LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

CS4051NI Fundamentals of Computing

60% Individual Coursework

2023/24 Spring

Student Name: Digdarshan Bhattarai
London Met ID: 23049051
College ID: NP01CP4A230320
Assignment Due Date: Tuesday, 7th May 2024
Assignment Submission Date: Tuesday, 7th May 2024
Word Count: 6145
Project File Links:

YouTube Link:	Keep Unlisted YouTube URL of your Project Here
Google Drive Link:	Keep Google Drive URL of your Project Here with Anyone in Organization can View Option Enabled

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

List of Figures

List of Tables:

1. Introduction:

1.1 Brief Introduction:

Python is a popular interpreted, object-oriented, high-level programming language well known throughout the world for its features which make it simple and easy to use.
Due to its dynamic semantic and built-in data structures, python is best suited for development of applications. Python encourages code reuse. It is said that a good programmer is the one who tackles the given problem by keeping the code simple and non-repetitive. To support that, python helps us to minimise the repetition of code by the help of code modularity. Because of Code Modularity, we can make multiple modules, with each module including some code that helps us attain certain result and then combining those modules to create a system altogether. In Python errors are raised as exceptions rather than segmentation faults, due to which debugging is simple.

1.2 Goals and Objectives:

Through this program, we intend to provide a land rental system through which clients can rent their desired piece of land which is charged on a monthly basis.
The main goal of this coursework is to develop a program which can effectively handle data on different lands which are available to be rented, show invoices as per need, and generate bill receipts for each transaction.

The key objectives include:

- We are supposed to complete this coursework by applying the concept of Modularity which means making multiple modules first, and then combining those modules to create a system. To understand simply, modularity is the splitting of the code. Each module contains all the necessary things to execute an aspect of the desired functionality. To make a land rental system with given instructions, 4 separate modules are a must.

- To read and update a text file that contains information about the lands from various areas available to rent.

- To generate an invoice for each transaction with information containing land id, the location of the land, the direction faced by land, price, area of the land, name of the customer, date and time of rent, the duration of rent, and the total amount.

- To make it possible for a single client to rent multiple lands but the amount should be added for all the lands s/he had rented.

- To again generate an invoice when the land is returned and the invoice should include the name of the client, name of the location , the direction of the land, date and time of return, the duration of rent, area of land, and the total amount.

- To add fine to the bill on a monthly basis if a client fails to return the rented land on time.

- To provide a simple user interface for renting and returning lands.

1.3 Short Summary:

In a nutshell, our Python program will automate the process of renting and returning lands of various places of Nepal and creating invoices for the system. Our aim is to provide an efficient and user-friendly solution that will simplify the rental systems land management process, meeting all our goals and objectives. By the use of a text file and an application, we can update availability status of the land which is desired by the client and provide all the transaction details simply, finally providing the rental system with a useful tool for land management and transaction details.

1.4 Tools Used:

1.4.1 IDLE

IDLE stands for Integrated Development and Learning Environment. IDLE helps users make it simple to write Python code. IDLE is intended to be a simple IDE and suitable for beginners/learners. Also, it is cross-platform and avoids feature clutter. IDLE's fully featured text editor has tools like autocompletion and smart indent.

Some of the features of IDLE are:

- Cross-platform: It runs same on Windows, Unix, and macOS.
- Python shell window with colorizing of code input, output, and error messages.
- Multi-window text editor with multiple undo, auto completion, and other features.
- Search within any window, replace within editor windows, and search through varieties files (Python, 2024)

1.4.2 Microsoft-Word

Microsoft is a word processing program which allows the creation of both simple and complex documents. The program can work on multiple platforms, like windows, macOS, and even smartphones. Microsoft-word was developed by Charles Simonyi and Richard Brodie in 1983. MS Word can be downloaded from Microsoft Office package. It does not exist in computers and must be purchased and installed in your computers. Old versions of Microsoft Word mainly created .doc files extension but the recent versions support .doc, .docs, .htm, .html, .txt, and so on. Features like being able to change the formatting of text, editing the font type and font size, inserting tables and images are the reasons why people choose MS Word over plain text editors. (Computer Hope, 2024)

1.4.3 Draw.io

Draw.io is a free, online diagramming tool which allows you to make flowcharts, diagrams, mind maps, charts and so on. Draw.io is completely integrated with Google Drive meaning you can directly save your completed work into your Gmail Account. In this work as well, Draw.io is used to illustrate the process of the whole system in a flowchart. Moreover, it can help us visualise networks using network diagrams. Sketching the design of a website is also possible using this free tool. So basically, it is one important tool for people of all levels in the field of computer science. (FOTC, 2023)

2. Discussion and Analysis

2.1 Algorithm:

An algorithm is a step wise procedure used for solving problems or doing computations. Algorithms are used throughout the IT sector in a wide scale. Algorithms are used for performing data processing and play a huge role in automated systems. Algorithms can be used to tackle complicated tasks by breaking them down into chunks which are carried out in a step wise sequence. They can be written in natural languages, programming languages, pseudocodes, and flowcharts too. (Gillis, July, 2023)

2.1.1 Algorithm of the Course Work:

STEP 1: Start

STEP 2: Show options to rent, return and exit the program.

STEP 3: If user input = 1, go to step 6

STEP 4: If user input = 2, go to step 12

STEP 5: If user input = 3, go to step 17

STEP 6: Enter customer name.

STEP 7: Enter duration in months.

STEP 8: If incorrect value is entered in Step 7, go to Step 2. If not, go to Step 9

STEP 9: Enter kitta number of lands you want to rent. If incorrect value, go to step 2

STEP 10: Else, show the message if the user wants to rent more lands (yes/no)

STEP 11: If choice = yes go to Step 9, if choice = no, generate invoice and break the loop.

STEP 12: Display the lands which are not available and ask for the kitta number of the land the user wants to return.

STEP 13: If user inputs the kitta number of previously rented lands, go to step 14 and if not go to Step 16.

STEP 14: Display text saying land with the entered kitta number returned successfully.

STEP 15: Generate invoice for returning of rented land/s.

STEP 16: Display a text saying the entered land was never rented.

STEP 17: STOP

2.2 Flow Chart

Flow-chart is a graphical representation of an algorithm. Flow charts are often used by programmers as a program-planning tool to solve problems. Flowcharts make to use of symbols which are connected among each other to show the flow of information and processing. Creating flowcharts has some rules of its own. The statement must start with ‘start’ keyword and end with ‘end’ keyword. All the symbols in a flowchart must be connected by an arrow line. There are several advantages of flowcharts for programmers. Flowcharts are like a blueprint of a program. They can act as a guide while creating programs. Better documentation is provided by flowcharts. With the use of flowcharts, it is easier to trace errors in the program. Flowcharts help in the correct implementation of logic. Due to the ease of tracing errors, flowcharts can be extremely useful during the debugging process too. (GeeksForGeeks, Oct, 2023)

