



CS4051NI Fundamentals of Computing

60% Individual Coursework

2023/24 Spring

Student Name: Raman Nath

London Met ID: 22085823

College ID: np01cp4s230147

Assignment Due Date: Tuesday, May 7, 2024

Assignment Submission Date: Tuesday, May 7, 2024

Word Count: 8653

I confirm that I understand my coursework needs to be submitted online via MySecondTeacher under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Table of Contents

1.Introduction	1
1.1 Goals and objectives	2
2.Tools used in coursework.....	3
2.1 Integrated Development and Learning Environment (IDLE)	3
2.3 Draw.io	5
3.Algorithm	6
3.1 The algorithm steps used in the program are as follows:.....	6
4.Flowchart.....	8
4.1 Display	9
4.2 Rent Flowchart.....	10
4.3 Return Flowchart	12
5.Pseudocode	14
5.1 main.py	14
5.2 function.py	16
5.3 display.py.....	24
5.4 operation.py	25
5.5 readLand.py.....	26
5.6 writeLand.py	27
6. Data structure	28
6.1 Primitive data structure	28
6.2 Non-Primitive data structures.....	31
7. Program	33
7.1 Explain the process of how the program operates and functions.	34
7.2 Overview of Land Rental System.....	35
7.3 Display Available Lands.....	36
7.4 Renting process.....	37
7.5 Returning process.....	40
7.6 Creation of a Txt file.....	42
7.7 Exiting the Program.	44
8. Testing	45
8.1 Test 1: Show implementation of try, except	46
8.2 Test 2: Selection rent and return of lands	47

8.3 Test 3: File generation of renting of land(s) (Renting multiple land(s))	49
8.4 Test 4: File generation of returning process of land(s) (Returning multiple land(s))	54
8.5 Test 5 Show the update in stock of land(s)	58
9. Conclusion	61
10. References	62
11. Appendix	63
11.1 Land.txt	63
11.2 Main.py	63
11.3 function.py	65
11.4 display.py	79
11.5 operation.py	82
11.6 read land.py	82
11.7 writeLand.py	83

Table of Figure

Figure 1 Integrated Development and Learning Environment (IDLE).....	3
Figure 2 Microsoft word.....	4
Figure 3 Draw.io.....	5
Figure 4 Display Flowchart.....	9
Figure 5 Rent Flowchart.....	10
Figure 6 Return Flowchart.....	12
Figure 7 Integer.....	29
Figure 8 Boolean.....	30
Figure 9 String.....	30
Figure 10 List.....	33
Figure 11 Overview of land rental system.....	35
Figure 12 Displaying available lands.....	36
Figure 13 Renting Land process.....	38
Figure 14 Renting Land process continue.....	39
Figure 15 Returning Land process.....	40
Figure 16 Returning Land Continue.....	41
Figure 17 Creation of rent txt file.....	43
Figure 18 Creation of return txt file.....	44
Figure 19 Creation of txt file.....	45
Figure 20 Provide invalid data a letter instead of a number.....	46
Figure 21 Provide negative value as input.....	47
Figure 22 Provide non existed value as input.....	48
Figure 23 File generation of renting of lands renting multiple.....	49
Figure 24 Full Renting process.....	50
Figure 25 Renting process continue.....	50
Figure 26 Showing output in cell as well.....	52
Figure 27 Finally show the rented lands details in a text file.....	53
Figure 28 Finally showing the rented lands in txt file.....	53
Figure 29 Show the complete returning process of the land(s) continue.....	55
Figure 30 Returning Multiple Lands File Generation.....	57
Figure 31 Finally show the returned land(s) details in text file.....	58
Figure 32 Data.txt.....	58
Figure 33 Update Stock of Lands Availability.....	59
Figure 34 Update Stock of Lands Availability.....	60

Table of Table

Table 1 Flowchart icon representation	8
Table 2 Provide invalid input and show message	46
Table 3 Provide the negative value as input.....	47
Table 4 Provide the non-existed value as input.....	48
Table 5 Show output in cell as well	51
Table 6 Show the complete returning process of the land(s).	54
Table 7 Show output in the shell as well.	56
Table 8 Finally show the returned land(s) details in text file.	57
Table 9 Update Stock of Lands Availability	59
Table 10 Update Stock of Lands Availability	60

1.Introduction

Techno Property Nepal is a leading private company in Nepal and offers a varied portfolio of land for rent across various locations in the country. With a commitment to excellence, the company facilitates smooth transactions for its clients, ensuring efficient utilization of available resources. This report explores the creation of a Python program customized specifically for Techno Property Nepal, with the goal of transforming the management of land rental transactions. Our focus is on developing a user-friendly computer program that empowers Techno Property Nepal to effortlessly manage the availability and rental of land holdings. By automating these processes, the program not only simplifies daily operations but also ensures a smooth experience for both the company and its customers. The programs main functions include reading data from a text file with information about available land, displaying the status of each property and updating it as rentals occur. It also generates invoices for each transaction and showing details like land ID, city, directions, rental duration, customer information, dates and total amount. Additionally, the program will include functions to manage situations like late returns by customers, applying monthly fines according to the contract terms. This ensures compliance with agreements and helps prevent potential revenue loss for Techno Property Nepal.

This report explores the details of the program, explaining how it benefits Techno Property Nepal and its clients in a clear and easy to understand way. By combining Pythons features with real estate management our goal is to improve efficiency in the industry, making the rental process smoother and more transparent. our Python project is a big step in our school journey and shows how technology can change old industries like real estate. By doing good research, working carefully, and using our program well, we've made something strong that makes things better and clearer, and helps real estate management move forward with technology.

1.1 Goals and objectives

Goals

Our main goal is to create a user-friendly Python program specifically designed for Techno Property Nepal. This project holds a lot of importance in our academic journey, making up a big part of our coursework grade and showing how well we know Python programming. By using Python's ability to adjust, we aim to update how Techno Property Nepal manages land rental transactions, making things work better and making customers happier. Through careful research, planning and doing things right, our goal is to make a strong software solution that lets Techno Property Nepal easily handle land rentals across different places in the country. Also we want to show how useful it is to use technology in traditional industries like real estate, making things run smoother and better overall. We want to make a real difference by helping improve how real estate is managed using new technology.

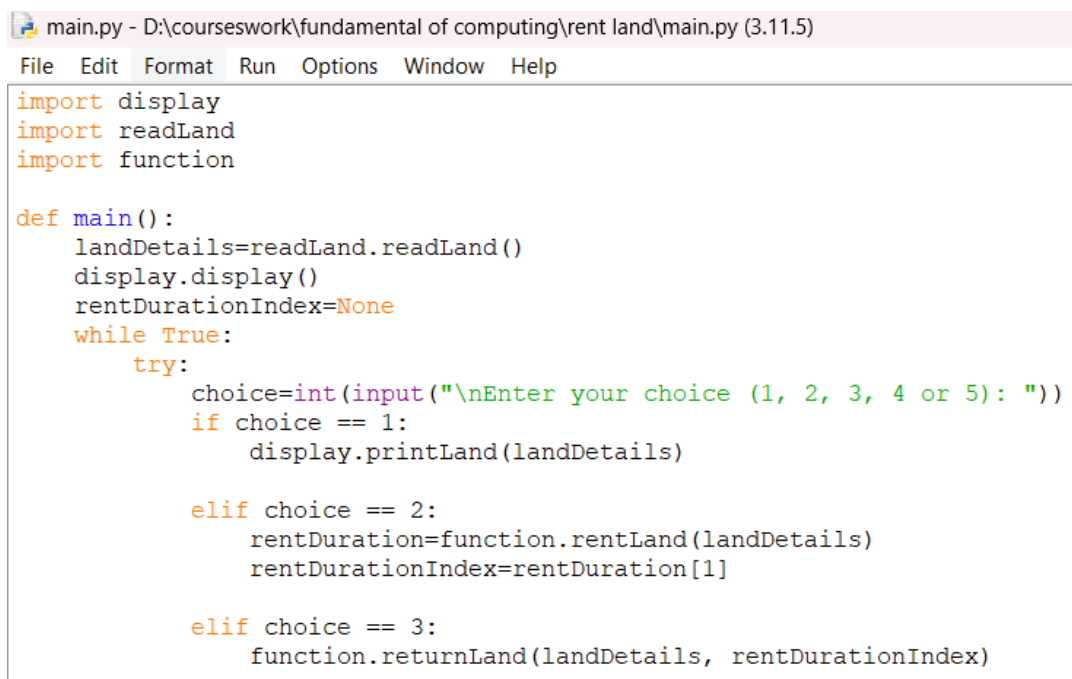
Objectives

- To Create a Python software to manage land rentals effectively.
- Create tools to manage land information and update rental details.
- Automate renting and returning land and create invoices automatically.
- Create a user-friendly interface for easy use
- Make sure the program can grow and change with Techno Property Nepal's needs.
- Add features to manage land information and update rental details.
- Show how using Python can make real estate management better.
- Use the right techniques and structures in Python coding.
- Test the program a lot to make sure it works well.
- Follow rules about plagiarism and write code ethically.
- Explain how the program works in simple terms.
- Format the report nicely and follow all the rules for submission.

2.Tools used in coursework

2.1 Integrated Development and Learning Environment (IDLE)

IDLE stand for “Integrated Development and Learning Environment”. It is a comprehensive software tool designed to facilitate Python programming. This versatile program serves as an IDE (Integrated Development Environment) it is specifically made for Python programming with combining various essential components to enhance the Python coding experience. (TutorialsTeacher.com. , n.d.) With the help of this we will find a capable code editor where we can easily write, edit something and save Python code and complete with features like automatic indentation and syntax highlighting to ensure clean and error free code. Its interactive Python shell stands out as a valuable asset allowing you to execute Python commands and immediately see the results, making it ideal for experimentation and testing code snippets and interactive learning.



```
main.py - D:\coursework\fundamental of computing\rent land\main.py (3.11.5)
File Edit Format Run Options Window Help

import display
import readLand
import function

def main():
    landDetails=readLand.readLand()
    display.display()
    rentDurationIndex=None
    while True:
        try:
            choice=int(input("\nEnter your choice (1, 2, 3, 4 or 5): "))
            if choice == 1:
                display.printLand(landDetails)

            elif choice == 2:
                rentDuration=function.rentLand(landDetails)
                rentDurationIndex=rentDuration[1]

            elif choice == 3:
                function.returnLand(landDetails, rentDurationIndex)
```

Figure 1 Integrated Development and Learning Environment (IDLE)

2.2 Ms Word

Microsoft Word is a computer program that helps you create and edit different types of documents. It is like a digital typewriter that serve to type and format text with too long text we can add any types of pictures and make your documents look good. We can use it for writing letters in different language, making school/college reports or creating all sorts of documents. (A-143, 9th Floor, Sovereign Corporate Tower, Sector-136,, n.d.) It also has tools to check your spelling which make report correct and nice which can be really helpful. Think of it as like a handy tool for writing and designing ant types of documents on our computer.

