data

- If choice is 1:
    - Call `read.show_data()`

**Step 5** - Rent a Land

- If choice is 2:
    - Call `read.available_land()`
    - Print a message asking for user details
    - Input user's name, address, and Kitta number
    - Open "file.txt" to check if the Kitta number is available
    - If Kitta is unavailable, print "Invalid Kitta number" and break
    - If Kitta is available, input number of months for renting
    - Call `write.rent_Info(kitta_no, name, address, months)`
    - Call `read.take_data(name, address, kitta_no, months)`
    - Call `operation.update_status(kitta_no)`
    - Call `write.write_record(name, address, kitta_no, months)`
    - Break the inner loop to return to the main menu

**Step 6** - Return a Land

- If choice is 3:
    - Print a message asking for user details
    - Input user's name, address, and Kitta number to return
    - Input number of delay months
    - Call `read.return_info(name, address, delay_month)`
    - Call `operation.return_status(kitta_no)`
    - Break the inner loop to return to the main menu

**Step 7** - Exit the Program

- If choice is 4:
    - Print "Thank you for renting from Techno Property Nepal"
    - Exit the program

**Step 8** - Handle Invalid Choice

- If the choice is invalid, print "Invalid option" and return to the main loop

**Step 9** - Ask to Perform More Operations

- Print a sub-menu asking if the user wants to continue
- If the user chooses "1" (Yes), continue to Step 2 - Main Loop
- If the user chooses "2" (No):
    - Print "Thank you for renting from Techno Property Nepal"
    - Break the outer loop
- If choice is invalid, print "Invalid choice" and re-enter the choice

**Step 10** - End

## 6. Introduction to Pseudocode

Pseudocode is a step-by-step representation of an algorithm. Pseudocode do not use any programming language in its representation (java T point, 2021). It uses the simple English language text which is human understanding rather than machine understanding (geeksforgeeks, 2023). Pseudocode is important part of designing an algorithm as it helps in planning the solution to the problem (geeksforgeeks, 2023). It is an intermediate state between algorithm and program (java T point, 2021).

### 6.1 Pseudocode for main.py

```
IMPORT read

IMPORT write

IMPORT operation


LOOP FOREVER:

    PRINT "\t----------------------------------------------------------------------\t"

    PRINT "\t-------------------Welcome to Techno Property Nepal-------------------\t"

    PRINT "\t----------------------------------------------------------------------\t"


    WHILE TRUE:

        PRINT "Select operation:"

        PRINT "1. Show all the data"

        PRINT "2. Rent a Land"

        PRINT "3. Return a Land"

        PRINT "4. Exit"
```

```
SET choice TO int(INPUT("Enter choice (1-4): "))


IF choice EQUALS 1:
    CALL read.show_data()


ELSE IF choice EQUALS 2:
    CALL read.available_land()
    PRINT "\t-----------------------------------------------------------------\t"
    PRINT "\t-------------------Please provide us your details---------------------\t"
    PRINT "\t-----------------------------------------------------------------\t"


    SET name TO INPUT("Enter your Name: ")
    SET address TO INPUT("Enter address: ")
    SET kitta_no TO INPUT("Enter Kitta Number that you want to rent: ")


    OPEN "file.txt" FOR READING AS f:
        FOR EACH line IN f:
            SPLIT line INTO record BY ", "
            IF record[0] EQUALS kitta_no:
                SET record[-1] TO "Not Available"
                PRINT "Invalid Kitta no"
                BREAK
```

```
        ELSE:
            SET months TO INPUT("Enter number of months you want to rent: ")
            CALL write.rent_Info(kitta_no, name, address, months)
            CALL read.take_data(name, address, kitta_no, months)
            CALL operation.update_status(kitta_no)
            CALL write.write_record(name, address, kitta_no, months)
            BREAK

    ELSE IF choice EQUALS 3:
        PRINT "\t-----------------------------------------------------------------\t"
        PRINT "\t-------------------Please provide us your details---------------------\t"
        PRINT "\t-----------------------------------------------------------------\t"

        SET name TO INPUT("Enter your Name: ")
        SET address TO INPUT("Enter address: ")
        SET kitta_no TO INPUT("Enter Kitta Number that you want to return: ")
        SET delay_month TO INPUT("Enter number of months you were delayed: ")

        CALL read.return_info(name, address, delay_month)
        CALL operation.return_status(kitta_no)

    ELSE IF choice EQUALS 4:
```

```
        PRINT "Thank you for renting from Techno Property Nepal"
        EXIT

    ELSE:
        PRINT "Invalid Option"
        CONTINUE

    WHILE TRUE:
        PRINT "Do you want to perform more operations?"
        PRINT "1. Yes"
        PRINT "2. No"

        SET new_choice TO INPUT("Enter choice (1-2): ")

        IF new_choice EQUALS "1":
            CONTINUE_OUTER_LOOP
            BREAK
        ELSE IF new_choice EQUALS "2":
            PRINT "Thank you for renting from Techno Property Nepal"
            EXIT
            BREAK
        ELSE:
```

```
            PRINT "Invalid choice"
            CONTINUE
```

## 6.2 Pseudocode for operation.py

```
DEFINE update_status(Kitta_no):
   SET updated_records TO []
   OPEN "C:\\Users\\User\\Downloads\\Python CourseWork\\file.txt" FOR READING ASf:
      FOR EACH line IN f:
         SPLIT line INTO record BY ", "
         IF record[0] EQUALS Kitta_no:
            SET record[-1] TO "Not Available"
         ADD ", ".join(record) TO updated_records

   OPEN "file.txt" FOR WRITING AS f:
      FOR EACH record IN updated_records:
         WRITE record + "\n" TO f


DEFINE return_status(Kitta_no):
   SET update_records TO []
   OPEN "file.txt" FOR READING AS f:
      FOR EACH line IN f:
```

```
        SPLIT line INTO record BY ", "
        IF record[0] EQUALS Kitta_no:
            SET record[-1] TO "Available"
        ADD ", ".join(record) TO update_records

    OPEN "file.txt" FOR WRITING AS f:
        FOR EACH record IN update_records:
        WRITE record + "\n" TO f
```

### 6.3 Pseudocode for write.py

```
IMPORT datetime


DEFINE write_record(Name, Address, Kitta_no, Months):
    WITH OPEN('file.txt', 'r') AS f:
        container = f.readlines()
        found_status = False
        FOR EACH contents IN container:
            content = contents.strip().split(',')
            IF content[0] EQUALS Kitta_no:
                content[4] = int(content[4]) * int(Months)


                # Creating invoice content
                invoice_content = (
                    '-------------------------------------\n'
                    f'Invoice Date: {datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")}\n'
                    '-------------------------------------\n'
                    f'Customer Name: {Name}\n'
                    '-------------------------------------\n'
                    f'Customer Address: {Address}\n'
                    '-------------------------------------\n'
```

```
        f'Kitta Number: {content[0]}\n'
        '-------------------------------------\n'
        f'City/District: {content[1]}\n'
        '-------------------------------------\n'
        f'Direction: {content[2]}\n'
        '-------------------------------------\n'
        f'Anna: {content[3]}\n'
        '-------------------------------------\n'
        f'Price: {content[4]}\n'
        '-------------------------------------\n'
        f'Status: {content[5]}\n'
        '-------------------------------------\n'
        f'Months Rented: {Months}\n'
        '=====================================\n'
    )

    # Writing invoice to a file
    invoice_filename = f'Invoice_{Kitta_no}_{Name.replace(" ", "_")}.txt'
    WITH OPEN(invoice_filename, 'w') AS invoice_file:
        invoice_file.write(invoice_content)

    found_status = True
```

```
            BREAK

        IF NOT found_status:
            PRINT "Kitta Number is not found, please enter valid one"

DEFINE rent_Info(Kitta_no, Name, Address, Months):
    WITH OPEN("C:\\Users\\User\\Downloads\\Python CourseWork\\file.txt", "r") AS f:
        rent = f.readlines()

    WITH OPEN('RentData.txt', 'a') AS f:
        FOR EACH contents IN rent:
            content = contents.strip().split(",")
            IF content[0] EQUALS Kitta_no:
                content[4] = int(content[4]) * int(Months)
                f.write(f'{Name},{Address},{Kitta_no},{content[4]},{Months}\n')
                BREAK
```