2.2.1 Flow Chart of the Course Work:

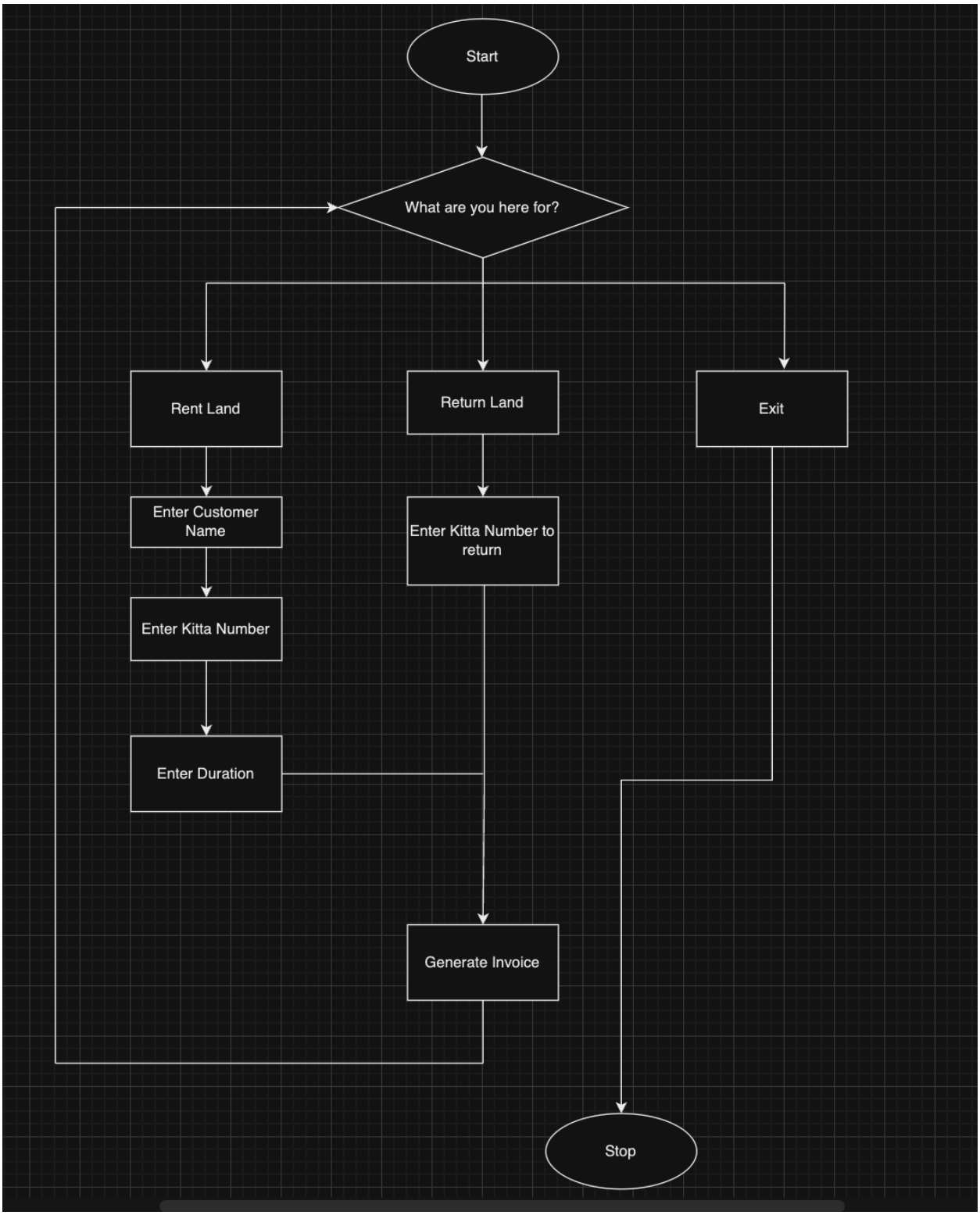

Figure 1:Flow Chart of the program.

2.3 Pseudocode:

Pseudocode is a simple way of writing programming code in English or any other language for the ease of the user. Pseudocode is not actual programming language which is why it is also known as ‘false code’. Pseudocode can be defined as a method of describing process or writing programming codes using natural language. Pseudocode in itself is not the code, but rather an explanation of what the code should perform. It is used as understandable stepwise blueprint from which programs can be written. Its main purpose is to provide clear description of the steps that will be taken in the code. Pseudocode does not have any particular syntax. Flowcharts can be difficult to modify as we must understand the whole flow of the program. So, due to the ease-of-use pseudocode is much more appealing. (Mahr, 2023)

2.3.1 Main.py Pseudocode:

Import FUNCTION read_land_data from read.py

Import FUNCTION write_land_data from write.py

Import FUNCTIONS display_available_lands, rent_land, return_land from operations.py

Define FUNCTION Main():

SET file_name : “land_info.txt”

Lands EQUALS read_land_data FUNCTION with file_name AS parameter

WHILE true:

DISPLAY “Welcome to TechnoPropertyNepal Land Rental System”

CALL display_available_lands FUNCTION

DISPLAY “Menu”

DISPLAY “1. Rent Land”

DISPLAY “2. Return Land”

DISPLAY “3. Exit”

Choice EQUALS ask for input “Enter your choice: ”

Try to execute:

IF choice EQUALS 1:

ASK for customer name as INPUT

THEN ASK for duration(in months) as INPUT

CALL FUNCTION rent_land with lands, customer_name, duration as parameters

ELSE IF choice EQUALS 2:

CALL FUNCTION return_land with 'lands' passed as parameter

ELSE IF choice EQUALS 3:

CALL FUNCTION write_land_data with 'file_name, lands' as parameters

BREAK THE LOOP

ELSE:

DISPLAY "Invalid choice. Please enter a number between 1 and 3."

CATCH EXCEPTION OF ValueError:

DISPLAY "Invalid input. Please enter a valid number."

IF __name__ EQUALS " __main__" :

CALL FUNCTION main()

2.3.2 Read.py Pseudocode:

DEFINE FUNCTION read_land_data with file_name as parameter

INITIALIZE empty list land_data[ ]

Try:

OPEN file_name in read mode
FOR line in file:
SPLIT LINE with comma as delimiter
CREATE dictionary containing land information:
‘kitta_number’: convert first index element into integer
‘city’: STRIP whitespace from second index
‘direction’: STRIP whitespace from third index
‘area’: convert fourth index element into integer
‘price’: convert fifth index element into integer
‘status’: STRIP whitespace from the sixth index
APPEND created dictionary to land_data

CATCH FileNotFoundError:

DISPLAY “Error: File not found.”

CATCH Exception as e:

DISPLAY “An error occurred: ”

RETURN land_data list

2.3.3 Write.py Pseudocode:

DEFINE FUNCTION write_land_data and pass file_name, land_data as parameters

TRY:

OPEN file with given file_name in write mode

FOR land dictionary in land_data:

WRITE land information to the file in the format:

Kitta_number, city, direction, area, price, status

Separate each attribute by comma and line by newline character

CATCH Exception as e:
DISPLAY “An exception occurred: ”

2.3.4 Operations.py Pseudocode:

Import datetime

DEFINE FUNCTION display_available_lands with lands as the parameter

DISPLAY “Available Lands:”
DISPLAY “Kitta No.TAB City/District TAB Direction\tArea TAB Price TAB Status”
For each land in lands:
If land[‘Status’] EQUALS ‘Available’
DISPLAY land details

DEFINE FUNCTION rent_land with lands, customer_name, duration as the parameters

INITIALIZE an empty list for rented_lands
SET total_amount to zero

FOR EACH land in lands:

IF land[‘status’] EQUALS ‘Available’
DISPLAY in FORMAT “kitta_number TAB city TAB direction TAB area TAB price TAB status”

WHILE True:

INPUT value for kitta_number

ASK IF user wants to continue

IF kitta_number.lower() EQUALS exit:

BREAK THE LOOP

ELSE:

FOR EACH land in lands:

IF kitta_number EQUALS integer(kitta_number) AND status EQUALS Available:

Rented_duration EQUALS duration

FUNCTION datetime.datetime.now() TO GET current datetime

Total amount EQUALS rented_duration*land['price']

CHANGE land['status'] TO 'Not Available'

APPEND rented_lands

DISPLAY "Land with {kitta number} rented successfully"

ASK IF user wants to rent more lands

IF more_lands.lower() EQUALS 'no':

GENERATE invoice and OPEN WITH file_name in write mode

WRITE "Rent invoice"

WRITE "Customer name"

WRITE "Date and time of rent"

WRITE "Rented lands:"

FOR EACH rented_lands IN rented_lands:

WRITE "kitta no, city, direction, area, total amount"

DISPLAY “Invoice generated!”

RETURN

IF more_lands.lower() EQUALS ‘yes’:

BREAK THE LOOP

ELSE:

DISPLAY “Invallid kitta number. Please try again”

DEFINE FUNCTION return_lands and pass ‘lands’ as the parameter

INITIALIZE empty list returned_lands

SET total_amount AND total_fine TO 0

DISPLAY “Rented Lands:”

DISPLAY “Kitta No.\tCity/District\tDirection\tArea\tPrice\tStatus”

FOR EACH land IN lands:

IF land[‘status’] EQUALS ‘Not Available’:

DISPLAY IN FORMAT “kitta_number TAB city TAB direction TAB area TAB price TAB status”

WHILE True:

INPUT enter kitta_number or exit

IF kitta_number.lower() EQUALS ‘exit’:

BREAK

ELSE:

SET value of found_land TO False

FOR EACH land in lands:

IF land[‘kitta_number’] EQUALS parse_int(kitta_number)

SET value of found_land TO True

IF land[‘status’] EQUALS ‘Not Available’

FUNCTION datetime.datetime.now()

RETURN datetime

IF rent_datetime is LESS THAN 1

Rented_duartion EQUALS 1

CALCULATE Fine

IF fine_months LESS THAN 0:

Fine_price EQUALS CALCULATE fine_price

Total_fine += fine_price

Total_amount +=rented_duration*land[‘price’]

APPEND returned_lands

CHANGE land[‘status’] TO ‘Available’

DISPLAY “Land with {kitta number} successfully returned.”