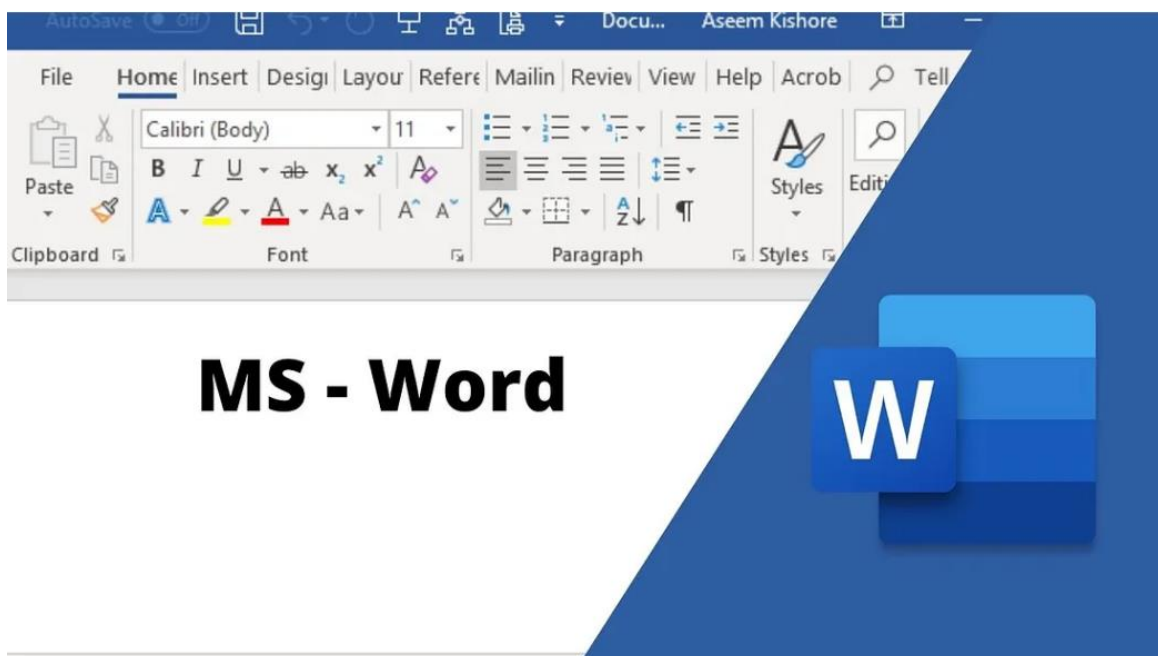


Figure 2 Microsoft word

2.3 Draw.io

It can work on diagrams simultaneously and save them in various formats including image files or cloud storage services. Overall Draw.io is a handy tool for visually illustrating concepts and ideas. Draw.io is a web-based diagramming tool that help users to create easily different types of diagrams and flowcharts. It provides a user-friendly interface with a drag representation of processes or structures. Draw.io is often used for creating organizational charts, flowcharts, mind maps and more. (JGraph Ltd, Artisans' House, n.d.).It is popular for its versatility, accessibility and collaboration features as multiple users and drop functionality which making it simple to add shapes, arrows and text.

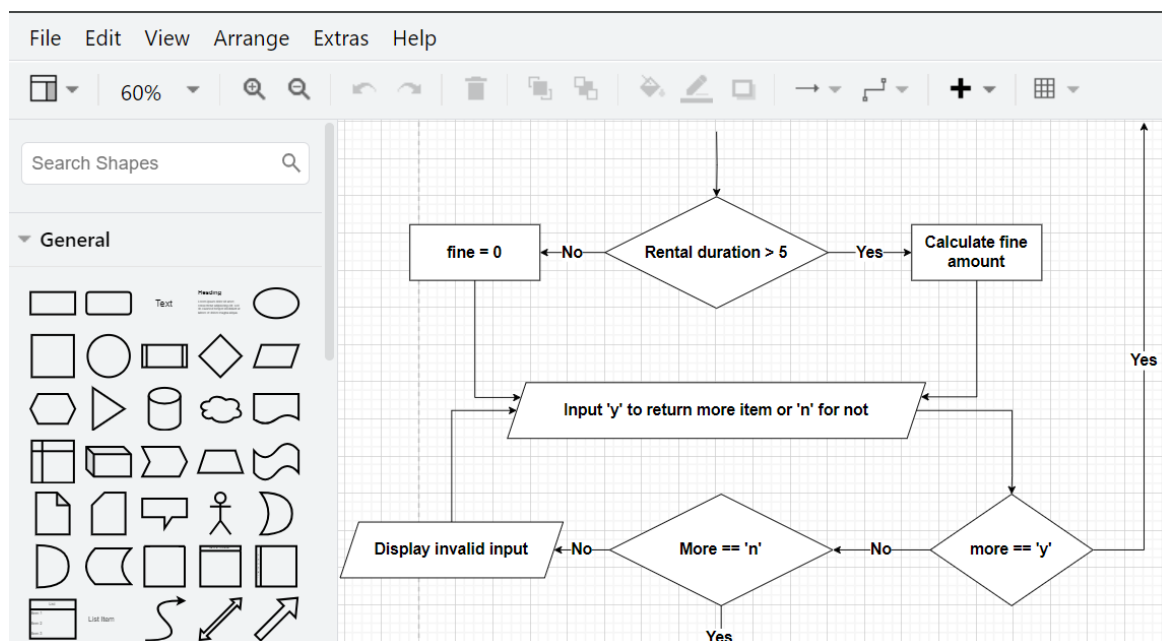


Figure 3 Draw.io

3.Algorithm

3.1 The algorithm steps used in the program are as follows:

Step 1: Start

Step 2: Read land data from “land.txt” file.

Step 3: Show the main menu options.

Step 4: Enter the user choice.

Step 5: If the user selects option 1:

Step 6: Display Available Land for Renting.

Step 7: If the user chooses option 2 (Rent Land):

Step 8: Enter customer information.

Step 9: Show available land for rent.

Step 10: Enter Kitta Number of the land to rent.

Step 11: If the Kitta Number is invalid, display an error message and return to Step 10.

Step 12: Input rental duration in months.

Step 13: If the duration is invalid, show an error message and go back to Step 12.

Step 14: Ask if the user wants to rent more land. If ‘y’ for yes, go to Step 10. If ‘n’ for no, continue.

Step 15: Update data file to show rented land as unavailable.

Step 16: Show rental invoice to customer with land rental information.

Step 17: Save customer’s rental invoice to a text file.

Step18: Confirm successful rental transaction and offer additional guidance if necessary.

Step 19: If the user selects option 3 (Return Land):

Step 20: Enter customer information for land return.

Step 21: Show rented lands available for return.

Step 22: Enter Kitta Number of the land to return.

Step 23: If the kitta number is invalid, display an error message and return to Step 22.

Step 24: Enter the duration of the return in months.

Step 25: If the duration is invalid, show an error message and return to Step 24.

Step 26: Calculate the total late fine (if any) for the return.

Step 27: Update file to indicate returned land as available.

Step 28: Show return invoice to the customer with details of the returned land and any fines.

Step 29: Save customer's return invoice to a text file.

Step 30: Confirm successful return transaction and offer additional instructions if necessary.

Step 31: If the user selects option 4 (View Comments/Feedback):

Step 32: Show any comments or feedback from customers.

Step 33: Let the user input new comments or feedback.

Step 34: Update comments/feedback file with the new input.

Step 35: If the user chooses option 5 (Exit):

Step 36: End the program with a suitable thank you message to the user.

Step 37: Else (invalid choice):

Step 38: Print invalid choice.

Step 39: End.

4.Flowchart

A flowchart is like a visual map that helps us to understand and explain processes or any systems. It uses simple shapes like rectangles, circles and arrows to represent different steps in a sequence. Each shape represents a specific task or decision and arrows connect them to show the order in which they happen. Flowcharts are great for breaking down complex tasks into simple easier to understand parts. They are used in various fields like programming, business and engineering to make plan and analyse system and making it easier to spot problems or inefficiencies. flowchart is a handy tool for visualizing and organizing steps in a process making it easier for people to follow and understand. Some symbol of flowchart which are used in my coursework are given describe.

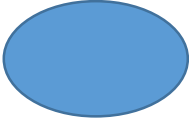



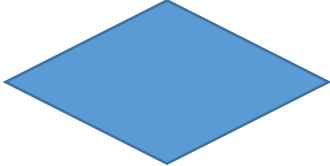
Symbol	Name	Function
	Start/End	An oval represent a start or end point.
	Arrows	A arrow is a connector that shows relationship between the representative shapes.
	Input/output	A parallelogram represent the input and output.
	Process	A rectangle represent the process.
	Decision	A diamond indicates a decision.