## 6.4 Pseudocode for read.py

```
IMPORT datetime


DEFINE take_data(Name, Address, Kitta_number, Months):
    WITH OPEN("file.txt", "r") AS f:
        container = f.readlines()
        FoundStatus = False
        FOR EACH contents IN container:
            content = contents.strip().split(",")
            IF content[0] EQUALS Kitta_number:
                content[4] = int(content[4]) * int(Months)
                content[5] = 'Rented'


                # Creating the invoice content
                invoice_content = (
                    f'Invoice Date: {datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")}\n'
                    f'-------------------------------------\n'
                    f'Customer Name: {Name}\n'
                    f'-------------------------------------\n'
                    f'Customer Address: {Address}\n'
                    f'-------------------------------------\n'
```

```
        f'Kitta Number: {content[0]}\n'
        f'-------------------------------------\n'
        f'Place: {content[1]}\n'
        f'-------------------------------------\n'
        f'Direction: {content[2]}\n'
        f'-------------------------------------\n'
        f'Anna: {content[3]}\n'
        f'-------------------------------------\n'
        f'Price: {content[4]}\n'
        f'-------------------------------------\n'
        f'Status: {content[5]}\n'
        f'-------------------------------------\n'
        f'Months Rented: {Months}\n'
        f'=====================================\n'
    )

    # Displaying the invoice in the terminal
    PRINT "=====================================\n"
    PRINT "Rent Invoice:\n"
    PRINT "=====================================\n"
    PRINT invoice_content
```

```
                FoundStatus = True
                BREAK
        IF NOT FoundStatus:
            PRINT "Entered Kitta Number is not found in the system"

DEFINE show_data():
    PRINT
"===========================================================================
===========================================================================
====================="
    PRINT "|\tKitta Number\t|\tcity/district\t\t|\tDirection\t|\tAnna\t | \tPrice\t
|\tAvailability\t |"
    PRINT
"===========================================================================
===========================================================================
====================="
    WITH OPEN("file.txt", "r") AS f:
        contents = f.readlines()
        FOR EACH content IN contents:
            cont = content.strip().split(",")
            PRINT f"|\t {cont[0]}    \t|\t {cont[1]}   \t| {cont[2]}    \t|\t {cont[3]}  \t |\t
{cont[4]}  \t|\t {cont[5]}   \t|"
            PRINT "-------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------"
```

```
DEFINE available_land():
    PRINT "=====================================================================================================================================================" 
    PRINT "|\tKitta Number\t|\tcity/district\t\t|\tDirection\t|\tAnna\t | \tPrice\t |\tAvailability\t |"
    PRINT "=====================================================================================================================================================" 
    WITH OPEN("C:\\Users\\User\\Downloads\\Python CourseWork\\file.txt", "r") AS f:
        contents = f.readlines()
        FOR EACH content IN contents:
            cont = content.strip().split(",")
            IF cont[5].strip().lower() EQUALS "available":
                PRINT f"|\t {cont[0]} \t|\t {cont[1]} \t| {cont[2]} \t|\t {cont[3]} \t |\t {cont[4]} \t|\t {cont[5]} \t|"


DEFINE return_info(Name, Address, Delaymonth):
    intDelaymonth = int(Delaymonth)  # Ensure Delaymonth is an integer
    TotalPayment = 0
    invoice_contents = []  # This will store each line of the invoice


    WITH OPEN("RentData.txt", "r") AS f:
```

```
    records = f.readlines()


  FOR EACH record IN records:
    content = record.strip().split(",")
    IF content[0] EQUALS Name AND content[1] EQUALS Address:
      payment_due = int(content[3]) + 5000 * intDelaymonth  # Calculate payment including delay penalty
      TotalPayment += payment_due
      invoice_line = (
        f'Name: {content[0]}\n'
        f'-------------------------------------\n'
        f'Address: {content[1]}\n'
        f'-------------------------------------\n'
        f'Kitta Number: {content[2]}\n'
        f'-------------------------------------\n'
        f'Months Rented: {content[4]}\n'
        f'-------------------------------------\n'
        f'Initial Payment: {content[3]}\n'
        f'-------------------------------------\n'
        f'Delay Penalty: {5000 * intDelaymonth}\n'
      )
      invoice_contents.append(invoice_line)
```

```
    IF invoice_contents:
        WITH OPEN(f'Return_Invoice_{Name}.txt', "a") AS f:
            FOR EACH line IN invoice_contents:
                f.write(line + '\n')

        PRINT "=====================================\n"
        PRINT "Return Invoice:\n"
        PRINT "=====================================\n"
        FOR EACH line IN invoice_contents:
            PRINT line

        PRINT '-----------------------------------------------------------------------------------------------
-'
        PRINT "Total Payment: ", TotalPayment
        PRINT '-----------------------------------------------------------------------------------------------
-'

    ELSE:
        PRINT "No such records found, please try again!"
```

## 7. Flowchart

Flowchart is a kind of diagram that shows a workflow or process of a program. Another definition of a flowchart is a diagrammatic description of an algorithm, which is a methodical process for completing a task. It shows method, procedure, or problem solutions step-by-step. It is a visual representation of the phases that most programmers at the beginning of the process find most helpful in understanding computer science algorithms, which helps debug the algorithm. An image of boxes showing the sequential process flow is called a flowchart. A flowchart makes a process or algorithm easy for anyone to understand since it is a visual representation of the process.