OPEN invoice file in WRITE mode

WRITE ”Return invoice”

WRITE “Date and time of return:”

WRITE “Returned Lands:”

FOR EACH returned_land in returned_lands:

WRITE “kitta no., city, direction, area, total amount”

IF total_fine is GREATER THAN 0:

WRITE ”Total fine:”

WRITE “Total amount with fine:”

DISPLAY ‘Invoice generated!”

RETURN

ELSE:

DISPLAY ”This land was never rented”

Land[‘status’] EQUALS ‘Available’

RETURN

IF NOT found_land:

DISPLAY “This land does not exist in the system”

RETURN

2.4 Data Structures:

Data structures in programming relate to the organisation and administration of data in a computer's memory for easy access and modification. The basic python data structures include list, set, tuples and dictionary. Each data structure is unique in its own way. Data structures are the containers that organize data based on their types.

2.4.1 List:

List can be defined as an ordered collection of items. 'Ordered collections' means each item in a list comes with an order which uniquely identifies them. When creating a list, desired items of the list should be put inside of square brackets and separated by commas. A list can be nested which means it can contain any type of object. A list can also include another list within it. Lists are mutable meaning they can be altered even after being created.

Example: names = ['ram','shyam','hari']
'names' is the list with 'ram', 'shyam', 'hari' as its elements.

2.4.2 Tuple:

Tuple is a built-in Python data structure. It is an ordered collection of objects. Tuples come with limited functionality unlike lists. Tuples are ordered but not changeable, meaning we cannot add or remove items after the tuple has been created.

Example: Fruits_tuple=(“apple”,”mango”,”orange”)

2.4.3 Set:

A set us a collection that is unindexed, unordered, and unchangeable. Sets can be used to store multiple items in a single variable. As Sets are unordered, we don’t know which item will appear when it is printed. In sets, duplicates are not allowed meaning a set cannot have two items of the same value.

Example: set={“messi”,”ronaldo”,”hazard”}

2.4.4 Dictionary:

Dictionaries are another data structures in python which are used to store values in key: value pairs. Dictionaries are ordered in the latest versions(3.7) but were unordered prior that. Dictionaries are mutable, meaning values inside it can be changed but dictionaries do not allow duplicates. It is written with curly brackets, with keys and values inside.

Example: dictionary = {“Name” : ”Ram”} #(key : value)

2.4.5 Data Structure used in Course Work:

In this course work, Ilist, dictionaries and string object (Datetime) were used as the data structure.

Lists are used to manage collections of land information, including available, rented, and returned lands. Functions of list such as append is used which adds single item to certain collection types. Dictionaries store detailed information about each land, facilitating efficient retrieval and manipulation. Strings provide informative prompts and messages to users and format output for invoices. Datetime objects record rental and return timestamps, ensuring accurate tracking of transactions.

In main.py, list is initialized by calling a function which stores dictionary containing land information.

```
def main():
    """Description of the function:
    -Reads the textfile
    -available lands are displayed by calling the function from operations.py file
    -3 choices are given and user input is requested
    -choosing 1 leads to renting lands
    -choosing 2 leads to returning lands
    -choosing 3 exits the program

    """
    file_name = "land_info.txt"
    lands = read_land_data(file_name)
    while True:
        print("\nWelcome to TechnoPropertyNepal Land Rental System\n")
        display_available_lands(lands)
        print("\nMenu:")
```

Figure 2: Use of List in main.py

In operations.py, rented_lands list is initialized to store information of lands which have already been rented.

```
def rent_land(lands, customer_name, duration):
    """Description of the function:
    -lands, customer_name and duration are taken as the parameters
    -Available lands are shown and you need to input kitta number of the land
    -Land gets rented and you have the option to continue or stop
    -Invoice is generated
    """
    rented_lands = []
```

Figure 3: Use of list in operations.py

Also, returned_lands list in initialized in operations.py to store information about the lands which have been returned.

```
def return_land(lands):
    """Description of the function:
    -lands is taken as the parameter
    -rented lands are shown
    -input for land to be returned is asked and land is returned
    -total amount is calculated and invoice is generated
    -fine is added if land is returned late and invoice with total amount is generated
    """
    returned_lands = []
```

Figure 4: Use of list in operations.py

In main.py, each element in lands list is a dictionary containing information of lands.

```
def main():
    """Description of the function:
    -Reads the textfile
    -available lands are displayed by calling the function from operations.py file
    -3 choices are given and user input is requested
    -choosing 1 leads to renting lands
    -choosing 2 leads to returning lands
    -choosing 3 exits the program

    """
    file_name = "land_info.txt"
    lands = read_land_data(file_name)
```

Figure 5: Use of dictionary in main.py

3. Program:

3.1 Implementation of the program:

The objective of this coursework is to create a well-functioning land rental system. The coursework implements the concept of modularity to accomplish a working system. The program is made up of four separate files: main.py, read.py, write.py, and Operations.py. All the files are combined, and the program is made. Data from a text file (land_info.txt) is extracted by the program which contains information about the land including land id, direction faced, location, status, price and so on. With every transaction, the file is updated, and a bill receipt based on user's activities is produced which contains client's information and fine if the client failed to return the rented land on time. Exception handling is also implemented in the program.

After running the main.py file, you will be asked a question 'What are you here for?' and the options contain 1. To rent a land. 2. To return a rented land and 3. Exit. You will be able to rent any available land from the text file if you press 1. If you press 2, you will be asked to input some information about the land you rented, and bill will be generated based on your input. The availability status of the land is again updated after the rent/return of the land meaning if you rented a land located in Pokhara, the availability status will be updated from available to not available. And after returning the land, the availability status of the land is again updated from not available to available.

This coursework aims to build a land rental system helping us learn more about code modularity and making a pleasing program without the use of many in-built functions.

3.2 Process of Rent/Return of the land:

The process of rent/return of land begins right after the main file is ran. After running the file, a menu appears and there are 3 options. If you choose option 1, process of rent starts. First, the name of costumer is asked by the program. After the user provides the name, next step is carried out in which the program asks you to enter the duration of time in months you'd like to rent the available land/s for. After you enter the duration, the program again displays the available lands for rent, and you are asked to enter the Kitta Number/ Land ID of your desired land. After you enter the kitta number, the land gets rented and bill invoice is made in another text file. You will again get option to rent more land/s if you want to. If you type yes, same process repeats, and you will rent another land and generate invoice just like before. But if you type no, the availability status of the lands will be updated, making the land you just rented unavailable. Then, you will be taken to the main menu again and will have 3 options again: to rent land, return land or exit the program.

```
Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
101             Manang          North           5               30000           Available
102             Lalitpur                South           3               60000           Available
103             Pokhara         East            4               50000           Available
105             Bhaktapur               West            5               80000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice: 1
Enter customer name: ram
Enter duration of rent (in months): 1
Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status

101             Manang          North           5               30000           Available

102             Lalitpur                South           3               60000           Available

103             Pokhara         East            4               50000           Available

105             Bhaktapur               West            5               80000           Available

Enter the kitta number of the land you want to rent (or type 'exit' to cancel): 101
Land with kitta number 101 rented successfully.
Do you want to rent more land? (yes/no): no
Invoice generated!

Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
102             Lalitpur                South           3               60000           Available
103             Pokhara         East            4               50000           Available
105             Bhaktapur               West            5               80000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice:
```

Figure 6: Process of renting of land

After renting land/s, invoice is generated in a text file and the process is for renting is complete.