Table 1 Flowchart icon representation

4.1 Display

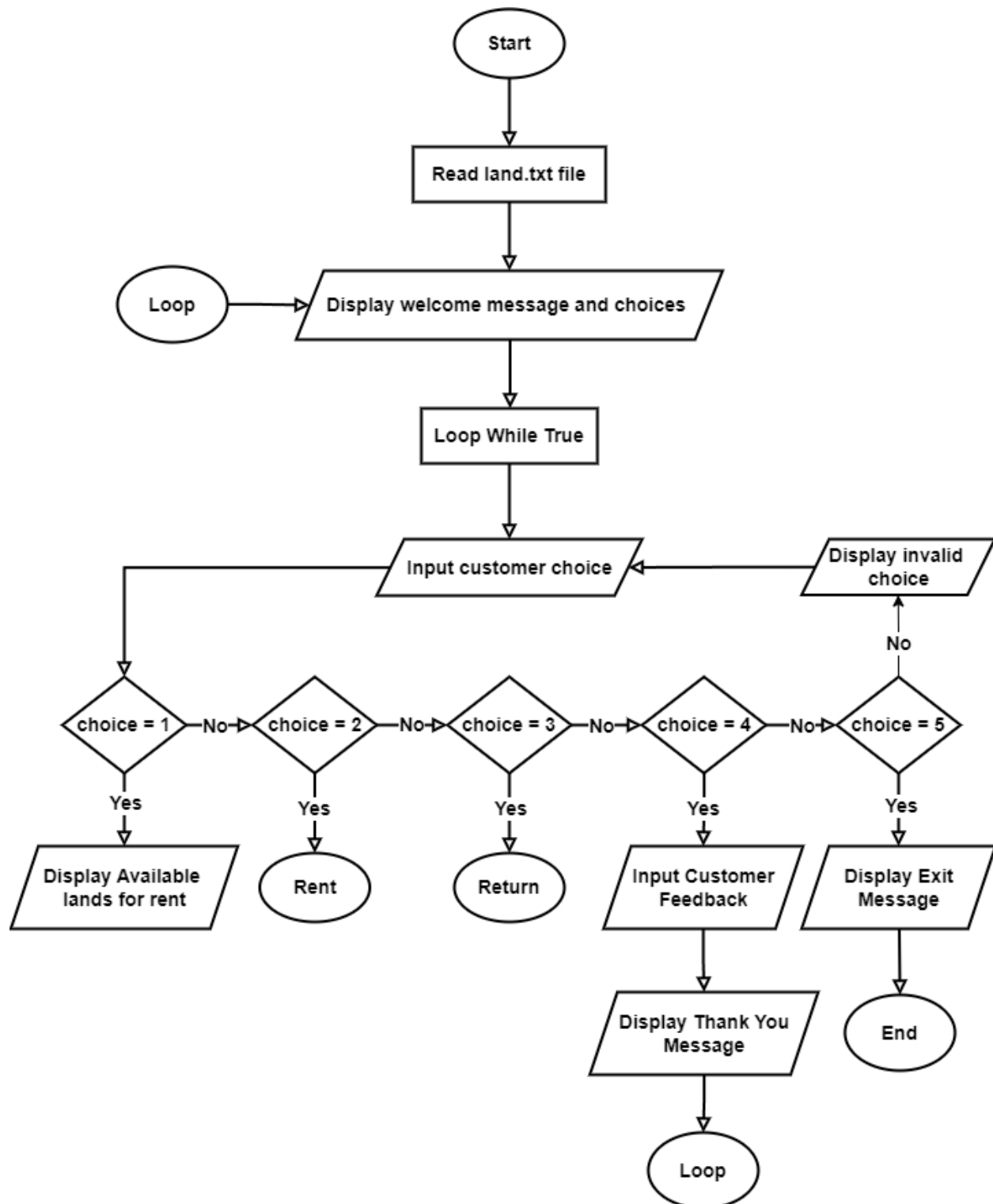


Figure 4 Display Flowchart

4.2 Rent Flowchart

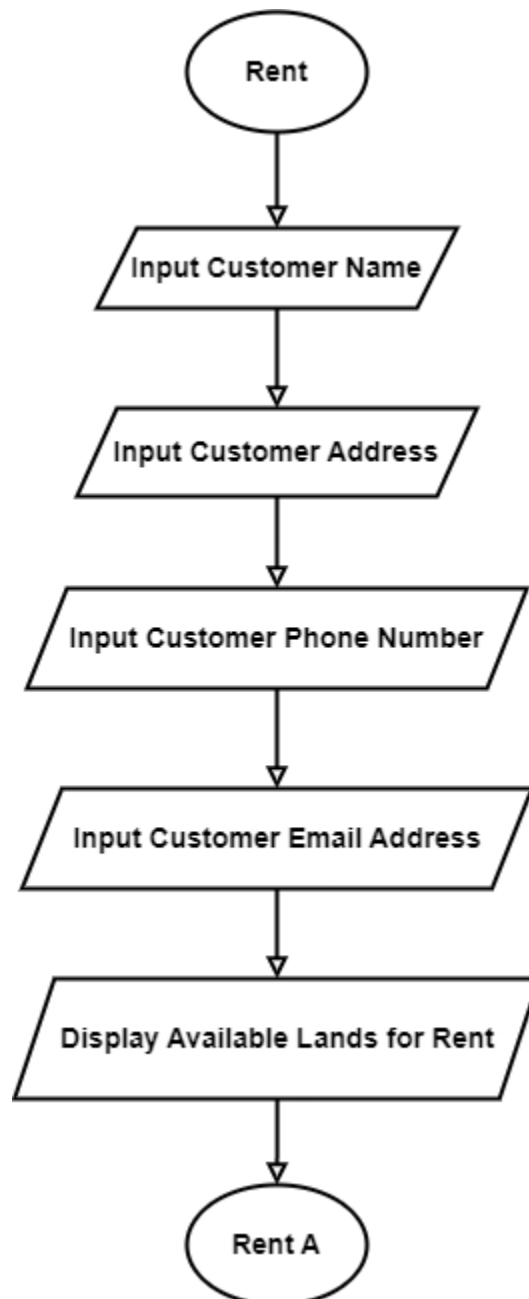
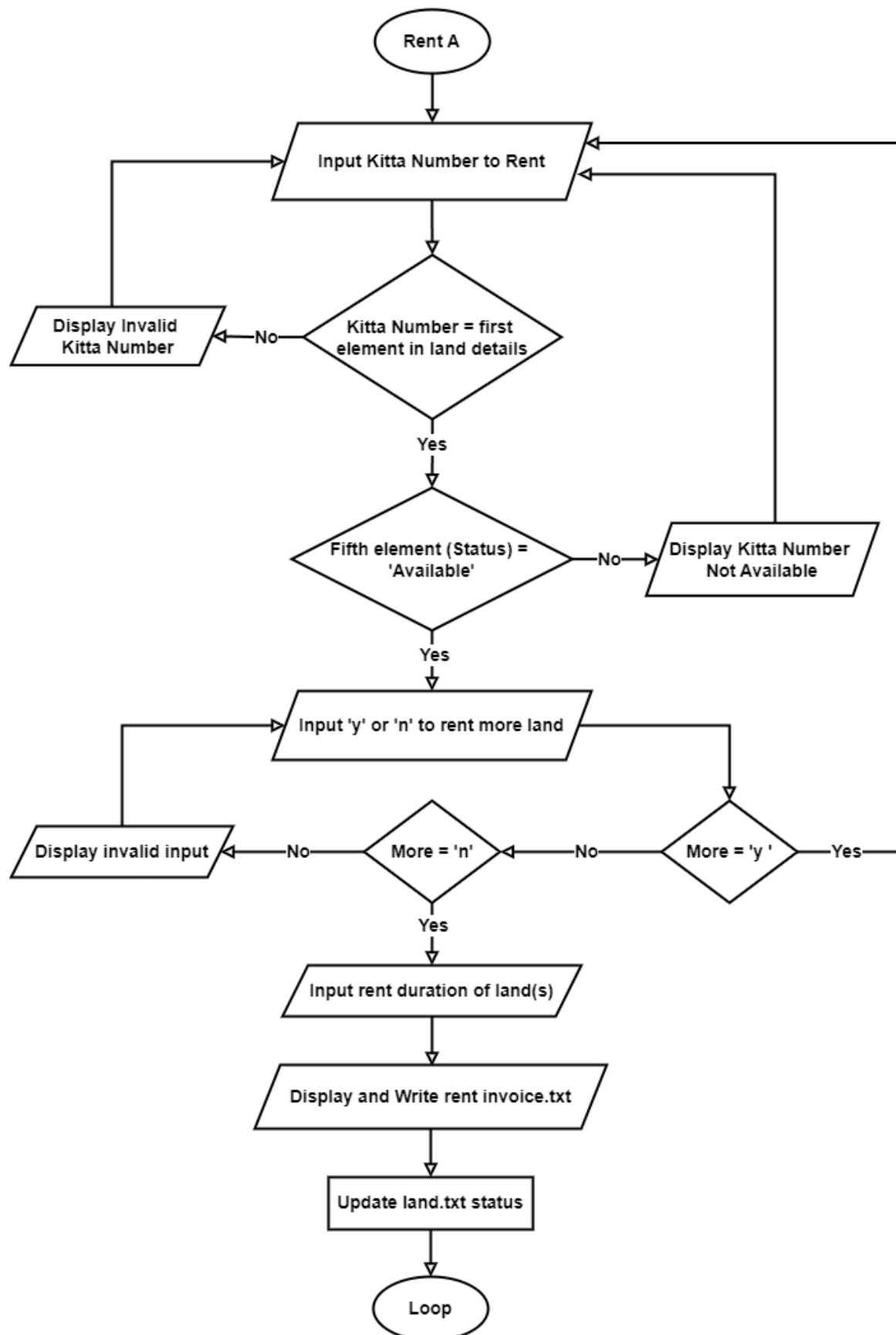


Figure 5 Rent Flowchart



4.3 Return Flowchart

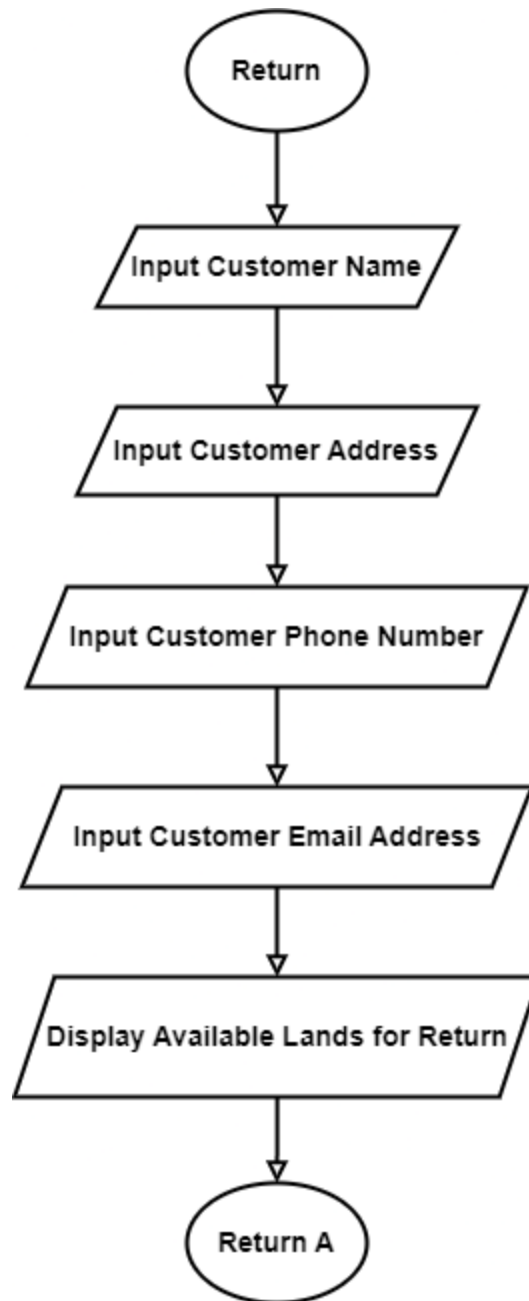
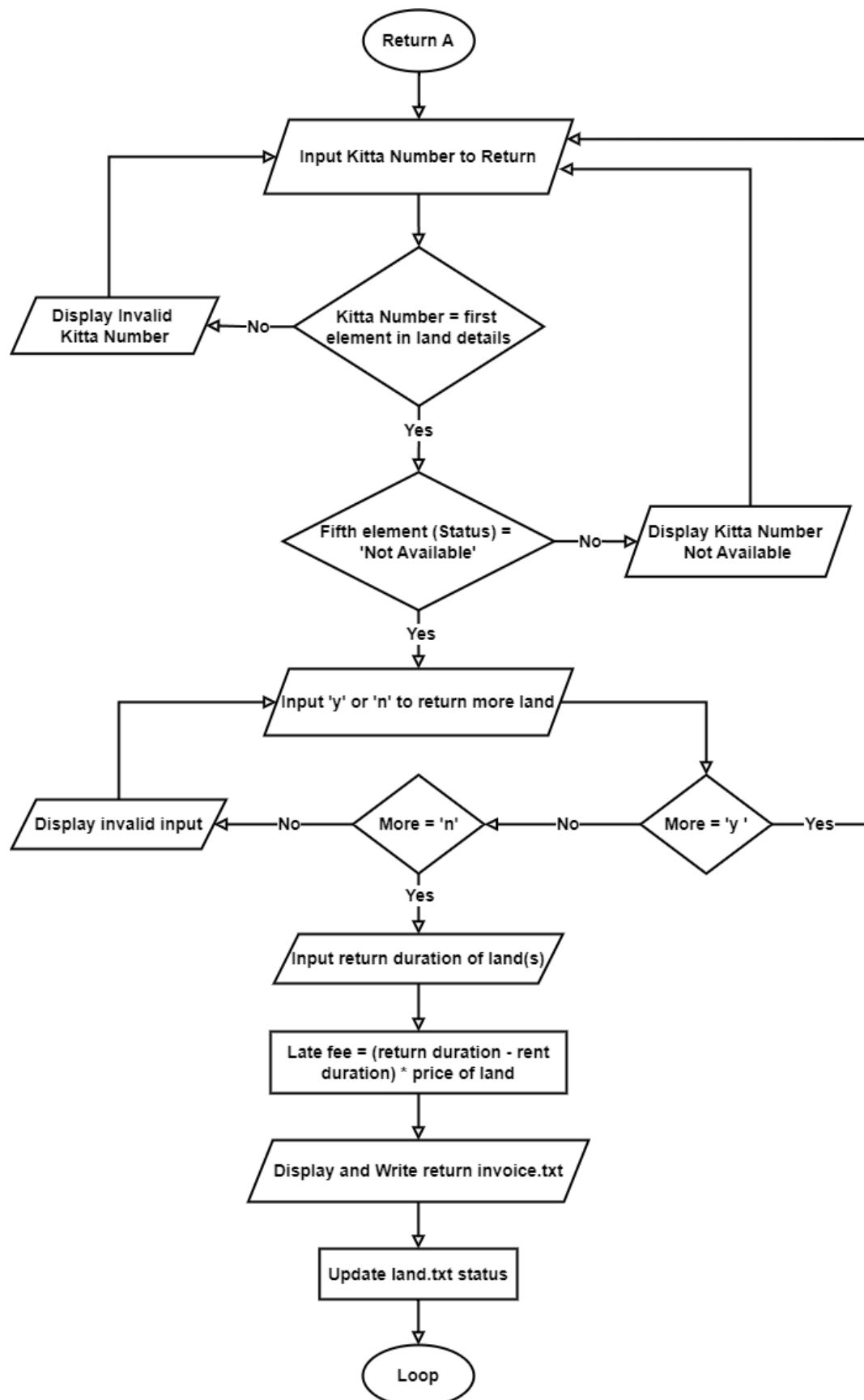


Figure 6 Return Flowchart



5.Pseudocode

Pseudocode serves as a plan or outline for a computer program, providing a basic overview of its functionality. The programming language is presented in a straightforward and understandable manner, making it easy for anyone to comprehend without confusion. It provides a thorough explanation of the solution's steps and facilitating understanding for programmers and other individuals.