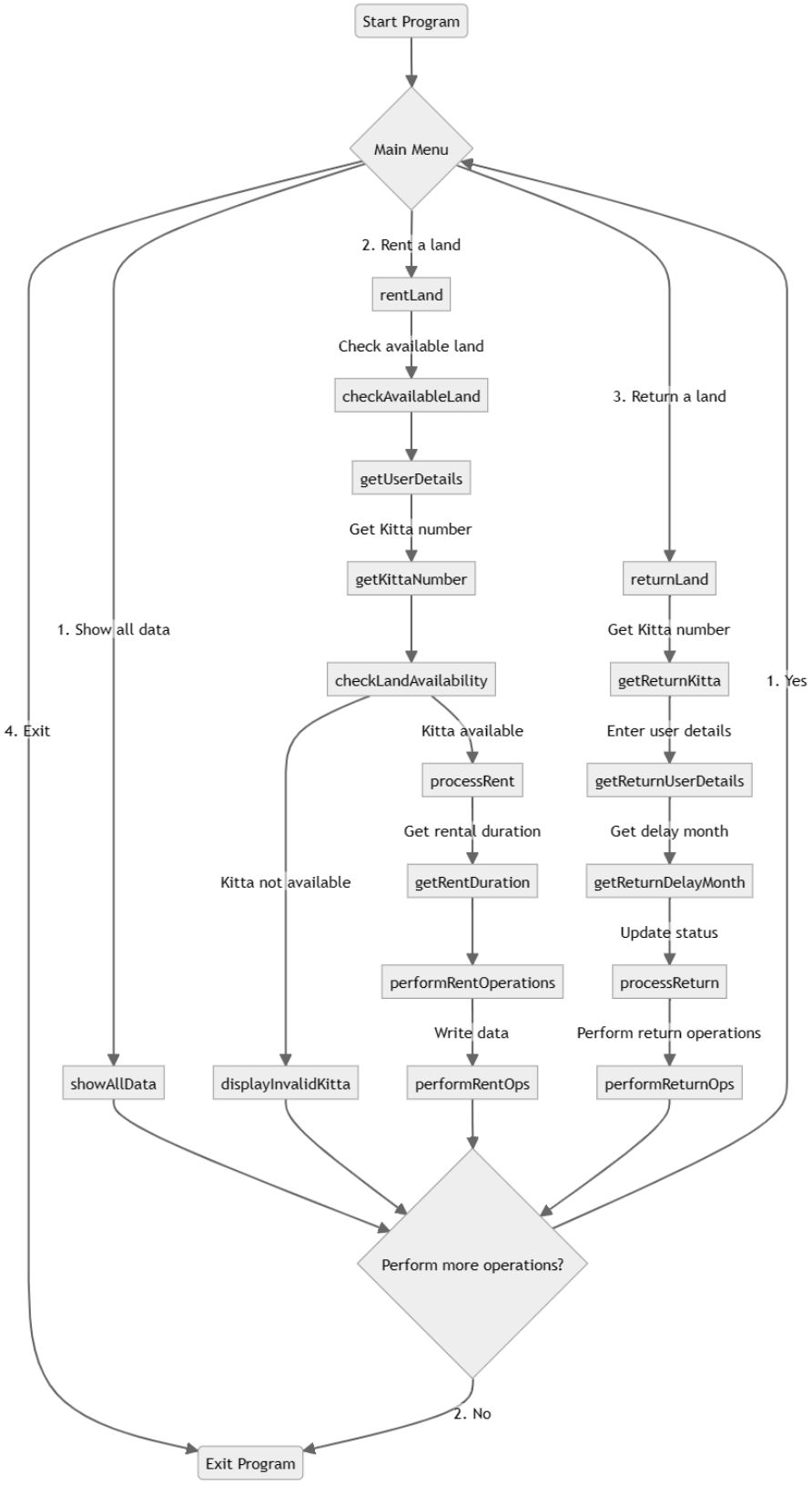


Figure 3 Flowchart of the program

## 8. Data Structure

A data structure is a way to store data (w3 school, 2024). Depending on the type of data we have and our purposes for it, we organise it differently. A family tree is used as the data structure when storing information about individuals who are connected to us. We want an overview so that we can quickly locate a certain family member, several generations back, and we have information about the people we are related to and how they are related, so we decided to utilise a family tree as the data structure (w3 school, 2024).

When one is presented with a family tree data structure like this, it becomes effortless to determine, for instance, the identity of my mother's mother—it's 'Emma,' right? (w3 school, 2024) But it would be challenging to ascertain the people' relationships if this data structure hadn't provided the ties from child to parents. With the help of data structures, we can effectively handle massive volumes of data for applications like internet indexing services and huge databases. Building quick and effective algorithms requires the use of data structures. They facilitate data management and organisation, simplify processes, and boost productivity.

There are two distinct types of data structures in computer science:

**8.1 Primitive Data Structures** are basic data structure provided by programming languages, which are simple data structures that are used to represent single values such characters, integers, floating-point numbers, and Booleans (w3 school, 2024).

**Boolean data type**

Boolean data type has two values i.e. true or false. The operators defined for the Boolean are and, or and not. Example: a && b, a || b,!a

```
while True:
```

Figure 4 Boolean Data type

**8.2 Non-Primitive Data Structures** types are created by the programmer using classes or structures. It offer more complex and specialised operations and are constructed using simple data types. Abstract data structures are frequently represented by trees, graphs, queues, stacks, linked lists, arrays, and stacks.

**List**

List is one of the data structure which is mutable, once the object is created, its internal state can be changed. Element or value inside a list is called item. In one dimensional list the elements are listed one after another. I have used 1D list to store the data from the text file

```
updated_records = []
```

Figure 5 List

I have used for loop to iterate over a list and checking the details

```
def write_record(Name, Address, Kitta_no, Months):
    with open('file.txt', 'r') as f:
        container = f.readlines()
        found_status = False
        for contents in container:
            content = contents.strip().split(',')
            if content[0] == Kitta_no:
                content[4] = int(content[4]) * int(Months)
```

Figure 6 iterate over a list with for loop

I have used readlines to read the text file

```
def write_record(Name, Address, Kitta_no, Months):
    with open('file.txt', 'r') as f:
        container = f.readlines()
        found_status = False
        for contents in container:
            content = contents.strip().split(',')
            if content[0] == Kitta_no:
                content[4] = int(content[4]) * int(Months)
```

*Figure 7 Screenshot showing the use of readlines*

I have used append to write in rentdata text and other invoice text file

```
with open('RentData.txt', 'a') as f:  # Open file in write mode ('w') outsid
    for contents in rent:
        content = contents.strip().split(",")
        if content[0] == Kitta_no:
            content[4]= int(content[4])*int(Months)
            f.write(f'{Name},{Address},{Kitta_no},{content[4]},{Months}\n')
            break
```

Figure 8 Screenshot showing use of append

## 9. Implementation of the overall Program

**Main User Interaction Loop:** The outer while True loop initializes the program with a welcome message and then enters an inner loop where users can choose an operation from a menu. The operations include:

- Displaying all land data.
- Renting a land plot.
- Returning a rented land plot.
- Exiting the program.

**Operations on Land Data:** The operations for renting and returning land plots involve reading and updating data from a text file ("file.txt"). This data represents the land plots, with each record containing details like Kitta number, city/district, direction, price, and availability status.

**Renting a Land Plot:** The user inputs their name, address, Kitta number, and rental duration in months. The program checks if the Kitta number exists and is available for rent. If valid, it updates the land's status to "Not Available", calculates the rent price, and generates an invoice. This information is then written to the "RentData.txt" file.

**Returning a Rented Land Plot:** The user provides their name, address, Kitta number, and any delay months. The program checks the return data and updates the land status to "Available." If there was a delay in returning, a penalty fee is added to the invoice. The information is also recorded in a text file.

**Invoice Generation:** For both renting and returning operations, the code creates invoices that detail the transaction. These invoices are generated and displayed on the console and written to text files for record-keeping. For rented land plots, invoices include customer details, Kitta number, rental price, and duration. For returned plots, invoices may include a delay penalty, reflecting additional charges for late returns.

**Status Updates:** The update_status function updates the land's status to "Not Available," indicating it has been rented. The return_status function updates the status

to "Available," signifying the land has been returned. These functions read the existing data, update the appropriate record(s), and write the changes back to the text file.

**Additional Error Handling:** The code checks for invalid input for Kitta numbers, rental duration, and delay months. If invalid input is detected, appropriate error messages are displayed, and the user is prompted to try again.

Overall, the program's structure aims to facilitate land rental and return operations, maintaining records of transactions and updating land status accordingly. The use of file-based data storage allows for a simple yet effective way to manage and update land data.