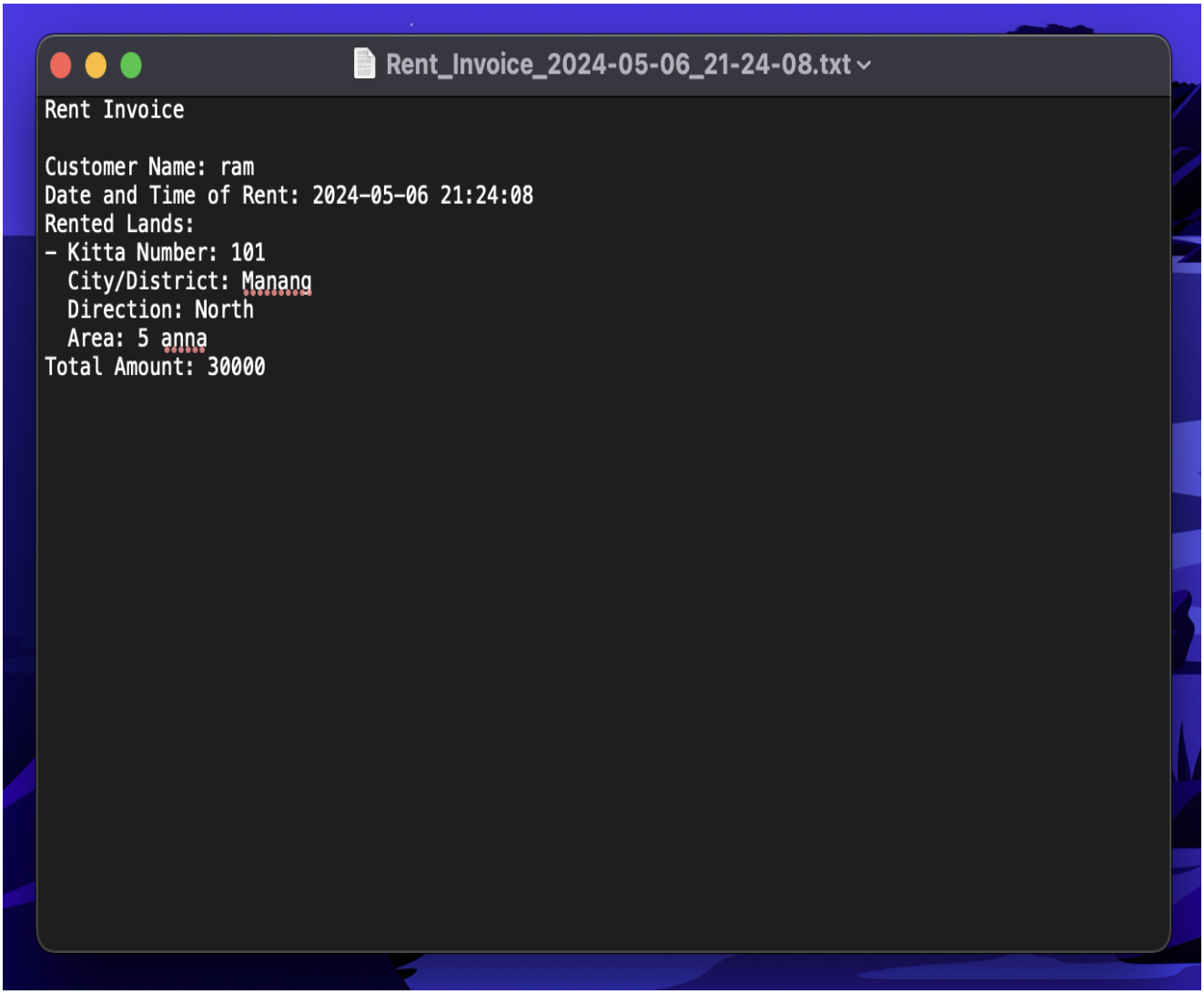

Figure 7: Invoice generation for rented land.

Similarly, for returning land/s, first, the land must be rented. After user selects 2 and proceeds, program displays the lands which were rented. Then, you will have to input the kitta number of the land you want to return. After doing so, the rented land will be returned and bill invoice will be generated on another text file containing the details such as time of rent, total amount and so on.

```
Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
102             Lalitpur                South           3               60000           Available
103             Pokhara         East            4               50000           Available
105             Bhaktapur               West            5               80000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice: 2
Rented Lands:
Kitta No.       City/District   Direction       Area    Price   Status
101             Manang          North           5               30000           Not Available
104             Kavre           South           10              80000           Not Available
Enter the kitta number of the land you want to return (or type 'exit' to cancel): 101
Land with kitta number 101 returned successfully.
Invoice generated!
```

Figure 8:Process of returning of land.

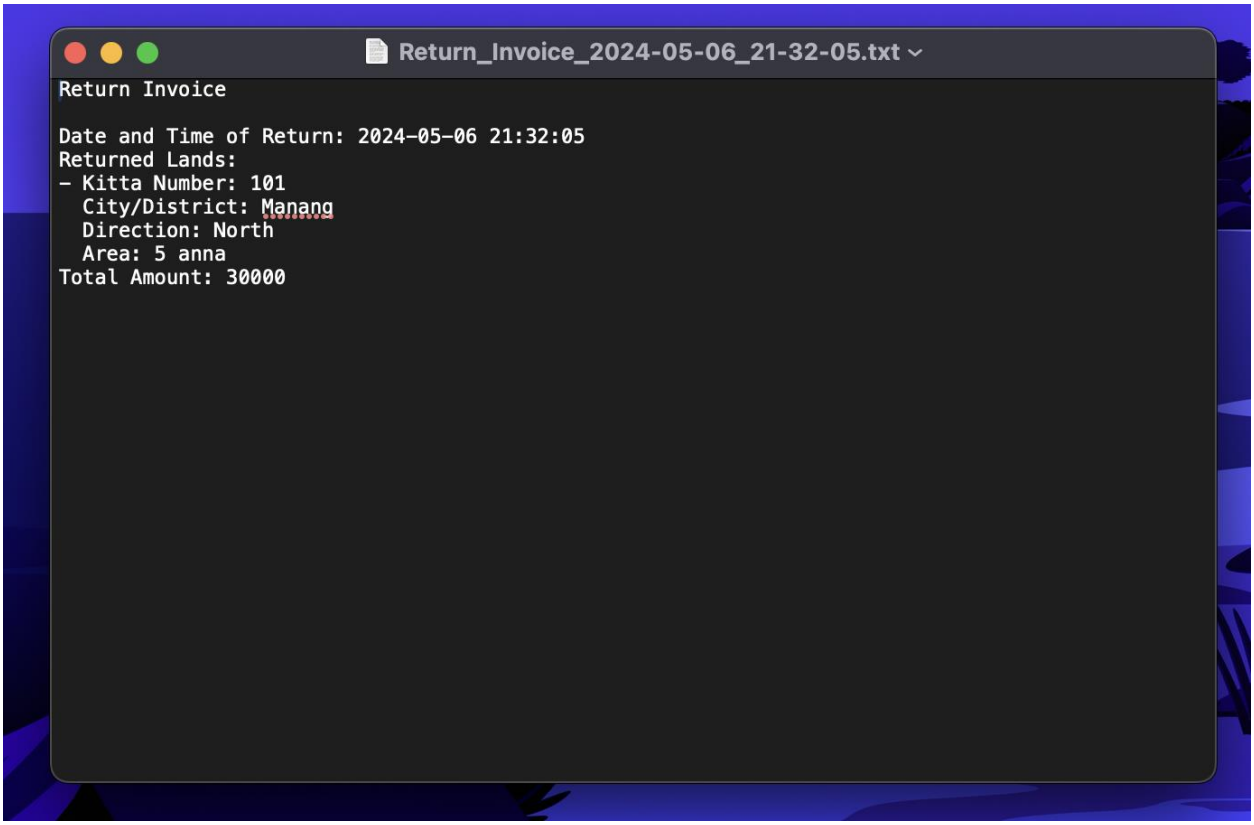

Figure 9: Invoice generation for returning land.

3.3 Creation of text file:

Text file including information such as kitta number of land, location of the land, direction faced, price, area of land and availability status was made by the use of TextEdit.