5.1 main.py

IMPORT display

IMPORT readLand

IMPORT function

CREATE a function named main

SET landDetails by calling the function readLand from readLand module

CALL display function from display module

SET rentDurationIndex to None

WHILE True

TRY

INPUT choice as an integer

IF choice is equal to 1 **THEN**

CALL printLand function from display module with landDetails as parameter

ELSE IF choice is equal to 2 **THEN**

SET rentDuration by calling the function rentLand from function module with landDetails as parameter

SET rentDurationIndex to the second element of rentDuration

ELSE IF choice is equal to 3 **THEN**

CALL returnLand function from function module with landDetails and
rentDurationIndex as parameters

ELSE IF choice is equal to 4 **THEN**

INPUT provide us your feedback message

PRINT thank you message

ELSE IF choice is equal to 5 **THEN**

PRINT thank you message

BREAK out of the loop

ELSE

Print please enter your choice message

EXCEPT ValueError

PRINT invalid choice message

END TRY

END WHILE

END the main function

CALL the main function

5.2 function.py

```
IMPORT display
IMPORT writeLand
IMPORT datetime
IMPORT operation
IMPORT readLand
```

SET currentDate to the current date and time

SET landDetails by calling the function readLand from readLand module

CREATE a function named generateRentInvoice with parameters customer, address, phoneNumber, rentLand, rentDuration, and emailAddress

INITIALIZE invoice string with the invoice header and customer details

INITIALIZE totalAmount to 0

FOR EACH rowOfLand IN rentLand

CALCULATE price by multiplying the land price with rentDuration

ADD price to totalAmount

APPEND land details to invoice

END FOR

APPEND totalAmount and footer to invoice

RETURN invoice

END generateRentInvoice function

CREATE a function named rentLand with parameter landDetails

INITIALIZE rentLand list

PRINT rental information prompt

WHILE True

INPUT customer name

```
    IF customer name is empty THEN
        PRINT error message
    ELSE
        BREAK the loop
    END IF
END WHILE
WHILE True
    INPUT customer address
    IF customer address is empty THEN
        PRINT error message
    ELSE
        BREAK the loop
    END IF
END WHILE
WHILE True
    TRY
        INPUT phone number as an integer
        BREAK the loop
    EXCEPT ValueError
        PRINT error message
    END TRY
END WHILE
WHILE True
    INPUT email address
    IF email address is empty THEN
        PRINT error message
    ELSE
        BREAK the loop
```

END IF

END WHILE

CALL the function printLand from display module with landDetails as parameter

WHILE True

TRY

INPUT kitta number as an integer

IF kitta number is found in landDetails **THEN**

IF land is available for rent **THEN**

WHILE True

INPUT rent duration in months

IF rent duration is less than or equal to 0 **THEN**

PRINT error message

ELSE

BREAK the loop

END IF

END WHILE

CALL generateRentInvoice with customer, address, phoneNumber, rentLand, rentDuration, and emailAddress as parameters

PRINT rent invoice

WRITE rent invoice to a file

UPDATE land status to "Not Available"

WRITE updated land details to file

RETURN rentLand and rentDuration

ELSE

PRINT land not available message

END IF

ELSE

PRINT invalid kitta number message

END IF

EXCEPT ValueError

PRINT invalid kitta number message

END TRY

END WHILE

END rentLand function

CREATE a function named generateReturnInvoice with parameters customer, address, phoneNumber, returnLand, rentDuration, returnDuration, and emailAddress

INITIALIZE invoice string with the invoice header and customer details

INITIALIZE totalAmount and totalLateAmount to 0

INITIALIZE lateFine to 0

FOR EACH rowOfLand IN returnLand

CALCULATE lateFine based on returnDuration and rentDuration

ADD lateFine to totalLateAmount

CALCULATE totalPrice based on returnDuration and land price

ADD totalPrice to totalAmount

APPEND land details to invoice

END FOR

APPEND totalLateAmount, totalAmount, and footer to invoice

RETURN invoice

END generateReturnInvoice function

CREATE a function named returnLand with parameters landDetails and rentDuration

INITIALIZE returnLand list

PRINT return information prompt

WHILE True

INPUT customer name

IF customer name is empty **THEN**

PRINT error message

ELSE

BREAK the loop

END IF

END WHILE

WHILE True

INPUT customer address

```
    IF customer address is empty THEN
        PRINT error message
    ELSE
        BREAK the loop
    END IF
END WHILE
WHILE True
    TRY
        INPUT phone number as an integer
        BREAK the loop
    EXCEPT ValueError
        PRINT error message
    END TRY
END WHILE
WHILE True
    INPUT email address
    IF email address is empty THEN
        PRINT error message
    ELSE
        BREAK the loop
    END IF
END WHILE
CALL printLand function from display module with landDetails as parameter
WHILE True
    TRY
        INPUT kitta number as an integer
        IF kitta number is found in landDetails THEN
            IF land is not available for rent THEN
```

```
ADD land details to returnLand
WHILE True
    INPUT more lands to return
    IF user does not want to return more lands THEN
        WHILE True
            INPUT return duration in months
            IF return duration is less than or equal to 0 THEN
                PRINT error message
            ELSE
                BREAK the loop
            END IF
        END WHILE
        CALL generateReturnInvoice with customer, address,
        phoneNumber, returnLand, rentDuration, returnDuration, and
        emailAddress as parameters
        PRINT return invoice
        WRITE return invoice to a file
        UPDATE land status to "Available"
        WRITE updated land details to file
        RETURN returnLand
    END IF
END WHILE
ELSE
    PRINT land available message
END IF
ELSE
    PRINT invalid kitta number message
END IF
EXCEPT ValueError
```

PRINT invalid kitta number message

END TRY

END WHILE

END returnLand function

5.3 display.py

CREATE a function named display

PRINT welcome message and menu options

END display function

CREATE a function named printLand with parameter landDetails

PRINT header for available land

PRINT column headers for land details

FOR EACH rowOfLand IN landDetails

EXTRACT kittaNumber, city, direction, area, price, and status from rowOfLand

PRINT formatted row of land details

END FOR

PRINT footer for available land

END printLand function

5.4 operation.py

IMPORT datetime

CREATE a function named dateAndTime

GET current year, month, day, hour, minute, and second using datetime module

CONCATENATE year, month, day, hour, minute, and second into a string separated by “-”

RETURN the concatenated dateAndTime string

END dateAndTime function

5.5 readLand.py

CREATE a function named readLand

OPEN "land.txt" file in read mode

INITIALIZE landDetails list

FOR EACH line **IN** file.readlines()

REMOVE newline character from lands string

SPLIT lands string using "," as the separator

APPEND the resulting list to landDetails

END FOR

RETURN landDetails

END readLand function

5.6 writeLand.py

CREATE a function named writeLand with parameter landDetails

OPEN "land.txt" file in write mode

FOR EACH rowOfLand **IN** landDetails

WRITE the rowOfLand to the file separated by "," and append newline character

END FOR

CLOSE the file

END writeLand function

6. Data structure

Data structures are like containers which help us to organize and group different kinds of data and information. They can be arranged and changed in different ways. In Python there are important data structures such as lists, Integer, string, sets and dictionaries. This software uses specific types of data to work and store different types of information. Here are the types of data and basic building blocks that this software uses Data structures are fundamental concepts in computer science that enable the organization, storage and manipulation of data in an efficient and organized manner. (jain, n.d.) They play a crucial role in designing and implementing algorithms and software systems. Data structures provide a framework for managing data of various types and complexities allowing for optimized access, storage and manipulation operations.

6.1 Primitive data structure

Primitive data structures are fundamental building blocks in programming languages which representing basic values. Integers are whole numbers without decimal points used for numerical calculations and indexing. Booleans represent logical values True or False, crucial for decision-making and conditional statements. Strings are sequences of characters, utilized for text processing and manipulation. Floats represent real numbers with decimal points, essential for mathematical calculations requiring precision. These data types are indivisible and immutable, serving as the foundation for more complex data structures and operations in programming.

6.1.1 Integer

The int class represents the values as integers. which are whole numbers without fractions or decimals. These integers can encompass both positive and negative numbers. (Hege, n.d.) They have an infinite length allowing them to be as huge as needed. In that project the code utilizes integers when developing the software. As an example where users can make selections such as renting returning or exiting by providing a number input of integer (int) type as shown in the image given below.

```
-----  
while True:  
    try:  
        phoneNumber = int(input("\nEnter your phone number: "))  
        break  
    except ValueError:
```

Figure 7 Integer

6.1.2 Boolean

In Python a Boolean is a simple concept used for making decisions in your code. Think of it like a light switch that can be in one of two states where either on (represented as True) or off (represented as False). Booleans are fundamental for creating conditions and determining whether something is true or false in your program. we can use this to control the flow of our code and make it respond differently based on specific conditions such as checking if a number is greater than another or if a certain condition is met. Booleans essentially help your program make choices and execute different actions accordingly.

```
while True:
    try:
        choice=int(input("\nEnter your choice (1, 2, 3, 4 or 5): "))
        if choice == 1:
            display.printLand(landDetails)

        elif choice == 2:
            rentDuration=function.rentLand(landDetails)
            rentDurationIndex=rentDuration[1]
```

Figure 8 Boolean

5.1.3 String

A string is a fundamental concept in computer programming and computer science. It refers to a sequence of characters which can include letters, digits and even any spaces. In programming strings are used to represent textual data and in a crucial component for handling and manipulating text based information. Operations on strings include concatenation joining two or more strings together in one time. Strings are often used in input/output operations, text processing, formatting, and many other tasks in or is define as a collection of one or more characters enclosed in a single or double and triple is known as a string. (Hege, n.d.)In Python, even a single word is considered a string because there is no separate type for individual characters. We use the “str” class to work with strings. In the program there are many places where we use strings as inputs mostly for things like names messages and text.

```
invoice += "Customer Name: " + customer + "\n"
invoice += "Customer Phone Number: " + str(phoneNumber) + "\n"
invoice += "Customer Address: " + address + "\n"
invoice += "Customer Email Address: " + emailAddress + "\n"
invoice += "Rent Date: " + str(currentDate) + "\n"
invoice += "Rent Duration: " + str(returnDuration) + " month(s)" + "\n"
```

Figure 9 String

6.1.4 Float

Float is a piece of code in the Python programming language that converts values into floating point numbers. Floating point numbers are numbers with decimal parts, such as 13.5, 27.11, and 71.13, while integers like 6, 22, and 3 have no decimal parts and are considered real numbers. Utilizing the float function in Python with a particular value will change it to either a decimal number or fractional representation. Simply put, the float function in Python converts integers or real numbers into floating point numbers.

6.2 Non-Primitive data structures

Non-primitive data types are flexible elements in coding that allow for the arrangement and control of groups of data. Mutable ordered sequences enclosed in square brackets [] and allowing elements to be dynamically modified are called lists. Conversely, tuples are unchangeable ordered groupings surrounded by parentheses (), perfect for holding predetermined series of items. Sets are curly brace-enclosed collections of distinct elements that are unordered, ideal for tasks such as eliminating duplicates and conducting set operations. Dictionaries, which consist of key-value pairs enclosed in curly braces {}, enable fast data retrieval and storage using distinct keys. Every complex data type has unique benefits and applications, enhancing versatility and effectiveness in programming assignments.

6.2.1 Dictionary

In Python, a dictionary is a fundamental data type that consists of key-value pairs, allowing for effective retrieval and manipulation of data. Dictionaries allow for the dynamic storage of information by being enclosed in curly braces and separated by colons. Keys must be unique and of immutable types, while values can vary. Dictionaries are essential tools for Python programs, providing simplicity and versatility in organizing and accessing data. For example:

```
user = {"name": "ram", "age": 30}
```

6.2.2 Sets

A set in Python is an unordered collection of unique elements. Set is useful for tasks like removing duplicates from lists, checking for membership and performing set operations like union and intersection. Sets are mutable allowing us to add and remove elements but their elements must be immutable. They provide efficient membership testing and set operations and making them handy for various data manipulation tasks.