## 10.11 Renting a land

```
*IDLE Shell 3.12.2*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\User\Downloads\Python CourseWork\main.py
        ---------------------------------------------------------------------
        -------------------Welcome to Techno Property Nepal-------------------
        ---------------------------------------------------------------------
Select operation:
1. Choose 1 to Show all the data
2. Choose 2 to Rent a Land
3. Choose 3 to Return a Land
4. Choose 4 to Exit
Enter choice (1-4): 2
=================================================================================================================================================
|       Kitta Number    |       city/district           |       Direction       |       Anna    |       Price           |       Availability            |
=================================================================================================================================================
|        101            |         Hetauda               |  Central              |         13    |        45000          |         Available             |
|        102            |         Dhangadhi             |  West                 |         17    |        70000          |         Available             |
|        104            |         Pokhara               |  West                 |         19    |        85000          |         Available             |
|        105            |         Janakpur              |  East                 |         14    |        46000          |         Available             |
|        107            |         Bhairahawa            |  South                |         11    |        48000          |         Available             |
|        109            |         Tikapur               |  West                 |         15    |        62000          |         Available             |
|        110            |         Inaruwa               |  East                 |         14    |        35000          |         Available             |
|        113            |         Dharan                |  East                 |         15    |        48000          |         Available             |
|        114            |         Siraha                |  East                 |         12    |        50000          |         Available             |
|        115            |         Butwal                |  West                 |         18    |        78000          |         Available             |
        ---------------------------------------------------------------------
        -------------------Please provide us your details---------------------
        ---------------------------------------------------------------------
```

Figure 9 renting a land

### 10.12 Invoice of that rented land

```
=====================================

Rent Invoice:

=====================================

Invoice Date: 2024-05-04 22:16:33
-------------------------------------
Customer Name: jiten
-------------------------------------
Customer Address: babiya
-------------------------------------
Kitta Number: 104
-------------------------------------
Place:  Pokhara
-------------------------------------
Direction:  West
-------------------------------------
Anna:  19
-------------------------------------
Price: 4590000
-------------------------------------
Status: Rented
-------------------------------------
Months Rented: 54
=====================================
```

Figure 10 Invoice of that rented land

## 10.13 Returning that land

```
Do you want to perform more operation:
1. Yes
2. No
Enter choice (1-2): 1
Select operation:
1. Choose 1 to Show all the data
2. Choose 2 to Rent a Land
3. Choose 3 to Return a Land
4. Choose 4 to Exit
Enter choice (1-4): 3
        ----------------------------------------------------------------------
        --------------------Please provide us your details---------------------
        ----------------------------------------------------------------------
Enter your Name: jiten
Enter address: babiya
Enter Kitta Number that you want to return: 104
Enter number of months you were  delay: 5
=====================================

Return Invoice:

=====================================

Name: jiten
-------------------------------------
Address: babiya
-------------------------------------
Kitta Number: 104
-------------------------------------
Months Rented: 54
-------------------------------------
Initial Payment: 4590000
-------------------------------------
Delay Penalty: 25000
=====================================

-------------------------------------------------------------------------------------------------------
Total Payment:  4615000
-------------------------------------------------------------------------------------------------------
```

Figure 11 returning that land

## 10.14 Creation of Text file

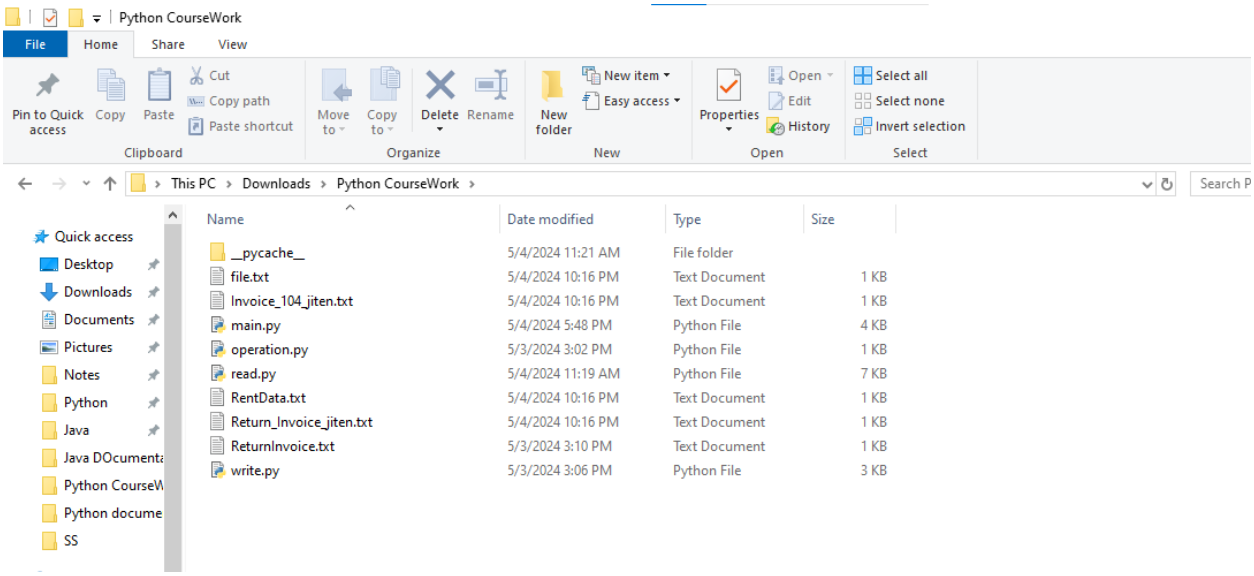


Figure 12 Creation of text file

## 10.15 Opening the bill text


Figure 13 Opening bill text file

## 10.16 Termination of program

```
        ----------------------------------------------------------------------
        -------------------Welcome to Techno Property Nepal-------------------
        ----------------------------------------------------------------------
Select operation:
1. Choose 1 to Show all the data
2. Choose 2 to Rent a Land
3. Choose 3 to Return a Land
4. Choose 4 to Exit
Enter choice (1-4): 4
Thank you for renting from Techno Property Nepal
>>>
```

Figure 14 Termination of program using option

# 10. Testing of the program

Testing is done to verify whether a program matches its expected requirements and to ensure the program is free from bug. By doing testing, we can prevent bugs and improve our code.

## 10.1 Test 1

| | |
|---|---|
| Objective | To show implementation of try, except |
| Action | Provided invalid input as one |
| Expected Result | Error message should be displayed |
| Actual Result | Error message displayed |
| Conclusion | Test is successful |