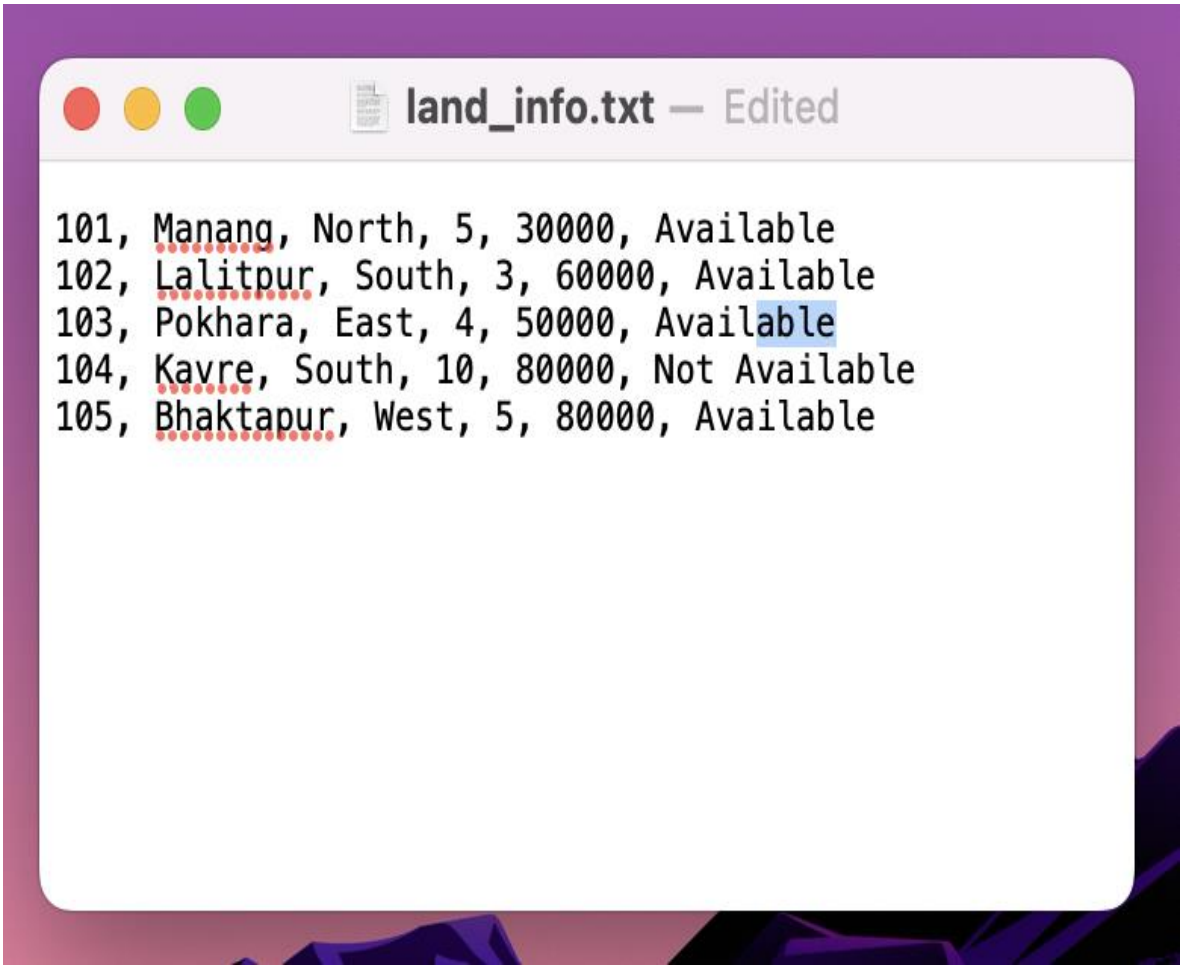

Figure 10: Creation of a Text File

3.4 Opening text and show the bill:

After getting the user input, certain process is carried out. If user decides to rent land and enter their name, duration and kitta number of the land they want to rent, the land gets rented successfully and a bill invoice is generated for the rented land. Similarly, if the user decides to return land, the kitta number of the land to be returned is asked and the land gets returned successfully. A bill invoice is generated for returned land in a separate text file.

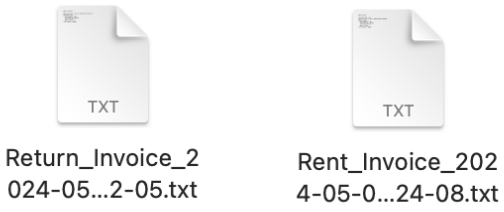

Figure 11: Generation of invoices.

After opening the invoice file, following result is displayed in a text file.

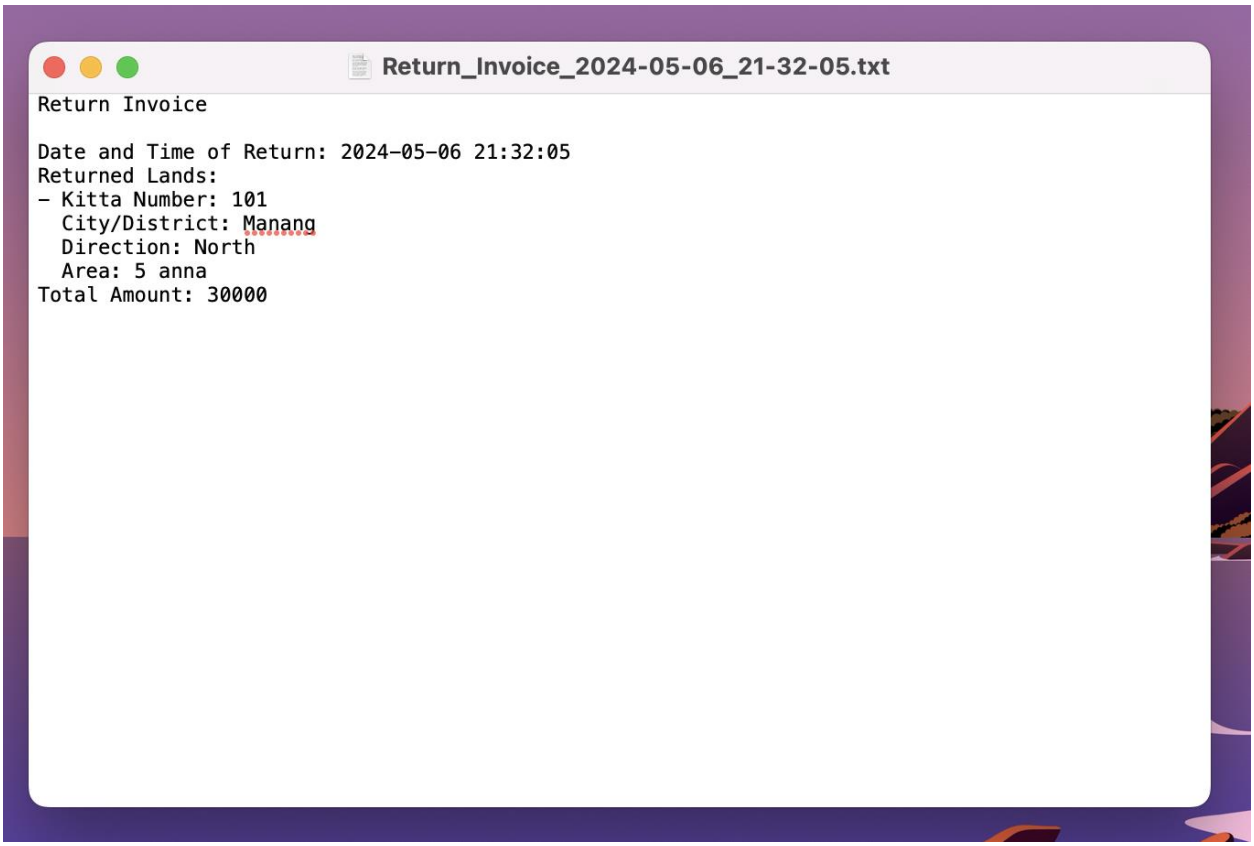

Figure 12: Opening the bill invoice.