For example:

```
numbers = {1, 2, 3, 4, 5}
```

6.2.3 Tuple

In Python, a tuple is a fixed ordered group of elements that cannot be changed after it is created, much like a list but with the key difference of immutability. Tuples are created by grouping items within parentheses () and dividing them with commas (','). They are frequently employed to group related data together and are especially handy for functions that provide more than one value. Because tuples cannot be changed, they provide benefits in terms of speed and ensuring the accuracy of data. For examples:

```
coordinates = (10, 20)
```

```
Animals = ("cat", "dog")
```

6.2.4List

A Python list is a flexible and changeable data structure that holds a set of elements, enabling simple editing, including and deleting of items. Lists are created by putting items inside square brackets and dividing them with commas. Lists have the capability to store elements of diverse data types and provide a range of functions for handling, iterating, and arranging. They are commonly utilized in Python for a variety of purposes, from basic data storage to intricate algorithm development.

```
if int(rowOfLand[0]) == kittaNumber:
    found = True
    if rowOfLand[5] == "Available":
        rentLand.append(rowOfLand)
```

Figure 10 List

7. Program

The Land Rental System program is built to manage Techno Property Nepal's land rentals effectively. It runs continuously, showing available lands and waiting for the administrator's input. It makes sure data is accurate by checking input and giving error messages for mistakes, keeping track of which lands are available. The program is designed with different parts for different jobs, like handling inputs and outputs, managing files, and making invoices. It updates land statuses, makes invoices, and helps with customer transactions to keep everything running smoothly. It is also designed to be easy to understand and maintain, with clear code and thorough documentation. It can handle unexpected errors without breaking, keeping the program strong. The user interface is simple and clear, making it easy for the controller to use.

7.1 Explain the process of how the program operates and functions.

A. Display Available Land

This feature displays a list of currently available land for rent. Users can access this information from the system's menu. The list includes details of Kitta number, City, Direction, Anna, Price and Status. helping users easily identify suitable properties for rent.

B. Renting land

The renting land feature enables users to rent available land. Users input their details, including name, address, and contact information, then select the desired kitta number from the list of available options. After specifying the rental duration, the system generates an invoice detailing the rental agreement terms and the total cost. This streamlined process simplifies the renting procedure for users.

C. Returning Land:

Returning land lets users end their rental agreements and give back leased property. They provide their details, pick the land they want to return and specify how long they rented it. Then the system calculates any extra charges, like late fees and confirms the rentals end. This makes returning rented land easy and clear for users.

D. Creating Invoice and file generation

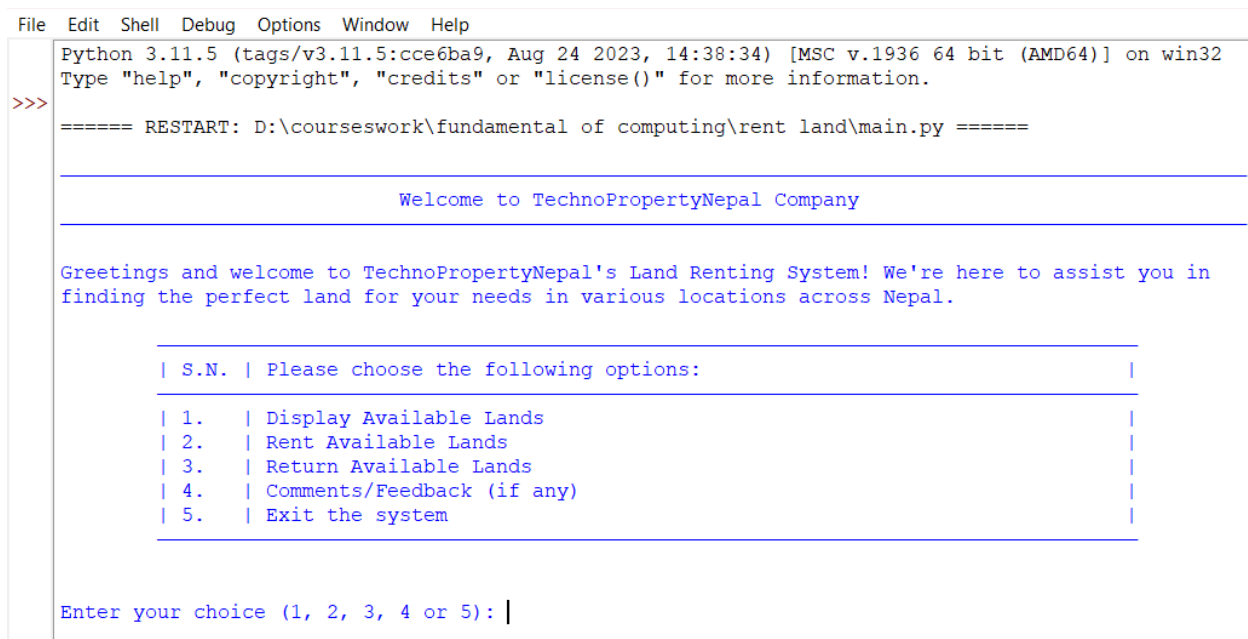
Creating an invoice involves generating a document that outlines the details of a land rental transaction. This document typically includes information such as the customer's name, contact details, rented land details, total rental cost, and any additional charges or fees. The invoice provides a clear record of the rental agreement for both the customer and the rental company.

E. Exit program

Exit the program means closing it down. When users choose this option, the program stops running, clears everything and closes the window. It is like saying “goodbye” to the program and going back to what you were doing before.

7.2 Overview of Land Rental System.

When we open the program then will see a menu. It lets us to check available lands, rent land, return a rented land, give feedback, or exit. If we click on Display Available Lands, we will see a list of lands to rent. To rent land, just fill in our information and how long we need it. If we have rented before, we can return it by clicking Return. we can also leave feedback. When we are done, just click Exit the system.



```
File Edit Shell Debug Options Window Help
Python 3.11.5 (tags/v3.11.5:cce6ba9, Aug 24 2023, 14:38:34) [MSC v.1936 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: D:\courseswork\fundamental of computing\rent land\main.py =====

Welcome to TechnoPropertyNepal Company

Greetings and welcome to TechnoPropertyNepal's Land Renting System! We're here to assist you in
finding the perfect land for your needs in various locations across Nepal.

| S.N. | Please choose the following options: |
|-----|-----|
| 1.   | Display Available Lands             |
| 2.   | Rent Available Lands                |
| 3.   | Return Available Lands              |
| 4.   | Comments/Feedback (if any)          |
| 5.   | Exit the system                     |
|-----|-----|

Enter your choice (1, 2, 3, 4 or 5): |
```

Figure 11 Overview of land rental system

7.3 Display Available Lands

When a user Choosing option 1(Display Available Lands) it allows users to see a list of lands that are currently available for rent. This list provides essential details such as the location of each land, kitta number, city, direction, Anna, price and its availability status, indicating whether it is ready for rental or already occupied. It is a straightforward way for users to check the available options before making a decision to rent a land.

Available land for Rent						
Kitta Number	City	Direction	Anna	Price	Status	
101	Kathmandu	North	8	150000	Available	
102	Pokhara	West	7	120000	Available	
103	Bhaktapur	East	4	70000	Not Available	
104	Lalitpur	South	5	80000	Not Available	
105	Rukum	West	3	30000	Available	
106	Jhapa	East	6	50000	Not Available	
107	Rolpa	North	2	40000	Available	

Enter your choice (1, 2, 3, 4 or 5):

Figure 12 Displaying available lands

7.4 Renting process

- **Select Rent Option:**
User picks “Rent Available Lands” from the menu.
- **Enter Details:**
User inputs their information (name, address, contact and Gmail.
- **See Available Lands:**
Program shows available lands.
- **Choose Land:**
User selects the land they want.
- **Enter Rental Duration:**
User specifies how long they want to rent.
- **Get Invoice:**
Program calculates the cost and shows a rent invoice.

S.N.	Please choose the following options:					
1.	Display Available Lands					
2.	Rent Available Lands					
3.	Return Available Lands					
4.	Comments/Feedback (if any)					
5.	Exit the system					

Enter your choice (1, 2, 3, 4 or 5): 2

★ Greetings! You are renting land ★

Please provide us your information.

Enter customer name: Raman

Enter customer address: Tokha

Enter your phone number: 9848767685

Enter customer email address: raman@gmail.com

Available land for Rent

Kitta Number	City	Direction	Anna	Price	Status
101	Kathmandu	North	8	150000	Available
102	Pokhara	West	7	120000	Available
103	Bhaktapur	East	4	70000	Not Available
104	Lalitpur	South	5	80000	Not Available
105	Rukum	West	3	30000	Available
106	Jhapa	East	6	50000	Not Available
107	Rolpa	North	2	40000	Available

Enter kitta number to rent: |

Figure 13 Renting Land process

Kitta Number	City	Direction	Anna	Price	Status
101	Kathmandu	North	8	150000	Available
102	Pokhara	West	7	120000	Available
103	Bhaktapur	East	4	70000	Not Available
104	Lalitpur	South	5	80000	Not Available
105	Rukum	West	3	30000	Available
106	Jhapa	East	6	50000	Not Available
107	Rolpa	North	2	40000	Available

Enter kitta number to rent: 101

Do you want to rent more lands (y/n)? : n

Enter duration of rent (in months): 8

Rented Land Invoice

Customer Name: Raman
 Customer Phone Number: 9848767685
 Customer Address: Tokha
 Customer Email Address: raman@gmail.com
 Rent Date: 2024-04-18
 Rent Duration: 8 month(s)

Kitta Number	City	Direction	Anna	Price
101	Kathmandu	North	8	Rs 1200000

Total Amount: NPR 1200000

We sincerely appreciate your decision to choose our system.

For any furthur inquiry or assistance, you can contact us via RamanNath@gmail.com or 9876543210.

Land has been rented successfully.

Figure 14 Renting Land process continue

Description: When users choose to rent lands, they first enter their information (Name, address, contact and Gmail. Then, they see a list of available lands with details (kittta number, city, direction and price. After picking a land, they say how long they want to rent it. Finally, the program gives them a clear bill. This way, users can easily rent lands that fit their needs.

7.5 Returning process

- **Select Return Option:**
Choose “Return Available Lands” from the menu.
- **Enter Details:**
Provide your name and contact information.
- **See Rented Lands:**
View the lands you’ve rented.
- **Choose Land to Return:**
Pick the land you want to return.
- **Enter Return Duration:**
Specify how long you’ve rented the land.
- **Generate Return Invoice:**
Get a bill for your return, including any fees.

Enter your choice (1, 2, 3, 4 or 5): 3

★ Greetings! You are returning land ★

Please provide us your information.

Enter customer name: Raman

Enter customer address: tokha

Enter your phone number: 9848767685

Enter customer email address: raman@gmail.com

Available land for Rent

Kitta Number	City	Direction	Anna	Price	Status
101	Kathmandu	North	8	150000	Not Available
102	Pokhara	West	7	120000	Available
103	Bhaktapur	East	4	70000	Not Available
104	Lalitpur	South	5	80000	Not Available
105	Rukum	West	3	30000	Available
106	Jhapa	East	6	50000	Not Available
107	Rolpa	North	2	40000	Available

Enter kitta number to return: 101

Figure 15 Returning Land process

```
Enter kitta number to return: 101
Do you want to return more lands (y/n)? : n
Enter duration of return (in months): 10

-----
                        Returned Land Invoice
-----

Customer Name: Raman
Customer Phone Number: 9848767685
Customer Address: tokha
Customer Email Address: raman@gmail.com
Rent Date: 2024-04-18
Rent Duration: 10 month(s)

-----
| Kitta Number | City           | Direction   | Anna        | Price       |
-----
| 101          | Kathmandu     | North       | 8           | Rs 1500000  |
-----

Total Fine: NPR 300000
Total Amount: NPR 1500000

-----

We sincerely appreciate your decision to choose our system.
For any furthur inquiry or assistance, you can contact us via RamanNath@gmail.com or 9876543210.

-----

Land has been returned successfully.
```

Figure 16 Returning Land Continue

Description: When a user picks option 3 (Return Available Lands), they can see a list of the lands they have rented. Each land's details (kitta number, city, direction, price, and availability status, are displayed. This helps users easily spot the land they want to return and understand its rental status. This simple presentation makes it easy for users to manage their rented lands effectively.