*Table 1 Table 1 Test 1 for try and except*

**Evidence of test 1**

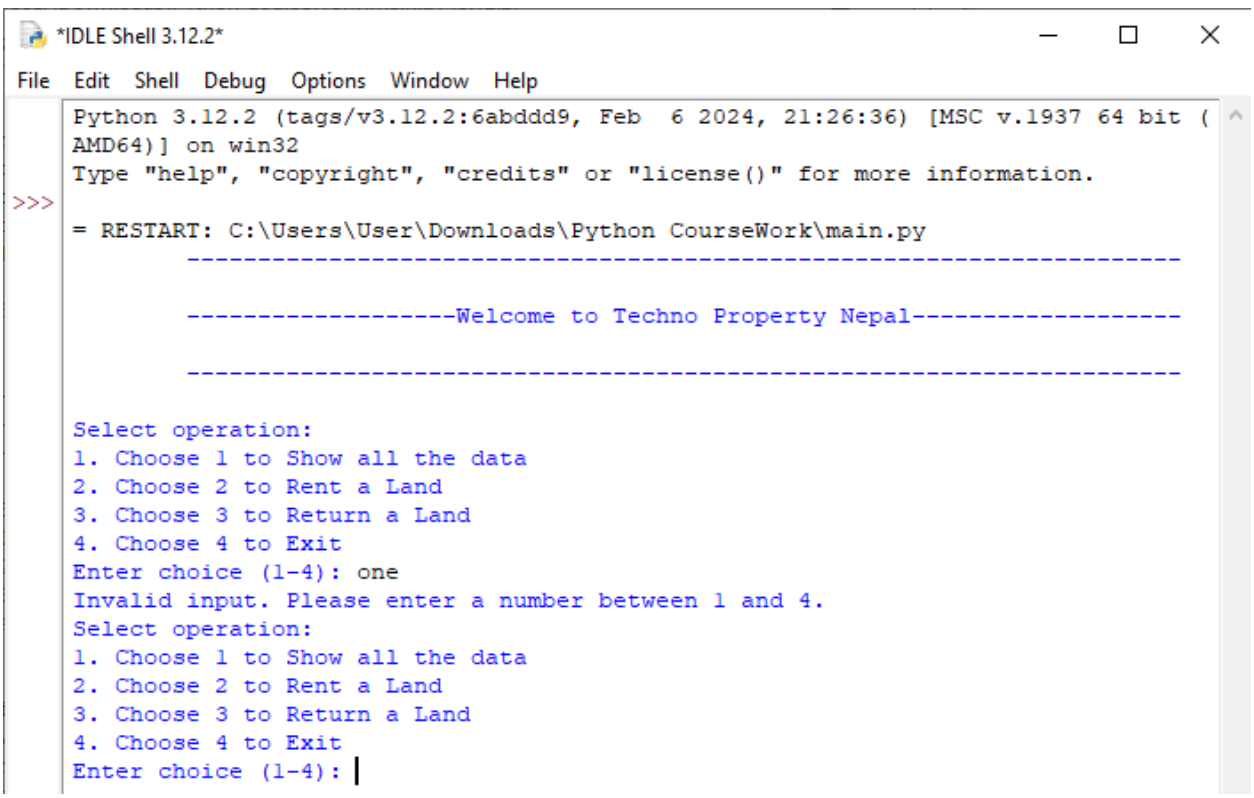


Figure 15 Evidence of test 1

## 10.2 Test 2

| | |
|---|---|
| Objective | To rent and return of lands |
| Action | Provided negative kitta number and not exited kitta number |
| Expected Result | Warning message should be displayed |
| Actual Result | Warning message displayed |
| Conclusion | Test is successful |

Table 2 test 2 to rent and return land

### Evidence of test 2

```
Select operation:
1. Choose 1 to Show all the data
2. Choose 2 to Rent a Land
3. Choose 3 to Return a Land
4. Choose 4 to Exit
Enter choice (1-4): 2
==========================================================================================================================
|       Kitta Number    |       city/district           |       Direction       |       Anna      |       Price           |       Availability            |
==========================================================================================================================
|        101            |         Hetauda               |  Central              |         13      |        45000          |         Available             |
|        102            |         Dhangadhi             |  West                 |         17      |        70000          |         Available             |
|        104            |         Pokhara               |  West                 |         19      |        85000          |         Available             |
|        105            |         Janakpur              |  East                 |         14      |        46000          |         Available             |
|        107            |         Bhairahawa            |  South                |         11      |        48000          |         Available             |
|        109            |         Tikapur               |  West                 |         15      |        62000          |         Available             |
|        110            |         Inaruwa               |  East                 |         14      |        35000          |         Available             |
|        113            |         Dharan                |  East                 |         15      |        48000          |         Available             |
|        114            |         Siraha                |  East                 |         12      |        50000          |         Available             |
|        115            |         Butwal                |  West                 |         18      |        78000          |         Available             |
        ----------------------------------------------------------------------
        -------------------Please provide us your details---------------------
        ----------------------------------------------------------------------
Enter your Name: manish
Enter address: inaruwa
Enter Kitta Number that you want to rent: -102
Invalid input: Kitta number must be a positive integer.. Please try again.
```

*Figure 16 Evidence warning message shown after providing negative value*

```
*IDLE Shell 3.12.2*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\User\Downloads\Python CourseWork\main.py
        ----------------------------------------------------------------------

        -------------------Welcome to Techno Property Nepal-------------------

        ----------------------------------------------------------------------

Select operation:
1. Choose 1 to Show all the data
2. Choose 2 to Rent a Land
3. Choose 3 to Return a Land
4. Choose 4 to Exit
Enter choice (1-4): 3
        ----------------------------------------------------------------------

        -------------------Please provide us your details---------------------

        ----------------------------------------------------------------------

Enter your Name: manish
Enter address: inaruwa
Enter Kitta Number that you want to return: 116
Enter number of months you were  delay: 5
No such records found, please try again!
```

*Figure 17 Evidence of trying invalid kitta no*

## 10.3 Test 3

| | |
|---|---|
| Objective | File generation of renting multiple of lands |
| Action | Multiple lands rented |
| Expected Result | File generation of that lands |
| Actual Result | File generated |
| Conclusion | Test successful |

Table 3 test 3 file generation for renting land

### Evidence of test 3

```
*IDLE Shell 3.12.2*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\User\Downloads\Python CourseWork\main.py ==========
        ---------------------------------------------------------------------
        -------------------Welcome to Techno Property Nepal-------------------
        ---------------------------------------------------------------------
Select operation:
1. Choose 1 to Show all the data
2. Choose 2 to Rent a Land
3. Choose 3 to Return a Land
4. Choose 4 to Exit
Enter choice (1-4): 2
=================================================================================================================================
|       Kitta Number    |       city/district           |       Direction       |       Anna    |       Price           |       Availability
=================================================================================================================================
|        101            |        Hetauda                | Central               |        13     |        45000          |        Available
|        102            |        Dhangadhi              | West                  |        17     |        70000          |        Available
|        104            |        Pokhara                | West                  |        19     |        85000          |        Available
|        105            |        Janakpur               | East                  |        14     |        46000          |        Available
|        107            |        Bhairahawa             | South                 |        11     |        48000          |        Available
|        109            |        Tikapur                | West                  |        15     |        62000          |        Available
|        110            |        Inaruwa                | East                  |        14     |        35000          |        Available
|        113            |        Dharan                 | East                  |        15     |        48000          |        Available
|        114            |        Siraha                 | East                  |        12     |        50000          |        Available
|        115            |        Butwal                 | West                  |        18     |        78000          |        Available
        ---------------------------------------------------------------------
        -------------------Please provide us your details---------------------
        ---------------------------------------------------------------------
Enter your Name: manish
Enter address: inaruwa
Enter Kitta Number that you want to rent: 115
Enter numbers of months you want to rent: 13
=====================================

Rent Invoice:

=====================================

Invoice Date: 2024-05-05 00:11:42
-------------------------------------
```

*Figure 18 process of renting multiple land*

```
*IDLE Shell 3.12.2*
File  Edit  Shell  Debug  Options  Window  Help
Enter address: inaruwa
Enter Kitta Number that you want to rent: 115
Enter numbers of months you want to rent: 13
=====================================

Rent Invoice:

=====================================

Invoice Date: 2024-05-05 00:11:42
-------------------------------------
Customer Name: manish
-------------------------------------
Customer Address: inaruwa
-------------------------------------
Kitta Number: 115
-------------------------------------
Place:  Butwal
-------------------------------------
Direction:  West
-------------------------------------
Anna:  18
-------------------------------------
Price: 1014000
-------------------------------------
Status: Rented
-------------------------------------
Months Rented: 13
=====================================

Do you want to perform more operation:
1. Yes
2. No
Enter choice (1-2): 1
Select operation:
1. Choose 1 to Show all the data
2. Choose 2 to Rent a Land
3. Choose 3 to Return a Land
4. Choose 4 to Exit
Enter choice (1-4): 2
=====================================================================================================
```

Figure 19 process of renting multiple land

```
*IDLE Shell 3.12.2*
File  Edit  Shell  Debug  Options  Window  Help
1. Choose 1 to Show all the data
2. Choose 2 to Rent a Land
3. Choose 3 to Return a Land
4. Choose 4 to Exit
Enter choice (1-4): 2
=================================================================================================================================================
|       Kitta Number    |       city/district           |       Direction       |       Anna    |       Price           |       Availability            |
=================================================================================================================================================
|        101            |        Hetauda                |  Central              |        13     |        45000          |        Available              |
|        102            |        Dhangadhi              |  West                 |        17     |        70000          |        Available              |
|        104            |        Pokhara                |  West                 |        19     |        85000          |        Available              |
|        105            |        Janakpur               |  East                 |        14     |        46000          |        Available              |
|        107            |        Bhairahawa             |  South                |        11     |        48000          |        Available              |
|        109            |        Tikapur                |  West                 |        15     |        62000          |        Available              |
|        110            |        Inaruwa                |  East                 |        14     |        35000          |        Available              |
|        113            |        Dharan                 |  East                 |        15     |        48000          |        Available              |
|        114            |        Siraha                 |  East                 |        12     |        50000          |        Available              |
        ---------------------------------------------------------------------
        -------------------Please provide us your details---------------------
        ---------------------------------------------------------------------
Enter your Name: manish
Enter address: inaruwa
Enter Kitta Number that you want to rent: 110
Enter numbers of months you want to rent: 65
=====================================

Rent Invoice:

=====================================

Invoice Date: 2024-05-05 00:12:18
-------------------------------------
Customer Name: manish
-------------------------------------
Customer Address: inaruwa
-------------------------------------
Kitta Number: 110
-------------------------------------
Place:  Inaruwa
-------------------------------------
Direction:  East
Ln: 120  Col: 20
```