3.5 Termination of the program:

In my code, I've included 3 options and corresponding methods are carried out based on which option the user chooses. After running the program, 3 choices appear. First one is to rent a land, second to return a rented land and, the third to terminate the program. If the user inputs '3', the operation of the third option 'Exit' is carried out and the program terminates.

```
Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
101             Manang          North           5               30000           Available
102             Lalitpur                South           3               60000           Available
103             Pokhara         East            4               50000           Available
105             Bhaktapur               West            5               80000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice: 3
>>>
>>>
>>>
>>>
>>>
>>>
```

Figure 13: Termination of the prog4. Testing:

4.1 Test 1: Implementation of Try, except:

Table 1: Test to show implementation of Try, except

Objective:	To show the implementation of try, except
Action:	-Provide the incorrect filename.
Expected Result:	-Exception occurs and it is caught by except block and the program continues.
Actual Result:	-The program did not terminate, code jumped to the corresponding except block.
Conclusion:	The test is successful.

The actual name of the text file which contains information about lands is “land_info.txt”. Let’s try using incorrect name and see what happens.

main.py - /Users/digdarshanbhattarai/Desktop/FoC CourseWork/main.py (3.12.2)

```
from read import read_land_data
from write import write_land_data
from operations import display_available_lands, rent_land, return_land

def main():
    """Description of the function:
    -Reads the textfile
    -available lands are displayed by calling the function from operations.py file
    -3 choices are given and user input is requested
    -choosing 1 leads to renting lands
    -choosing 2 leads to returning lands
    -choosing 3 exits the program

    """
    file_name = "land_infasfsdhgbrw.txt"
```

Figure 14: Giving incorrect filename.

Error is shown but the program is not terminated.

```
============================================================ RESTART: /Users/d
Error: File not found.

Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice:
```

Figure 15: Result by implementing try, except.

4.2 Test 2: Selection rent and return of lands.

Table 2: Test to show Selection rent and return of lands.

Objective:	Selection rent and return of lands.
Action:	-Provide negative value as input. -Provide non existing value as input.
Expected Result:	The entered value is not valid, and you are asked to enter again.
Actual Result:	You are asked to try again, and the program does not terminate.
Conclusion:	The test is successful.

```
Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice: 1
Enter customer name: ram
Enter duration of rent (in months): 2
Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status

101             Manang          North           5               30000           Available

102             Lalitpur                South           3               60000           Available

103             Pokhara         East            4               50000           Available

105             Bhaktapur               West            5               80000           Available

Enter the kitta number of the land you want to rent (or type 'exit' to cancel): -1
Invalid kitta number. Please try again.
Enter the kitta number of the land you want to rent (or type 'exit' to cancel):
```

Figure 16: Entering negative value

```
Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
101             Manang          North           5               30000           Available
102             Lalitpur                South           3               60000           Available
103             Pokhara         East            4               50000           Available
105             Bhaktapur               West            5               80000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice: 1
Enter customer name: digdarshan
Enter duration of rent (in months): 1
Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status

101             Manang          North           5               30000           Available

102             Lalitpur                South           3               60000           Available

103             Pokhara         East            4               50000           Available

105             Bhaktapur               West            5               80000           Available

Enter the kitta number of the land you want to rent (or type 'exit' to cancel): 109
Invalid kitta number. Please try again.
Enter the kitta number of the land you want to rent (or type 'exit' to cancel):
```

Figure 17:Entering non-existing value.

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

Table 3: Test to show File generation of renting of land(s) (Renting multiple land(s))

Objective:	File generation of renting of land(s) (Renting multiple land(s))
Action:	-Multiple lands are rented.
Expected Result:	-Bills for multiple lands should be made.
Actual Result:	-Bills are generated for multiple lands.
Conclusion:	The test is successful.

Complete renting process of the land:

```
Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
101             Manang          North           5               30000           Available
102             Lalitpur                South           3               60000           Available
103             Pokhara         East            4               50000           Available
105             Bhaktapur               West            5               80000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice: 1
Enter customer name: hari
Enter duration of rent (in months): 3
Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status

101             Manang          North           5               30000           Available

102             Lalitpur                South           3               60000           Available

103             Pokhara         East            4               50000           Available

105             Bhaktapur               West            5               80000           Available

Enter the kitta number of the land you want to rent (or type 'exit' to cancel): 101
Land with kitta number 101 rented successfully.
Do you want to rent more land? (yes/no): yes
Enter the kitta number of the land you want to rent (or type 'exit' to cancel): 102
Land with kitta number 102 rented successfully.
Do you want to rent more land? (yes/no): yes
Enter the kitta number of the land you want to rent (or type 'exit' to cancel): 105
Land with kitta number 105 rented successfully.
Do you want to rent more land? (yes/no): no
Invoice generated!

Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
103             Pokhara         East            4               50000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice:
```

Figure 18: Complete renting process of the land(multiple)

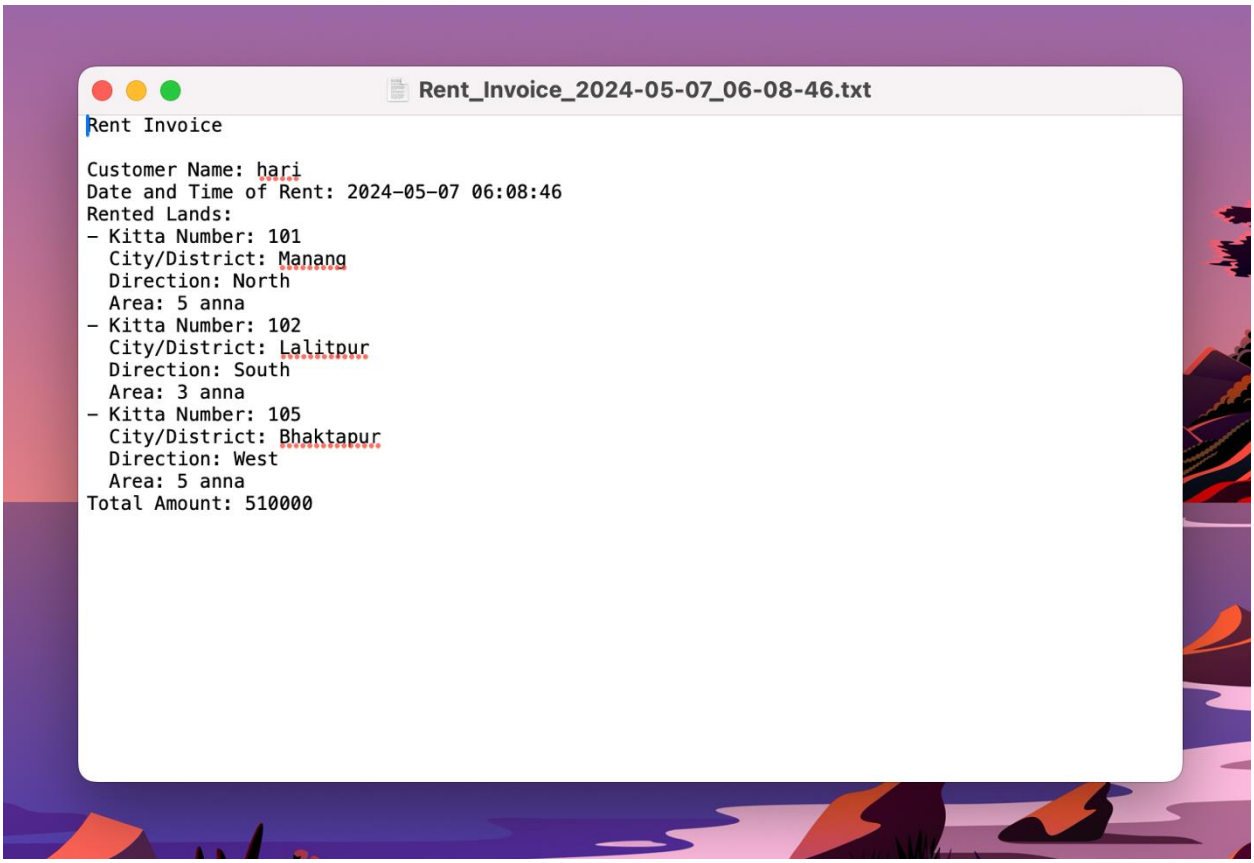

Figure 19: Invoice generation after renting multiple lands.