7.6 Creation of a Txt file

When a user rents or returns a land, the program automatically creates a text file containing all the relevant details of the transaction. This text file serves as a record of the rental or return process, capturing information such as the customer's name and contact details, the land's details, the rental duration, and the total amount due. By creating a text file for each transaction, the program ensures that all data is stored securely and can be easily accessed for future reference or auditing purposes.

7.6.1 Rented land text file

When a user rents a land, the program generates an invoice in the form of a text file. This invoice contains detailed information about the rental transaction, including the customer's name, contact details, the rented land specifics (such as kitta number, city, and direction) the rental duration and the total amount due. By providing this invoice, the program ensures transparency and clarity regarding the rental terms and costs, facilitating a smooth rental process for both the customer and the rental company.

----- Rented Land Invoice -----					
Customer Name: Raman Customer Phone Number: 9848767685 Customer Address: Tokha Customer Email Address: raman@gmail.com Rent Date: 2024-04-18 Rent Duration: 8 month(s)					

Kitta Number	City	Direction	Anna	Price	
101	Kathmandu	North	8	Rs 1200000	

Total Amount: NPR 1200000					

We sincerely appreciate your decision to choose our system.					
For any further inquiry or assistance, you can contact us via RamanNath@gmail.com or 9876543210.					

Figure 17 Creation of rent txt file

7.6.2 Returned land text file

When a user returns a rented land, the program generates an invoice in the form of a text file. This invoice includes full details about the return transaction, such as the customer's name, contact information, the land's specifics (such as kitta number and city), the rental duration, any applicable late fees (if the return is overdue), and the total amount refunded or outstanding. By providing this invoice, the program ensures transparency and documentation of the return process, enabling both the customer and the rental company to keep accurate records of the transaction.


```
-----
                          Returned Land Invoice
-----

Customer Name: Raman
Customer Phone Number: 9848767685
Customer Address: tokha
Customer Email Address: raman@gmail.com
Rent Date: 2024-04-18
Rent Duration: 10 month(s)

-----
| Kitta Number | City           | Direction   | Anna        | Price       |
-----
| 101          | Kathmandu     | North       | 8           | Rs 1500000  |
-----

Total Fine: NPR 300000

Total Amount: NPR 1500000

-----

We sincerely appreciate your decision to choose our system.

For any furthur inquiry or assistance, you can contact us via RamanNath@gmail.com or 9876543210.

-----
```

Figure 18 Creation of return txt file

7.7 Exiting the Program.

Once the user selects an option 5 from the menu and completes the desired action (such as renting or returning a land), the program displays a confirmation message indicating the “Thank you for using our system. Do visit again!”. Following this confirmation, the program terminates gracefully, ending the session and returning the user to their system environment. This ensures that the program closes properly after each interaction, providing a seamless user experience and preventing unnecessary resource consumption.

```

Welcome to TechnoPropertyNepal Company

Greetings and welcome to TechnoPropertyNepal's Land Renting System! We're here to assist you in
finding the perfect land for your needs in various locations across Nepal.

| S.N. | Please choose the following options: |
| 1.   | Display Available Lands             |
| 2.   | Rent Available Lands                |
| 3.   | Return Available Lands              |
| 4.   | Comments/Feedback (if any)         |
| 5.   | Exit the system                    |

Enter your choice (1, 2, 3, 4 or 5): 5

Thank you for using our system. Do visit again!
```

Figure 19 Creation of txt file

8. Testing

During testing, we carefully look at the program to ensure that it is functioning correctly. We use strict testing methods to verify every parts of the program, making sure it follows all guidelines and functionality as intended. Screenshots are captured during the testing process to show the actions taken, such as inputting information and receiving results. These screenshots serve as a proof that we have tested the program and assist us in monitoring our progress. By testing the program very carefully, we make sure it works right and meets the rules we set.

8.1 Test 1: Show implementation of try, except

8.1.1 Provide invalid input and show the message

Testing table

Objective	Test error handling with try and except blocks.
Action	Input invalid data, a letter instead of a number.
Expected Result	Program catches the error and displays an invalid message.
Actual Result	Error is handled as expected.
Conclusion	Error handling works well

Table 2 Provide invalid input and show message

Proof:

Provide invalid data, a letter instead of a number

```

Welcome to TechnoPropertyNepal Company

Greetings and welcome to TechnoPropertyNepal's Land Renting System! We're here to assist you in
finding the perfect land for your needs in various locations across Nepal.

| S.N. | Please choose the following options: |
| 1.   | Display Available Lands             |
| 2.   | Rent Available Lands                |
| 3.   | Return Available Lands              |
| 4.   | Comments/Feedback (if any)         |
| 5.   | Exit the system                    |

Enter your choice (1, 2, 3, 4 or 5): a

Invalid choice. Please enter a valid choice.

Enter your choice (1, 2, 3, 4 or 5): |

```

Figure 20 Provide invalid data a letter instead of a number

8.2 Test 2: Selection rent and return of lands

8.2.1 Provide the negative value as input

Testing table

Objective	Test program's response to negative input for rental or return duration.
Action	Input negative value
Expected Result	Program rejects negative input, prompts for valid entry.
Actual Result	Program correctly rejects negative input.
Conclusion	Program behaves as expected, rejecting negative input.

Table 3 Provide the negative value as input

Proof:

```

Welcome to TechnoPropertyNepal Company

Greetings and welcome to TechnoPropertyNepal's Land Renting System! We're here to assist you in
finding the perfect land for your needs in various locations across Nepal.

| S.N. | Please choose the following options: |
| 1.   | Display Available Lands             |
| 2.   | Rent Available Lands                |
| 3.   | Return Available Lands              |
| 4.   | Comments/Feedback (if any)         |
| 5.   | Exit the system                    |

Enter your choice (1, 2, 3, 4 or 5): -2

Please enter your choice as (1, 2, 3, 4, or 5).

Enter your choice (1, 2, 3, 4 or 5):

```

Figure 21 Provide negative value as input

8.2.1 Provide the non-existed value as input

Testing table

Objective	Selection rent and return of lands
Action	Input a non-existent option.
Expected Result	The program should display a message indicating that the option is invalid.
Actual Result	The program correctly responds with an error message.
Conclusion	The program handles invalid selections effectively.

*Table 4 Provide the non-existed value as input***Proof:**

```

| S.N. | Please choose the following options: |
|-----|-----|
| 1.   | Display Available Lands              |
| 2.   | Rent Available Lands                 |
| 3.   | Return Available Lands               |
| 4.   | Comments/Feedback (if any)          |
| 5.   | Exit the system                     |
|-----|-----|

Enter your choice (1, 2, 3, 4 or 5): %

Invalid choice. Please enter a valid choice.

Enter your choice (1, 2, 3, 4 or 5): |

```

Figure 22 Provide non existed value as input

8.3 Test 3: File generation of renting of land(s) (Renting multiple land(s))

8.3.1 Show complete renting process

Testing table

Objective	Validate file generation for renting multiple lands.
Action	Complete the renting process for multiple lands and verify file creation.
Expected Result	The program should display the renting process in the shell and generate a text file containing details of the rented lands.
Actual Result	The renting process is successfully displayed in the shell, and a text file with the rented lands' details is created.
Conclusion	The program correctly records the renting process, creating a file for future reference.

Figure 23 File generation of renting of lands renting multiple

Screenshot of the full renting process

```

Welcome to TechnoPropertyNepal Company

Greetings and welcome to TechnoPropertyNepal's Land Renting System! We're here to assist you in
finding the perfect land for your needs in various locations across Nepal.

| S.N. | Please choose the following options: |
| 1.   | Display Available Lands              |
| 2.   | Rent Available Lands                 |
| 3.   | Return Available Lands               |
| 4.   | Comments/Feedback (if any)          |
| 5.   | Exit the system                     |

Enter your choice (1, 2, 3, 4 or 5): 2

★ Greetings! You are renting land ★
-----
Please provide us your information.

Enter customer name: nath
Enter customer address: tokha
Enter your phone number: 9808010874
Enter customer email address: raman@gmail.com

```

Figure 24 Full Renting process

```

Available land for Rent

| Kitta Number | City           | Direction | Anna | Price  | Status      |
| 101          | Kathmandu     | North    | 8    | 150000 | Available   |
| 102          | Pokhara       | West     | 7    | 120000 | Not Available |
| 103          | Bhaktapur     | East     | 4    | 70000  | Available   |
| 104          | Lalitpur      | South    | 5    | 80000  | Available   |
| 105          | Rukum         | West     | 3    | 30000  | Available   |
| 106          | Jhapa         | East     | 6    | 50000  | Not Available |
| 107          | Rolpa         | North    | 2    | 40000  | Available   |

Enter kitta number to rent: 103
Do you want to rent more lands (y/n)? : y
Enter kitta number to rent: 104
Do you want to rent more lands (y/n)? : n
Enter duration of rent (in months): 8

```

Figure 25 Renting process continue

Rented Land Invoice					
Customer Name: nath Customer Phone Number: 9808010874 Customer Address: tokha Customer Email Address: raman@gmail.com Rent Date: 2024-04-18 Rent Duration: 8 month(s)					
Kitta Number	City	Direction	Anna	Price	
103	Bhaktapur	East	4	Rs 560000	
104	Lalitpur	South	5	Rs 640000	
Total Amount: NPR 1200000					
We sincerely appreciate your decision to choose our system. For any furthur inquiry or assistance, you can contact us via RamanNath@gmail.com or 9876543210.					
Land has been rented successfully.					

8.3.2 Show output in the shell as well

Tasting table

Objective	File generation for renting lands (Renting multiple lands)
Action	Simulate the complete renting process for multiple lands.
Expected Result	The renting process should be displayed step by step in the shell and should generate a text file containing details of the rented lands.
Actual Result	The program accurately executes the renting process and generates the expected output.
Conclusion	The renting process functions correctly, providing a clear record of rented lands in a text file.

Table 5 Show output in cell as well

Screenshot of proof**Proof**

Rented Land Invoice					
Customer Name: nath					
Customer Phone Number: 9808010874					
Customer Address: tokha					
Customer Email Address: raman@gmail.com					
Rent Date: 2024-04-18					
Rent Duration: 8 month(s)					
Kitta Number	City	Direction	Anna	Price	
103	Bhaktapur	East	4	Rs 560000	
104	Lalitpur	South	5	Rs 640000	
Total Amount: NPR 1200000					
We sincerely appreciate your decision to choose our system.					
For any furthur inquiry or assistance, you can contact us via RamanNath@gmail.com or 9876543210.					
Land has been rented successfully.					

Figure 26 Showing output in cell as well

8.3.3 Finally show the rented lands details in a text file

Testing table

Objective	Renting Multiple Lands File Generation
Action	Rent multiple lands and display details in a text file.
Expected Result	Rented lands details shown in a text file.
Actual Result	Rented lands details displayed in the text file.
Conclusion	Successful generation of rented lands details in the text file.

*Figure 27 Finally show the rented lands details in a text file***Proof:**

Rented Land Invoice					

Customer Name: nath					
Customer Phone Number: 9808010874					
Customer Address: tokha					
Customer Email Address: raman@gmail.com					
Rent Date: 2024-04-18					
Rent Duration: 8 month(s)					

Kitta Number	City	Direction	Anna	Price	
103	Bhaktapur	East	4	Rs 560000	
104	Lalitpur	South	5	Rs 640000	

Total Amount: NPR 1200000					

We sincerely appreciate your decision to choose our system.					
For any furthur inquiry or assistance, you can contact us via RamanNath@gmail.com or 9876543210.					