Figure 20 process of renting multiple land

```
        ----------------------------------------------------------------------
Enter your Name: manish
Enter address: inaruwa
Enter Kitta Number that you want to rent: 110
Enter numbers of months you want to rent: 65
=====================================

Rent Invoice:

=====================================

Invoice Date: 2024-05-05 00:12:18
-------------------------------------
Customer Name: manish
-------------------------------------
Customer Address: inaruwa
-------------------------------------
Kitta Number: 110
-------------------------------------
Place:  Inaruwa
-------------------------------------
Direction:  East
-------------------------------------
Anna:  14
-------------------------------------
Price: 2275000
-------------------------------------
Status: Rented
-------------------------------------
Months Rented: 65
=====================================

Do you want to perform more operation:
1. Yes
2. No
```

Figure 21 process of renting multiple land

| Name | Date modified | Type | Size |
|---|---|---|---|
| __pycache__ | 5/5/2024 12:11 AM | File folder | |
| file.txt | 5/5/2024 12:12 AM | Text Document | 1 KB |
| Invoice_110_manish.txt | 5/5/2024 12:12 AM | Text Document | 1 KB |
| Invoice_115_manish.txt | 5/5/2024 12:11 AM | Text Document | 1 KB |
| main.py | 5/4/2024 11:50 PM | Python File | 4 KB |
| operation.py | 5/3/2024 3:02 PM | Python File | 1 KB |
| read.py | 5/4/2024 11:19 AM | Python File | 7 KB |
| RentData.txt | 5/5/2024 12:20 AM | Text Document | 1 KB |
| ReturnInvoice.txt | 5/3/2024 3:10 PM | Text Document | 1 KB |
| write.py | 5/3/2024 3:06 PM | Python File | 3 KB |

Figure 22 text file created

Invoice_110_manish.txt - Notepad

File Edit Format View Help

```
-------------------------------------
Invoice Date: 2024-05-05 00:12:18
-------------------------------------
Customer Name: manish
-------------------------------------
Customer Address: inaruwa
-------------------------------------
Kitta Number: 110
-------------------------------------
City/District:  Inaruwa
-------------------------------------
Direction:  East
-------------------------------------
Anna:  14
-------------------------------------
Price: 2275000
-------------------------------------
Status:  Not Available
-------------------------------------
Months Rented: 65
=====================================
```

Invoice_115_manish.txt - Notepad

File Edit Format View Help

```
-------------------------------------
Invoice Date: 2024-05-05 00:11:42
-------------------------------------
Customer Name: manish
-------------------------------------
Customer Address: inaruwa
-------------------------------------
Kitta Number: 115
-------------------------------------
City/District:  Butwal
-------------------------------------
Direction:  West
-------------------------------------
Anna:  18
-------------------------------------
Price: 1014000
-------------------------------------
Status:  Not Available
-------------------------------------
Months Rented: 13
=====================================
```

Figure 23 Text file opened

## 10.4 Test 4

| | |
|---|---|
| Objective | To return multiple lands |
| Action | Multiple lands returned |
| Expected result | File should be generated |
| Actual result | File generated |
| Conclusion | Test successful |

Table 4 File generation for returning land

### Evidence of text 4

```
*IDLE Shell 3.12.2*
File  Edit  Shell  Debug  Options  Window  Help
Enter choice (1-2): 1
Select operation:
1. Choose 1 to Show all the data
2. Choose 2 to Rent a Land
3. Choose 3 to Return a Land
4. Choose 4 to Exit
Enter choice (1-4): 3
        ------------------------------------------------------------------------
        -------------------Please provide us your details---------------------
        ------------------------------------------------------------------------
Enter your Name: manish
Enter address: inaruwa
Enter Kitta Number that you want to return: 115
Enter number of months you were  delay: 2
=====================================

Return Invoice:

=====================================

Name: manish
-------------------------------------
Address: inaruwa
-------------------------------------
Kitta Number: 115
-------------------------------------
Months Rented: 13
-------------------------------------
Initial Payment: 1014000
-------------------------------------
Delay Penalty: 10000
=====================================

Name: manish
-------------------------------------
Address: inaruwa
-------------------------------------
Kitta Number: 110
-------------------------------------
Months Rented: 65
-------------------------------------
```

Figure 24 Process of returning land

```
*IDLE Shell 3.12.2*
File  Edit  Shell  Debug  Options  Window  Help

=====================================

Name: manish
-------------------------------------
Address: inaruwa
-------------------------------------
Kitta Number: 115
-------------------------------------
Months Rented: 13
-------------------------------------
Initial Payment: 1014000
-------------------------------------
Delay Penalty: 10000
=====================================

Name: manish
-------------------------------------
Address: inaruwa
-------------------------------------
Kitta Number: 110
-------------------------------------
Months Rented: 65
-------------------------------------
Initial Payment: 2275000
-------------------------------------
Delay Penalty: 10000
=====================================

-------------------------------------------------------------------------------------------
Total Payment:  3309000
-------------------------------------------------------------------------------------------
        ------------------------------------------------------------------
        -------------------Welcome to Techno Property Nepal-------------------
        ------------------------------------------------------------------
Select operation:
1. Choose 1 to Show all the data
2. Choose 2 to Rent a Land
3. Choose 3 to Return a Land
4. Choose 4 to Exit
Enter choice (1-4): |
```

Figure 25 Process of returning land

This PC › Downloads › Python CourseWork

| Name | Date modified | Type | Size |
|---|---|---|---|
| __pycache__ | 5/5/2024 12:11 AM | File folder | |
| file.txt | 5/5/2024 12:28 AM | Text Document | 1 KB |
| Invoice_110_manish.txt | 5/5/2024 12:12 AM | Text Document | 1 KB |
| Invoice_115_manish.txt | 5/5/2024 12:11 AM | Text Document | 1 KB |
| main.py | 5/4/2024 11:50 PM | Python File | 4 KB |
| operation.py | 5/3/2024 3:02 PM | Python File | 1 KB |
| read.py | 5/4/2024 11:19 AM | Python File | 7 KB |
| RentData.txt | 5/5/2024 12:20 AM | Text Document | 1 KB |
| Return_Invoice_manish.txt | 5/5/2024 12:28 AM | Text Document | 1 KB |
| ReturnInvoice.txt | 5/3/2024 3:10 PM | Text Document | 1 KB |
| write.py | 5/3/2024 3:06 PM | Python File | 3 KB |

Figure 26 Table 5 File generated for returning land

Return_Invoice_manish.txt - Notepad

File Edit Format View Help

```
Name: manish
-------------------------------------
Address: inaruwa
-------------------------------------
Kitta Number: 115
-------------------------------------
Months Rented: 13
-------------------------------------
Initial Payment: 1014000
-------------------------------------
Delay Penalty: 10000
=====================================

Name: manish
-------------------------------------
Address: inaruwa
-------------------------------------
Kitta Number: 110
-------------------------------------
Months Rented: 65
-------------------------------------
Initial Payment: 2275000
-------------------------------------
Delay Penalty: 10000
=====================================
```