4.4 Test 4: Test to show File generation of returning process of land(s) (Returning multiple land(s))

Table 4: Test to show File generation of returning process of land(s) (Returning multiple land(s))

Objective:	Test to show File generation of returning process of land(s) (Returning multiple land(s))
Action:	-Take user input to return land. -Ask the kitta number to be returned.
Expected Result:	-Entered kitta number is returned and status of land is updated in the shell.
Actual Result:	-Entered kitta number is returned and status of land is updated in the shell.
Conclusion:	Test is successful.

```
Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
103             Pokhara         East            4               50000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice: 2
Rented Lands:
Kitta No.       City/District   Direction       Area    Price   Status
101             Manang          North           5               30000           Not Available
102             Lalitpur                South           3               60000           Not Available
104             Kavre           South           10              80000           Not Available
105             Bhaktapur               West            5               80000           Not Available
Enter the kitta number of the land you want to return (or type 'exit' to cancel): 105
Land with kitta number 105 returned successfully.
Invoice generated!

Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
103             Pokhara         East            4               50000           Available
105             Bhaktapur               West            5               80000           Available
```

Figure 20: Returning process of rented lands.

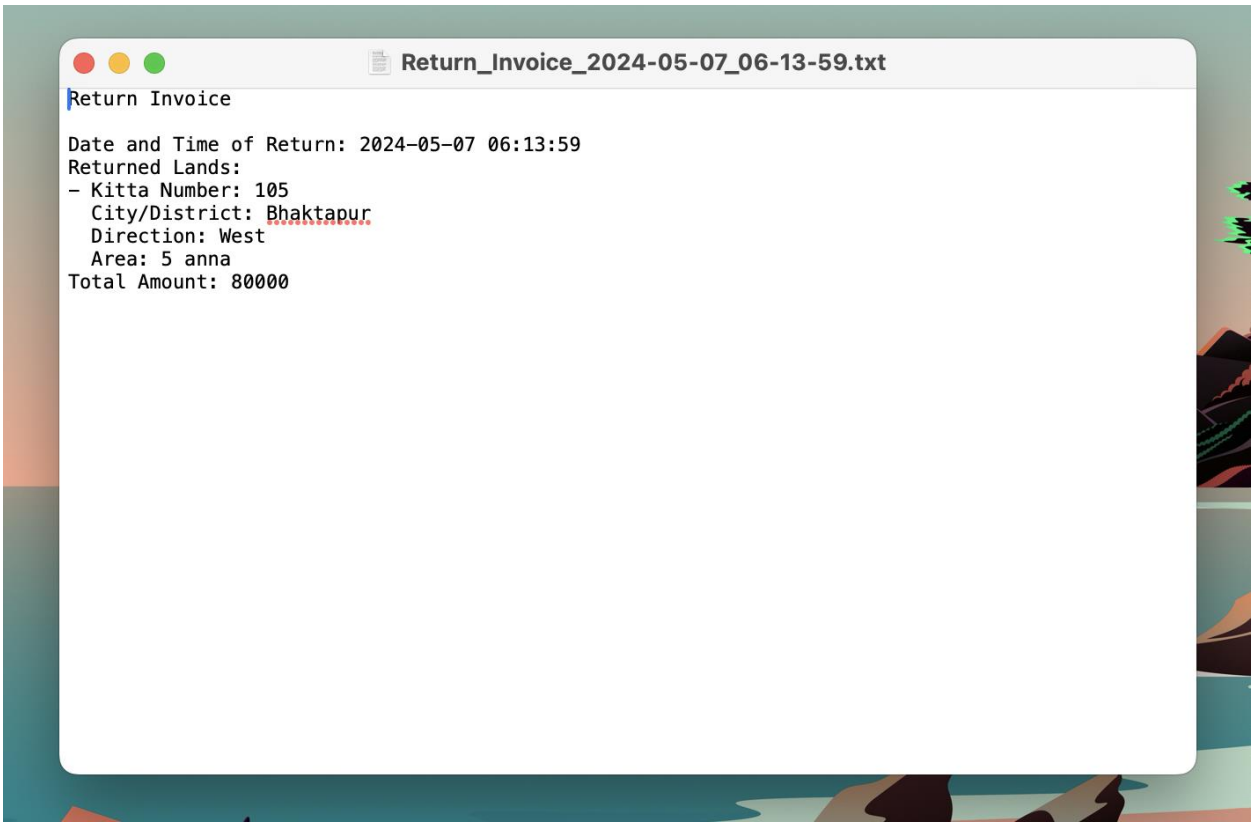

Figure 21: Invoice details of returned land in a text file.

4.5 Test 5: Test to Show the update in stock of land(s)

Table 5: Test to show the update in stock of land(s)

Objective:	Show the update in stock of land(s)
Action:	-Firstly, land is rented so that availability status changes from available to not available. -Land is then returned and observations are made.
Expected Result:	-When land is rented, status needs to be updated to not available and when it is returned, the status needs to be updated back to available once again.
Actual Result:	-Stock of land is updated.
Conclusion:	The test is successful.

```
Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
101             Manang          North           5               30000           Available
102             Lalitpur                South           3               60000           Available
103             Pokhara         East            4               50000           Available
105             Bhaktapur               West            5               80000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice: 1
Enter customer name: john
Enter duration of rent (in months): 5
Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status

101             Manang          North           5               30000           Available

102             Lalitpur                South           3               60000           Available

103             Pokhara         East            4               50000           Available

105             Bhaktapur               West            5               80000           Available

Enter the kitta number of the land you want to rent (or type 'exit' to cancel):
```

Figure 22:Showing the available lands to rent.

```
Enter the kitta number of the land you want to rent (or type 'exit' to cancel): 103
Land with kitta number 103 rented successfully.
Do you want to rent more land? (yes/no): no
Invoice generated!

Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
101             Manang          North           5               30000           Available
102             Lalitpur                South           3               60000           Available
105             Bhaktapur               West            5               80000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice:
```

Figure 23: Updated stock after renting process.

```
Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice: 2
Rented Lands:
Kitta No.       City/District   Direction       Area    Price   Status
103             Pokhara         East            4               50000           Not Available
104             Kavre           South           10              80000           Not Available
Enter the kitta number of the land you want to return (or type 'exit' to cancel):
```

Figure 24: Stock of rented lands.

```
Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice: 2
Rented Lands:
Kitta No.       City/District   Direction       Area    Price   Status
103             Pokhara         East            4               50000           Not Available
104             Kavre           South           10              80000           Not Available
Enter the kitta number of the land you want to return (or type 'exit' to cancel): 103
Land with kitta number 103 returned successfully.
Invoice generated!

Welcome to TechnoPropertyNepal Land Rental System

Available Lands:

Kitta No.       City/District   Direction       Area    Price   Status
101             Manang          North           5               30000           Available
102             Lalitpur                South           3               60000           Available
103             Pokhara         East            4               50000           Available
105             Bhaktapur               West            5               80000           Available

Menu:
1. Rent Land
2. Return Land
3. Exit
Enter your choice:
```

Figure 25: Updated stock after returning a rented land

5. Conclusion:

5.1 Evaluation of the coursework:

In this coursework, I had to develop a land rental system using python's IDLE which performs tasks, renting land/s and returning the rented land/s. After a client rents a land, a bill invoice is generated in a text file including information about the land that the client rented, the time of rent, and the total price. Similar invoice is generated when a client returns rented land/s. If a client fails to return the land on time, fine is charged which is added to the total sum. This project helped me learn more about programming. It helped me get ideas about things like modularity which helps to make the code maintainable, organized and reusable. This project helped me understand file handling in python. Opening a file, reading, and writing on the text file are the things I did in this project. Also, I used list and dictionary as my data structure to work with data in the program.

So overall, this coursework provided me a great learning experience which also comes handy in real world scenarios ultimately boosting my skillset, teaching me things I did not know how to do before. I am satisfied with the result of my program and feel like I have learned and grown in some positive manner.

5.2 Learning from the coursework

From this coursework I got the opportunity to develop knowledge on file handling in python, data structures and manipulating them, making multiple modules so that the program can be easy to handle and much more understandable, defining functions and calling them when required and so much more. The documentation also helped me learn how to write a good report of the task I did. I believe it improved my researching abilities as I had to look up for references by searching through the web. Through this work I've been able to learn how to present my work professionally and neatly. Additionally,

I learned to develop proper algorithm, flowchart and, pseudocodes for my program. Besides that, the task has helped me in my personal life too as time management was necessary while doing the coursework. It helped me spend my time wisely and in a disciplined way. Altogether, this coursework was of tremendous benefit for my skill development and was really a valuable learning experience.