Figure 28 Finally showing the rented lands in txt file

8.4 Test 4: File generation of returning process of land(s) (Returning multiple land(s))

8.4.1 Show the complete returning process of the land(s).

Testing table

Objective	Returning Multiple Lands File Generation
Action	Return multiple lands and display details in a text file.
Expected Result	Returned lands details shown in a text file.
Actual Result	Returned lands details displayed in the text file.
Conclusion	Successful generation of returned lands details is show on the invoice and store in text file.

Table 6 Show the complete returning process of the land(s).

Screenshot of full multiple retuning process

Enter your choice (1, 2, 3, 4 or 5): 3

★ Greetings! You are returning land ★

Please provide us your information.

Enter customer name: nath

Enter customer address: tokha

Enter your phone number: 9808010874

Enter customer email address: raman@gmail.com

Available land for Rent

Kitta Number	City	Direction	Anna	Price	Status
101	Kathmandu	North	8	150000	Available
102	Pokhara	West	7	120000	Not Available
103	Bhaktapur	East	4	70000	Not Available
104	Lalitpur	South	5	80000	Not Available
105	Rukum	West	3	30000	Available
106	Jhapa	East	6	50000	Not Available
107	Rolpa	North	2	40000	Available

Enter kitta number to return: 103

Do you want to return more lands (y/n)? : y

Enter kitta number to return: 104

Do you want to return more lands (y/n)? : n

Enter duration of return (in months): 9

Figure 29 Show the complete returning process of the land(s) continue

Returned Land Invoice				
Customer Name: nath Customer Phone Number: 9808010874 Customer Address: tokha Customer Email Address: raman@gmail.com Rent Date: 2024-04-18 Rent Duration: 9 month(s)				
Kitta Number	City	Direction	Anna	Price
103	Bhaktapur	East	4	Rs 630000
104	Lalitpur	South	5	Rs 720000
Total Fine: NPR 150000				
Total Amount: NPR 1350000				
We sincerely appreciate your decision to choose our system. For any furthur inquiry or assistance, you can contact us via RamanNath@gmail.com or 9876543210.				
Land has been returned successfully.				

8.4.2 Show output in the shell as well.

Testing table

Objective	Returning Multiple Lands File Generation
Action	Return multiple lands and display details in a text file.
Expected Result	Returned lands details shown in both the text file and the shell.
Actual Result	Returned lands details displayed in the text file and the shell.
Conclusion	Successful generation of returned lands details in both the text file and the shell.

Table 7 Show output in the shell as well.

Returned Land Invoice				
Customer Name: nath Customer Phone Number: 9808010874 Customer Address: tokha Customer Email Address: raman@gmail.com Rent Date: 2024-04-18 Rent Duration: 9 month(s)				
Kitta Number	City	Direction	Anna	Price
103	Bhaktapur	East	4	Rs 630000
104	Lalitpur	South	5	Rs 720000
Total Fine: NPR 150000				
Total Amount: NPR 1350000				
We sincerely appreciate your decision to choose our system. For any furthur inquiry or assistance, you can contact us via RamanNath@gmail.com or 9876543210.				
Land has been returned successfully.				

Figure 30 Returning Multiple Lands File Generation

8.4.3 Finally show the returned land(s) details in text file.

Testing table

Objective	Returning Multiple Lands File Generation
Action	Return multiple lands and display details in a text file.
Expected Result	Returned lands details shown in a text file.
Actual Result	Returned lands details displayed in the text file.
Conclusion	Successful generation of returned lands details in a text file.

Table 8 Finally show the returned land(s) details in text file.

Returned Land Invoice

Customer Name: nath
 Customer Phone Number: 9808010874
 Customer Address: tokha
 Customer Email Address: raman@gmail.com
 Rent Date: 2024-04-18
 Rent Duration: 9 month(s)

Kitta Number	City	Direction	Anna	Price
103	Bhaktapur	East	4	Rs 630000
104	Lalitpur	South	5	Rs 720000

Total Fine: NPR 150000

Total Amount: NPR 1350000

We sincerely appreciate your decision to choose our system.

For any furthur inquiry or assistance, you can contact us via RamanNath@gmail.com or 9876543210.

Figure 31 Finally show the returned land(s) details in text file.

8.5 Test 5 Show the update in stock of land(s)

8.5.1 Data.txt

File	Edit	View
101,Kathmandu,North,8,150000,Available 102,Pokhara,West,7,120000,Not Available 103,Bhaktapur,East,4,70000,Available 104,Lalitpur,South,5,80000,Available 105,Rukum,West,3,30000,Available 106,Jhapa,East,6,50000,Not Available 107,Rolpa,North,2,40000,Available		

Figure 32 Data.txt

8.5.2 Show the availability land being rented to 'Not available' after renting the land (Update should be reflected in a .txt file as well, availability of the land should be not available)

Testing table

Objective	Update Stock of Lands Availability
Action	Rent a land and verify the update in stock availability.
Expected Result	Availability of the rented land should change to 'Not available' in the system and reflect in a .txt file.
Actual Result	After renting the land, the availability status was updated to 'Not available' both in the system and the .txt file.
Conclusion	Successful update in the stock availability of the rented land.

Table 9 Update Stock of Lands Availability

Proof:

File	Edit	View
<pre> 101,Kathmandu,North,8,150000,Available 102,Pokhara,West,7,120000,Not Available 103,Bhaktapur,East,4,70000,Not Available 104,Lalitpur,South,5,80000,Not Available 105,Rukum,West,3,30000,Available 106,Jhapa,East,6,50000,Not Available 107,Rolpa,North,2,40000,Available </pre>		

Figure 33 Update Stock of Lands Availability

8.5.3 how the availability of land being returned to 'Available' after returning the land (Update should be reflected in a .txt file as well, non-availability of the land should be available)

Testing table

Objective	Update Stock of Lands Availability
Action	Return lands and verify the update in availability status.
Expected Result	Availability status of returned lands updated to 'Available'.
Actual Result	Availability status of returned land updated to 'Available'.
Conclusion	Successful update in the availability status of land to 'Available' after returning, reflected in the .txt file.

Table 10 Update Stock of Lands Availability

File	Edit	View
<pre> 101,Kathmandu,North,8,150000,Available 102,Pokhara,West,7,120000,Not Available 103,Bhaktapur,East,4,70000,Available 104,Lalitpur,South,5,80000,Available 105,Rukum,West,3,30000,Available 106,Jhapa,East,6,50000,Not Available 107,Rolpa,North,2,40000,Available </pre>		

Figure 34 Update Stock of Lands Availability

9. Conclusion

This experience with my Python programming project in the Fundamentals of Computing course has been really important to me. It is a big step in my learning journey so I will always remember. I worked on software development, making algorithms and utilizing data structures in Python, particularly through creating the Land Rental System which involved learning and facing challenges.

I faced problems in arranging the report, finding reliable sources and making things clear. But I did not give up and kept going. This task showed me how important it is to use what I know to make a land rental management app, create invoices and handle inventory well. The Techno Property Nepal program does a good job of managing land rentals by keeping track of available land and renters, making invoices, and keeping records updated. It avoids mistakes by checking inputs and keeping records current. The importance of data structures in making programs run faster was a big thing we figured out. Using dictionaries for land information and lists for rental transactions made things work better. This program makes renting land from Techno Property Nepal easier for everyone involved. Going through this practical experience boosted my problem solving and technical abilities. Reflecting on it, I feel proud of what I have achieved. I did not just learn about making reports and conducting research but also enhanced my project management, critical thinking and communication skills.

This chance did not just make my technical skills better, but also boosted my ability to manage projects, think critically and communicate effectively. In the future, I am excited to use these skills in new tasks and projects. This project boosted my interest and passion for learning, leading me to explore more in computing and other fields. I am grateful to the teachers for giving me this valuable opportunity, guiding my academic journey and preparing me for future challenges.

10. References

A-143, 9th Floor, Sovereign Corporate Tower, Sector-136,, n.d. *Geeksofgeeks*. [Online]
Available at: <https://www.geeksforgeeks.org/introduction-to-microsoft-word/>
[Accessed 28 April 2024].

Hege, s., n.d. *W3 schools*. [Online]
Available at: https://www.w3schools.com/python/python_numbers.asp
[Accessed 28 April 2024].

jain, S., n.d. *Geeksofgeeks*. [Online]
Available at: <https://www.geeksforgeeks.org/python-data-structures/>
[Accessed 28 April 2024].

JGraph Ltd, Artisans' House, n.d. [Online]
Available at: <https://www.drawio.com/doc/getting-started-diagram-types>
[Accessed 28 April 2024].

TutorialsTeacher.com. , n.d. *tutorials teacher*. [Online]
Available at: <https://www.tutorialsteacher.com/python/python-idle>
[Accessed 28 April 2024].

11. Appendix

11.1 Land.txt

101,Kathmandu,North,8,150000,Available
102,Pokhara,West,7,120000,Available
103,Bhaktapur,East,4,70000,Not Available
104,Lalitpur,South,5,80000,Not Available
105,Rukum,West,3,30000,Available
106,Jhapa,East,6,50000,Not Available
107,Tokha,North,2,40000,Available

11.2 Main.py

```
import display
import readLand
import function

def main():
    landDetails=readLand.readLand()
    display.display()
    rentDurationIndex=None
    while True:
        try:
            choice=int(input("\nEnter your choice (1, 2, 3, 4 or 5): "))
            if choice == 1:
                display.printLand(landDetails)
```

```
elif choice == 2:
    rentDuration=function.rentLand(landDetails)
    rentDurationIndex=rentDuration[1]

elif choice == 3:
    function.returnLand(landDetails, rentDurationIndex)

elif choice == 4:
    input("\nProvide us your feedback (press Enter to finish):\n\n")

print("\n_____")
print("_____")
print("                Thank you for your feedback!")

print("_____")
print("_____")

elif choice == 5:

print("\n_____")
print("_____")
print("                Thank you for using our system. Do visit again!")

print("_____")
print("_____")
print("_____")
print("                \n")

break

else:

print("\n_____")
print("_____")
print("                Please enter your choice as (1, 2, 3, 4, or 5).")
```

```

print("_____")
_____")
    except ValueError:

print("\n_____")
_____")
    print("                Invalid choice. Please enter a valid choice.")

print("_____")
_____")

main()

```

11.3 function.py

```

import display
import writeLand
import datetime
import operation
import readLand

```

```

currentDate = datetime.datetime.now().date()
landDetails=readLand.readLand()

```

```

def generateRentInvoice(customer, address, phoneNumber, rentLand, rentDuration,
emailAddress):

```

```

    invoice = "\n"+"-----"
    -----+"\n"

    invoice += "                Rented Land Invoice"

    invoice += "\n"+"-----"
    -----+"\n\n"

```

```

invoice += "Customer Name: " + customer + "\n"
invoice += "Customer Phone Number: " + str(phoneNumber) + "\n"
invoice += "Customer Address: " + address + "\n"
invoice += "Customer Email Address: " + emailAddress + "\n"
invoice += "Rent Date: " + str(currentDate) + "\n"
invoice += "Rent Duration: " + str(rentDuration) + " month(s)" + "\n"
invoice += "\n" + "-----"
"-----" + "\n"

invoice += "| Kitta Number | City          | Direction    | Anna        | Price"
|
invoice += "\n" + "-----"
"-----" + "\n"

totalAmount = 0
for rowOfLand in rentLand:
    kittaNumber=rowOfLand[0]
    city=rowOfLand[1]
    direction=rowOfLand[2]
    anna=rowOfLand[3]
    price = int(rowOfLand[4]) * rentDuration
    totalAmount += price
    invoice += ("| " + str(kittaNumber) + " " * (13 - len(str(kittaNumber)))) +
        "| " + city + " " * (22 - len(city)) +
        "| " + direction + " " * (18 - len(direction)) +
        "| " + str(anna) + " " * (17 - len(str(anna))) +
        "| Rs " + str(price) + " " * (14 - len(str(price))) + "|" + "\n"
    )
invoice += "-----"
"-----" + "\n"

invoice += "\nTotal Amount: NPR " + str(totalAmount) + "\n\n"