Figure 27 Generated return file opened

## 10.5 Test 5

| | |
|---|---|
| Objective | To show the update in stock of land |
| Action | Lands rented and then returned |
| Expected result | Text file should first make status not available after rented and again make available after returned |
| Actual result | Text file updated status not available after rented and again make available after returned |
| Conclusion | Test successful |

Table 5 test 5 to show the update in stock of land

### Evidence of Test 5

file.txt - Notepad

File Edit Format View Help

```
101, Hetauda, Central, 13, 45000, Available
102, Dhangadhi, West, 17, 70000, Available
103, Nepalgunj, West, 18, 75000, Not Available
104, Pokhara, West, 19, 85000, Available
105, Janakpur, East, 14, 46000, Available
106, Itahari, East, 20, 53000, Not Available
107, Bhairahawa, South, 11, 48000, Available
108, Bharatpur, Central, 13, 65000, Not Available
109, Tikapur, West, 15, 62000, Available
110, Inaruwa, East, 14, 35000, Not Available
111, Birgunj, South, 11, 78000, Not Available
112, Tansen       , West    , 16, 60000, Not Available
113, Dharan       , East, 15, 48000, Available
114, Siraha       , East, 12, 50000, Available
115, Butwal       , West, 18, 78000, Not Available
```

Figure 28 Status become not available after renting

file.txt - Notepad

File Edit Format View Help

```
101, Hetauda, Central, 13, 45000, Available
102, Dhangadhi, West, 17, 70000, Available
103, Nepalgunj, West, 18, 75000, Not Available
104, Pokhara, West, 19, 85000, Available
105, Janakpur, East, 14, 46000, Available
106, Itahari, East, 20, 53000, Not Available
107, Bhairahawa, South, 11, 48000, Available
108, Bharatpur, Central, 13, 65000, Not Available
109, Tikapur, West, 15, 62000, Available
110, Inaruwa, East, 14, 35000, Available
111, Birgunj, South, 11, 78000, Not Available
112, Tansen       , West    , 16, 60000, Not Available
113, Dharan       , East, 15, 48000, Available
114, Siraha       , East, 12, 50000, Available
115, Butwal       , West, 18, 78000, Available
```

Figure 29 Status become Available after returning

## 11. Conclusion

After finishing the Fundamentals of Computing program, I have acquired a lot of information and abilities that have really improved my comprehension of programming in python. I had the chance to develop a thorough Python program just for the coursework because of this experience. This program facilitates the renting and returning of lands, keeps track of lands in a text file, and creates thorough invoices for every transaction. Along with improving my understanding of Python programming, the creation of this program gave me a firm base on which to build.

I improved my ability to solve problems by identifying and fixing issues in the application by participating in the program's curriculum. In addition, I have mastered the use of Python's several flexible data structures, like dictionaries and lists, which I anticipate will be crucial for my upcoming projects. My perspective on programming's function across various sectors and industries has been widened by this practical grasp of its real-world applications. My desire to explore more complex programming approaches has risen, driving me to pursue more proficiency in the field of computing.

# 12. Appendix

## 12.1 Appendix of main.py

```
import read

import write

import operation

while True:

    print("\t-----------------------------------------------------------------\t")

    print("\t-------------------Welcome to Techno Property Nepal-------------------\t")

    print("\t-----------------------------------------------------------------\t")

    while True:

        print("Select operation:")

        print("1. Choose 1 to Show all the data")

        print("2. Choose 2 to Rent a Land")

        print("3. Choose 3 to Return a Land")

        print("4. Choose 4 to Exit")

        # Using exception handling to ensure valid input

        try:

            choice = int(input("Enter choice (1-4): "))

        except ValueError:

            print("Invalid input. Please enter a number from1 to 4.")

            continue

        if choice == 1:

            read.show_data()
```

```
    elif choice == 2:
        read.available_land()
        print("\t-----------------------------------------------------------------\t")
        print("\t-------------------Please provide us your details---------------------\t")
        print("\t-----------------------------------------------------------------\t")
        name = input("Enter your Name: ")
        address = input("Enter address: ")
        kitta_no = input("Enter Kitta Number that you want to rent: ")
        with open("file.txt", "r") as f:


            for line in f:
                record = line.strip().split(", ")
                if record[0] == kitta_no:
                    record[-1]="Not Available"
                    print("Invalid Kitta no")
                    break
                else:
                    months = input("Enter numbers of months you want to rent: ")
                    write.rent_Info(kitta_no, name, address, months)
                    read.take_data(name, address, kitta_no, months)
                    operation.update_status(kitta_no)
                    write.write_record(name, address, kitta_no, months)
```

```
                break  # Break out of the inner loop to return to the main menu
        elif choice == 3:
            print("\t-----------------------------------------------------------------\t")
            print("\t-------------------Please provide us your details---------------------\t")
            print("\t-----------------------------------------------------------------\t")
            name = input("Enter your Name: ")
            address = input("Enter address: ")
            kitta_no=input("Enter Kitta Number that you want to return: ")
            delay_month=input("Enter number of months you were  delay: ")
            read.return_info(name, address, delay_month)
            operation.return_status(kitta_no)
            break  # Break out of the inner loop to return to the main menu
        elif choice == 4:
            print('Thank you for renting from Techno Property Nepal')
            exit()  # Exit the program
        else:
            print('Invalid Option')
            continue

        # Ask user if they want to perform more operations
        while True:
```

```
        print("Do you want to perform more operation: ")
        print("1. Yes")
        print("2. No")
        try:
            choice = input("Enter choice (1-2): ")
        except ValueError:
            print("Invalide choice please enter number 1 or 2")
        if choice == "1":
            continue_status=True
            break
        elif choice == "2":
            print("Thank you for renting from Techno Property Nepal")
            continue_status = False
            break
        else:
            print("Invalid choice")
            choice = input("Enter choice (1-2): ")
    if not continue_status:
        break
```

## 12.2 Appendix of Operation.py

```
def update_status(Kitta_no):
    updated_records = []
    with open("C:\\Users\\User\\Downloads\\Python CourseWork\\file.txt", "r") as f:
        for line in f:
            record = line.strip().split(", ")
            if record[0] == Kitta_no:
                record[-1] = "Not Available"
            updated_records.append(", ".join(record))

    with open("file.txt", "w") as f:
        for record in updated_records:
            f.write(record + "\n")


def return_status(Kitta_no):
    update_records = []
    with open("file.txt", "r") as f:

        for line in f:
            record = line.strip().split(", ")
            if record[0] == Kitta_no:
                record[-1]="Available"
```

```
        update_records.append(", ".join(record))


with open("file.txt", "w") as f:
    for record in update_records:
        f.write(record + "\n")
```