```

```

    invoice += "-----"
    invoice += "\n\n"

    invoice += "We sincerely appreciate your decision to choose our system."+"\n\n"

    invoice += "For any furthur inquiry or assistance, you can contact us via
RamanNath@gmail.com or 9876543210."+"\n\n"

    invoice += "-----"
    "

    return invoice

```

```
def rentLand(landDetails):
```

```
    rentLand = []
```

```
    print("\n-----")
```

```
        print("          ★ Greetings! You are renting land ★")
```

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

```
        print("          Please provide us your information.")
```

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

```
    while True:
```

```
        customer = input("\nEnter customer name: ")
```

```
        if customer == "":
```

```
            print("\n-----")
```

```
                print("          Customer name cannot be empty. Please enter a valid
customer name.")
```

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

```
        else:
```



```
        break
    while True:
        address = input("\nEnter customer address: ")
        if address == "":
            print("\n_____")
            print("                Address cannot be empty. Please enter a valid address.")
            print("\n_____")
        else:
            break
    while True:
        try:
            phoneNumber = int(input("\nEnter your phone number: "))
            break
        except ValueError:
            print("\n_____")
            print("                Invalid number! Please enter a valid phone number.")
            print("\n_____")

    while True:
        emailAddress = input("\nEnter customer email address: ")
        if emailAddress == "":
            print("\n_____")
```

```

        print("Email address cannot be empty. Please enter a valid address.")

print("_____")

else:

    break

display.printLand(landDetails)

while True:

    try:

        kittaNumber = int(input("\nEnter kitta number to rent: "))

        found = False

        for rowOfLand in landDetails:

            if int(rowOfLand[0]) == kittaNumber:

                found = True

                if rowOfLand[5] == "Available":

                    rentLand.append(rowOfLand)

                    while True:

                        more = input("\nDo you want to rent more lands (y/n)? ")

                        if more.lower() == 'n':

                            while True:

                                try:

                                    rentDuration = int(input("\nEnter duration of rent (in months):"))

                                    if rentDuration <= 0:

                                        print("\n_____")

```

```

        print("Invalid duration! Duration must be
greater than 0.")

print("_____")
_____")

        continue
        break
    except ValueError:

print("\n_____")
_____")

        print("Invalid input! Please enter a valid number for
duration.")

print("_____")
_____")

        rentInvoice = generateRentInvoice(customer, address,
phoneNumber, rentLand, rentDuration, emailAddress)
        print(rentInvoice)

print("\n_____")
_____")

        print("Land has been rented successfully.")

print("_____")
_____")

        rentInvoiceName = customer+"-"+ "Rent" + "-"
        "+str(operation.dateAndTime())+".txt"
        file=open(rentInvoiceName,"w")
        file.write(rentInvoice)

        for rowOfland in rentLand:

```

```

        kittaNumber = int(rowOfland[0])
        for rowOfland in landDetails:
            if int(rowOfland[0]) == kittaNumber:
                rowOfland[5] = "Not Available"
        writeLand.writeLand(landDetails)
        return rentLand, rentDuration

    if more.lower() == 'y':
        break
    else:

print("\n_____")
print("Invalid input. Please enter 'y' or 'n'.")

print("_____")

    else:

print("\n_____")
print("This land is not available for rent.")

print("_____")

    if not found:

print("\n_____")
print("Please enter a valid Kitta Number.")

```

```

print("_____")
_____
except ValueError:

print("\n_____")
_____
print("          Invalid Kitta Number! Please enter a vaild Kitta Number.")

print("_____")
_____

```

```

def generateReturnInvoice(customer, address, phoneNumber, returnLand, rentDuration,
returnDuration, emailAddress):

```

```

    invoice = "\n"+"-----"
    -----+"\n"

    invoice += "          Returned Land Invoice"

    invoice += "\n"+"-----"
    -----+"\n\n"

    invoice += "Customer Name: " + customer + "\n"
    invoice += "Customer Phone Number: " + str(phoneNumber) + "\n"
    invoice += "Customer Address: " + address + "\n"
    invoice += "Customer Email Address: " + emailAddress + "\n"
    invoice += "Rent Date: " + str(currentDate) + "\n"
    invoice += "Rent Duration: " + str(returnDuration) + " month(s)" + "\n"

    invoice += "\n"+"-----"
    -----+"\n"

    invoice += "| Kitta Number | City          | Direction      | Anna          | Price
|"

```

```

invoice += "\n" + "-----"
-----" + "\n"

```

```
totalAmount = 0
```

```
totalLateAmount = 0
```

```
lateFine=0
```

```
for rowOfLand in returnLand:
```

```
    kittaNumber=rowOfLand[0]
```

```
    city=rowOfLand[1]
```

```
    direction=rowOfLand[2]
```

```
    anna=rowOfLand[3]
```

```
    price=rowOfLand[4]
```

```
    if returnDuration>rentDuration:
```

```
        lateFine=(returnDuration-rentDuration)*int(price)
```

```
    else:
```

```
        lateFine=0
```

```
    totalLateAmount+=int(lateFine)
```

```
    totalPrice = int(rowOfLand[4]) * returnDuration
```

```
    totalAmount += int(rowOfLand[4]) * rentDuration
```

```
    invoice += ("| " + str(kittaNumber) + " " * (13 - len(str(kittaNumber)))) +
```

```
        "| " + city + " " * (22 - len(city)) +
```

```
        "| " + direction + " " * (18 - len(direction)) +
```

```
        "| " + str(anna) + " " * (17 - len(str(anna))) +
```

```
        "| Rs " + str(totalPrice) + " " * (14 - len(str(totalPrice))) + "|" + "\n" )
```

```

invoice += "-----"
" + "\n\n"

```

```
if int(lateFine) > 0:
```

```
    invoice += "\nTotal Fine: NPR " + str(totalLateAmount) + "\n"
```

```

total=str(totalAmount+totalLateAmount)
invoice += "\nTotal Amount: NPR " + str(total) + "\n\n"
invoice += "-----"
"+ "\n\n"

invoice += "We sincerely appreciate your decision to choose our system."+" \n\n"

invoice += "For any furthur inquiry or assistance, you can contact us via
RamanNath@gmail.com or 9876543210."+" \n\n"

invoice += "-----"
"

return invoice

```

```

def returnLand(landDetails, rentDuration):
    returnLand=[]

```

```

print("\n-----")

print("          ★ Greetings! You are returning land ★")
print("          -----")
print("          Please provide us your information.")

print("-----")

while True:
    customer = input("\nEnter customer name: ")
    if customer == "":

print("\n-----")

        print("          Customer name cannot be empty. Please enter a valid
customer name.")

```

```
print("_____")
_____

else:
    break
while True:
    address = input("\nEnter customer address: ")
    if address == "":
        print("\n_____")
        _____
        print("          Address cannot be empty. Please enter a valid address.")

print("_____")
_____

else:
    break
while True:
    try:
        phoneNumber = int(input("\nEnter your phone number: "))
        break
    except ValueError:
        print("\n_____")
        _____
        print("          Invalid number! Please enter a valid phone number.")

print("_____")
_____

while True:
    emailAddress = input("\nEnter customer email address: ")
```



```
    if emailAddress == "":

        print("\n_____")
        print("_____")
        print("        Email address cannot be empty. Please enter a valid address.")

        print("_____")
        print("_____")

    else:
        break

display.printLand(landDetails)
while True:
    try:
        kittaNumber = int(input("\nEnter kitta number to return: "))
        found = False
        for rowOfLand in landDetails:
            if int(rowOfLand[0]) == kittaNumber:
                found = True
                if rowOfLand[5] == "Not Available":
                    returnLand.append(rowOfLand)

        while True:
            more = input("\nDo you want to return more lands (y/n)? ")
            if more.lower() == 'n':
                while True:
                    try:
                        returnDuration = int(input("\nEnter duration of return (in months): "))
```

```

        if returnDuration <= 0:

print("\n_____")
_____")
        print("        Invalid duration! Duration must be
greater than 0.")

print("_____")
_____")
        continue
        break
    except ValueError:

print("\n_____")
_____")
        print("        Invalid input! Please enter a valid number for
duration.")

print("_____")
_____")

        returnInvoice = generateReturnInvoice(customer, address,
phoneNumber, returnLand, rentDuration, returnDuration, emailAddress)
        print(returnInvoice)

print("\n_____")
_____")
        print("        Land has been returned successfully.")

print("_____")
_____")

        returnInvoiceName = customer+"-"+ "Return"+"-
"+str(operation.dateAndTime())+".txt"
        file=open(returnInvoiceName,"w")

```

```
file.write(returnInvoice)
```

```
for rowOfLand in returnLand:
```

```
    kittaNumber = int(rowOfLand[0])
```

```
    for rowOfLand in landDetails:
```

```
        if int(rowOfLand[0]) == kittaNumber:
```

```
            rowOfLand[5] = "Available"
```

```
writeLand.writeLand(landDetails)
```

```
return returnLand
```

```
if more.lower() == 'y':
```

```
    break
```

```
else:
```

```
print("\n_____")
```

```
    print("Invalid input. Please enter 'y' or 'n'.")
```

```
print("_____")
```

```
else:
```

```
print("\n_____")
```

```
    print("This land is available for rent.")
```

```
print("_____")
```

```
if not found:
```

```

print("\n_____")
print("                Please enter a vaild Kitta Number.")

print("_____")

except ValueError:

print("\n_____")
print("                Invalid Kitta Number. Please enter a vaild Kitta Number.")

print("_____")

```

11.4 display.py

```

def display():
    print("""
_____
                Welcome to TechnoPropertyNepal Company
_____
_____

```

Greetings and welcome to TechnoPropertyNepal's Land Renting System! We're here to assist you in finding the perfect land for your needs in various locations across Nepal.

| S.N. | Please choose the following options: |

| 1. | Display Available Lands |

| 2. | Rent Available Lands |

| 3. | Return Available Lands |

| 4. | Comments/Feedback (if any) |

| 5. | Exit the system |

""")

def printLand(landDetails):

print("\n_____
_____")

print("_____
_____ Available land for Rent")

print("_____
_____ \n")

print("_____
_____")

print("| Kitta Number | City | Direction | Anna | Price | Status
|")

```
print("_____")
_____")

for rowOfLand in landDetails:
    kittaNumber=rowOfLand[0]
    city=rowOfLand[1]
    direction=rowOfLand[2]
    anna=rowOfLand[3]
    price=rowOfLand[4]
    status=rowOfLand[5]

    print("|", str(kittaNumber), " " * (11 - len(str(kittaNumber))),
          "|", city, " " * (15 - len(city)),
          "|", direction, " " * (11 - len(direction)),
          "|", str(anna), " " * (11 - len(str(anna))),
          "|", str(price), " " * (11 - len(str(price))),
          "|", status, " " * (14 - len(status)), "|"
    )

print("_____")
_____")
```

11.5 operation.py

```
import datetime

def dateAndTime():
    year=datetime.datetime.now().year
    month=datetime.datetime.now().month
    day=datetime.datetime.now().day
    hour=datetime.datetime.now().hour
    minute=datetime.datetime.now().minute
    second=datetime.datetime.now().second

    dateAndTime=str(year)+"-"+str(month)+"-"+str(day)+"-"+str(hour)+"-"+str(minute)+"-
"+str(second)

    return dateAndTime
```

11.6 read land.py

```
def readLand():
    file=open("land.txt","r")
    landDetails=[]
    for lands in file.readlines():
        landDetails.append(lands.replace("\n","").split(","))
    return landDetails
```

11.7 writeLand.py

```
def writeLand(landDetails):  
    file = open("land.txt", "w")  
    for row in landDetails:  
        file.write(", ".join(row) + "\n")  
    file.close()
```