## 12.3 Appendix of read.py

```
import datetime



def take_data(Name,Address,Kitta_number,Months):
    with open("file.txt","r")as f:
        container=f.readlines()
        FoundStatus=False
        for contents in container:
            content=contents.strip().split(",")
            if content[0]==Kitta_number:
                content[4]=int(content[4])*int(Months)
                # Creating invoice
                content[5]='Rented'
                # Create invoice content
                invoice_content = (
                    f'Invoice Date: {datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")}\n'
                    f'-------------------------------------\n'
                    f'Customer Name: {Name}\n'
                    f'-------------------------------------\n'
                    f'Customer Address: {Address}\n'
                    f'-------------------------------------\n'
```

```
        f'Kitta Number: {content[0]}\n'
        f'-------------------------------------\n'
        f'Place: {content[1]}\n'
        f'-------------------------------------\n'
        f'Direction: {content[2]}\n'
        f'-------------------------------------\n'
        f'Anna: {content[3]}\n'
        f'-------------------------------------\n'
        f'Price: {content[4]}\n'
        f'-------------------------------------\n'
        f'Status: {content[5]}\n'
        f'-------------------------------------\n'
        f'Months Rented: {Months}\n'
        f'=====================================\n'
    )

    # Displaying invoice in terminal
    print("=====================================\n")
    print("Rent Invoice:\n")
    print("=====================================\n")
    print(invoice_content)
```

```
            FoundStatus=True
            break
        if not FoundStatus:
            print("Entered Kitta Number is not found in the system")


#Function to show recorded data
def show_data():

print("==================================================================
==================================================================
=========================")
    print("|\tKitta Number\t|\tcity/district\t\t|\tDirection\t|\tAnna\t |  \tPrice\t       |\tAvailability\t
|")

print("==================================================================
==================================================================
=========================")
    with open("file.txt", "r") as f:
        contents = f.readlines()
        for content in contents:
            cont = content.strip().split(",")


            print(f"|\t {cont[0]}            \t|\t {cont[1]}        \t| {cont[2]}              \t|\t {cont[3]}    \t |\t
{cont[4]}   \t|\t {cont[5]}      \t|")
```

```
        print("-----------------------------------------------------------------------------------------------------------------------------------------------------------")


#Function to show available land data

def available_land():


print("==================================================================================================================================================================")

    print("|\tKitta Number\t|\tcity/district\t\t|\tDirection\t|\tAnna\t |  \tPrice\t       |\tAvailability\t |")


print("==================================================================================================================================================================")

    with open("C:\\Users\\User\\Downloads\\Python CourseWork\\file.txt", "r") as f:

        contents = f.readlines()

        for content in contents:

            cont = content.strip().split(",")

            if cont[5].strip().lower() == "available":

                print(f"|\t {cont[0]}          \t|\t {cont[1]}       \t| {cont[2]}             \t|\t {cont[3]}    \t |\t {cont[4]}   \t|\t {cont[5]}      \t|")

            #  print("-----------------------------------------------------------------------------------------------------------------------------------------------------------")

def return_info(Name,Address,Delaymonth):
```

```
intDelaymonth = int(Delaymonth)  # Ensure Delaymonth is an integer
TotalPayment = 0
invoice_contents = []  # This will store each line of the invoice

with open("RentData.txt", "r") as f:
    records = f.readlines()

for record in records:
    content = record.strip().split(",")
    if content[0] == Name and content[1] == Address:
        # Calculate payment including the delay penalty
        payment_due = int(content[3]) + 5000 * intDelaymonth
        TotalPayment += payment_due
        # Prepare the invoice details
        invoice_line = (
            f'Name: {content[0]}\n'
            f'-------------------------------------\n'
            f'Address: {content[1]}\n'
            f'-------------------------------------\n'
            f'Kitta Number: {content[2]}\n'
            f'-------------------------------------\n'
            f'Months Rented: {content[4]}\n'
```

```
            f'-------------------------------------\n'
            f'Initial Payment: {content[3]}\n'
            f'-------------------------------------\n'
            f'Delay Penalty: {5000 * intDelaymonth}\n'
            f'=====================================\n'
            # f'Total Payment Due: {payment_due}\n'
        )
        invoice_contents.append(invoice_line)

    # Check if there are any invoices to process
    if invoice_contents:
        # Write the invoice details to a file
        with open(f'Return_Invoice_{Name}.txt', "a") as f:
            for line in invoice_contents:
                f.write(line + '\n')

        # Display the invoice in the terminal
        print("=====================================\n")
        print("Return Invoice: \n")
        print("=====================================\n")
        for line in invoice_contents:
            print(line)
```

```
        print('-------------------------------------------------------------------------------------------------
--')
        print("Total Payment: ", TotalPayment)
        print('-------------------------------------------------------------------------------------------------
--')

    else:
        print("No such records found, please try again!")
```

## 12.4 Appendix of write.py

```
import datetime


def write_record(Name, Address, Kitta_no, Months):
    with open('file.txt', 'r') as f:
        container = f.readlines()
        found_status = False
        for contents in container:
            content = contents.strip().split(',')
            if content[0] == Kitta_no:
                content[4] = int(content[4]) * int(Months)
                # Creating invoice content
                invoice_content = (
                    f'-------------------------------------\n'
                    f'Invoice Date: {datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")}\n'
                    f'-------------------------------------\n'
                    f'Customer Name: {Name}\n'
                    f'-------------------------------------\n'
                    f'Customer Address: {Address}\n'
                    f'-------------------------------------\n'
                    f'Kitta Number: {content[0]}\n'
                    f'-------------------------------------\n'
```

```
        f'City/District: {content[1]}\n'
        f'-------------------------------------\n'
        f'Direction: {content[2]}\n'
        f'-------------------------------------\n'
        f'Anna: {content[3]}\n'
        f'-------------------------------------\n'
        f'Price: {content[4]}\n'
        f'-------------------------------------\n'
        f'Status: {content[5]}\n'
        f'-------------------------------------\n'
        f'Months Rented: {Months}\n'
        f'=====================================\n'
    )




    # Writing invoice to a file
    invoice_filename = f'Invoice_{Kitta_no}_{Name.replace(" ", "_")}.txt'
    with open(invoice_filename, 'w') as invoice_file:
        invoice_file.write(invoice_content)
    found_status = True
    break
```

```
        if not found_status:
            print("Kitta Number is not found, please enter valid one")


def rent_Info(Kitta_no, Name, Address, Months):
    with open("C:\\Users\\User\\Downloads\\Python CourseWork\\file.txt", "r") as f:
        rent = f.readlines()


    with open('RentData.txt', 'a') as f:  # Open file in write mode ('w') outside the loop
        for contents in rent:
            content = contents.strip().split(",")
            if content[0] == Kitta_no:
                content[4]= int(content[4])*int(Months)
                f.write(f'{Name},{Address},{Kitta_no},{content[4]},{Months}\n')  # Write data for
each entry on a new line
                break
```

## 13. References

Course sidekick, 2024. *computer-science/4132751.* [Online]
Available at: https://www.coursesidekick.com/computer-science/4132751
[Accessed 02 may 2024].

geeksforgeeks, 2023. *what-is-pseudocode-a-complete-tutorial.* [Online]
Available at: https://www.geeksforgeeks.org/what-is-pseudocode-a-complete-tutorial/
[Accessed 02 may 2024].

jaro education, 2024. *commonly-asked-python-realted-interview-qna.* [Online]
Available at: https://jaroeducation.com/blog/commonly-asked-python-realted-interview-qna/
[Accessed 01 may 2024].

java T point, 2021. *pseudocode-java.* [Online]
Available at: https://www.javatpoint.com/pseudocode-java

w3 school, 2024. *dsa_intro.php.* [Online]
Available at: https://www.w3schools.com/dsa/dsa_intro.php
[Accessed 01 may 2024].

## 14. Originality test report

5/5/24, 2:48 PM 23049274 Manish Kumar Chaudhary

# Originality report

COURSE NAME
CC4051 Fundamentals of Computing

STUDENT NAME
MANISH KUMAR CHAUDHARY

FILE NAME
23049274 Manish Kumar Chaudhary

REPORT CREATED
May 5, 2024

### Summary

| | | |
|---|---|---|
| Flagged passages | 2 | 1% |
| Cited/quoted passages | 6 | 2% |

Web matches

| | | |
|---|---|---|
| jaroeducation.com | 2 | 2% |
| geeksforgeeks.org | 1 | 0.5% |
| medium.com | 1 | 0.3% |
| softwaretestinghelp.com | 1 | 0.2% |
| nonstopio.com | 1 | 0.2% |
| onecompiler.com | 1 | 0.2% |

Figure 30 Originality repot