5.3 Challenges and Solution:

While doing the coursework, I faced some challenges which helped me develop my learning and researching habit. I believe it has helped improve my problem-solving skills. I'd like to express my gratitude towards Mr. Bijaya Gautam Sir who not only provided us just the lecture slides and reference materials, but also recorded videos to tackle the

common problems students face. Through those resources I was able to gain ideas about the project and execute them while doing it. I looked up the lecture slides and other reference materials first and applied that knowledge. I looked up tutorials on YouTube to learn things in depth. I learned about exception handling through videos and implemented it in my code. The whole process was didactic and enjoyable at the same time.

6. References:

Bibliography

Computer Hope, 2024. *Home Page.* [Online] Available at: https://www.computerhope.com/jargon/m/microsoft-word.htm [Accessed 2 May 2024].

FOTC, 2023. *Blog.* [Online] Available at: https://fotc.com/blog/draw-io-online-guide/#:~:text=What%20is%20Draw.io%3F,fully%20integrated%20with%20Google%20Drive. [Accessed 2 May 2024].

GeeksForGeeks, Oct, 2023. *Blog.* [Online] Available at: https://www.geeksforgeeks.org/an-introduction-to-flowcharts/ [Accessed 2 May 2024].

Gillis, A. S., July, 2023. *Home page.* [Online] Available at: https://www.techtarget.com/whatis/definition/algorithm#:~:text=An%20algorithm%20is%20a%20procedure,%2D%20or%20software%2Dbased%20routines. [Accessed 2 May 2024].

Mahr, N., 2023. *Business Courses/Course.* [Online] Available at: https://study.com/learn/lesson/pseudocode-examples-what-is-

pseudocode.html

[Accessed 2 May 2024].

Python, 2024. *Python*. [Online] Available at: https://docs.python.org/3/library/idle.html

[Accessed 2 May 2024].

7. Appendix:

7.1 Appendix for Main.py:

```
from read import read_land_data
from write import write_land_data
from operations import display_available_lands, rent_land, return_land

def main():
    """Description of the function:
    -Reads the textfile
    -available lands are displayed by calling the function from operations.py file
    -3 choices are given and user input is requested
    -choosing 1 leads to renting lands
    -choosing 2 leads to returning lands
    -choosing 3 exits the program
    """
    file_name = "land_info.txt"
    lands = read_land_data(file_name)
    while True:
        print("\nWelcome to TechnoPropertyNepal Land Rental System\n")
        display_available_lands(lands)
        print("\nMenu:")
        print("1. Rent Land")
        print("2. Return Land")
```

```
        print("3. Exit")
        choice = input("Enter your choice: ")



    try:
            if choice == '1':
                customer_name = input("Enter customer name: ")
                duration = int(input("Enter duration of rent (in months): "))
                #After entering name and duration of rent, rent_land function from operations
is called.
                rent_land(lands, customer_name, duration)
            elif choice == '2':
                #return_land function is called if 2 is selected.
                return_land(lands)
            elif choice == '3':
                write_land_data(file_name, lands)
                break
            else:
                print("Invalid choice. Please enter a number between 1 and 3.")
        except ValueError:
            print("Invalid input. Please enter a valid number.")

if __name__ == "__main__":
    main()
```

7.2 Appendix for Operations.py:

```
import datetime

def display_available_lands(lands):
    """This function displays available lands from the text file"""
    print("Available Lands:\n")
    print("Kitta No.\tCity/District\tDirection\tArea\tPrice\tStatus")
    for land in lands:
        if land['status'] == 'Available':

print(f"{land['kitta_number']}\t\t{land['city']}\t\t{land['direction']}\t\t{land['area']}\t\t{land['pric
e']}\t\t{land['status']}")

def rent_land(lands, customer_name, duration):
    """Description of the function:
    -lands, customer_name and duration are taken as the parameters
    -Available lands are shown and you need to input kitta number of the land
    -Land gets rented and you have the option to continue or stop
    -Invoice is generated
    """
    rented_lands = []
    total_amount = 0
    print("Available Lands:\n")
    print("Kitta No.\tCity/District\tDirection\tArea\tPrice\tStatus\n")
    for land in lands:
```

```
        if land['status'] == 'Available':

print(f"{land['kitta_number']}\t\t{land['city']}\t\t{land['direction']}\t\t{land['area']}\t\t{land['price']}\t\t{land['status']}\n")
    while True:
        kitta_number = input("Enter the kitta number of the land you want to rent (or type 'exit' to cancel): ")
        if kitta_number.lower() == 'exit':
            break
        else:
            for land in lands:
                if land['kitta_number'] == int(kitta_number) and land['status'] == 'Available':
                    rented_duration = duration
                    current_datetime = datetime.datetime.now()
                    rent_datetime = current_datetime.strftime('%Y-%m-%d %H:%M:%S')
                    total_amount += rented_duration * land['price']
                    land['status'] = 'Not Available'
                    land['rent_date'] = rent_datetime
                    rented_lands.append(land)
                    print(f"Land with kitta number {kitta_number} rented successfully.")
                    more_lands = input("Do you want to rent more land? (yes/no): ")
                    if more_lands.lower() == 'no':
                        techno_prop_invoice_name = f"Rent_Invoice_{current_datetime.strftime('%Y-%m-%d_%H-%M-%S')}.txt"
                        with open(techno_prop_invoice_name, 'w') as techno_prop_invoice:
                            techno_prop_invoice.write("Rent Invoice\n\n")
                            techno_prop_invoice.write(f"Customer Name: {customer_name}\n")
                            techno_prop_invoice.write(f"Date and Time of Rent: {rent_datetime}\n")
```

```
                    techno_prop_invoice.write("Rented Lands:\n")
                    for land in rented_lands:
                        techno_prop_invoice.write(f"-Kitta Number: {land['kitta_number']}\n")
                        techno_prop_invoice.write(f"  City/District: {land['city']}\n")
                        techno_prop_invoice.write(f"  Direction: {land['direction']}\n")
                        techno_prop_invoice.write(f"  Area: {land['area']} anna\n")
                    techno_prop_invoice.write(f"Total Amount: {total_amount}\n")
                print("Invoice generated!")
                return
            elif more_lands.lower() == 'yes':
                break
        else:
            print("Invalid kitta number. Please try again.")


def return_land(lands):
    """Description of the function:
    -lands is taken as the parameter
    -rented lands are shown
    -input for land to be returned is asked and land is returned
    -total amount is calculated and invoice is generated
    -fine is added if land is returned late and invoice with total amount is generated
    """
    returned_lands = []
    total_amount = 0
    total_fine = 0
    print("Rented Lands:")
    print("Kitta No.\tCity/District\tDirection\tArea\tPrice\tStatus")
```

```
    for land in lands:
        if land['status'] == 'Not Available':

print(f"{land['kitta_number']}\t\t{land['city']}\t\t{land['direction']}\t\t{land['area']}\t\t{land['price']}\t\t{land['status']}")
    while True:
        kitta_number = input("Enter the kitta number of the land you want to return (or type 'exit' to cancel): ")
        if kitta_number.lower() == 'exit':
            break
        else:
            found_land = False
            for land in lands:
                if land['kitta_number'] == int(kitta_number):
                    found_land = True
                    if land['status'] == 'Not Available':
                        current_datetime = datetime.datetime.now()
                        return_datetime = current_datetime.strftime('%Y-%m-%d %H:%M:%S')
                        rent_datetime = land.get('rent_date')
                        if rent_datetime:
                            rent_date = datetime.datetime.strptime(rent_datetime, '%Y-%m-%d %H:%M:%S')
                            rented_duration = (current_datetime.year - rent_date.year) * 12 + current_datetime.month - rent_date.month
                            if rented_duration < 1:
                                rented_duration = 1
                            fine_months = (current_datetime.year - rent_date.year) * 12 + current_datetime.month - rent_date.month - rented_duration
```

```
                if fine_months > 0:
                    fine_price = 0.1 * fine_months * land['price']
                    total_fine += fine_price
                total_amount += rented_duration * land['price']
                returned_lands.append(land)
            land['status'] = 'Available'
            print(f"Land with kitta number {kitta_number} returned successfully.")

techno_prop_invoice_name=f"Return_Invoice_{current_datetime.strftime('%Y-%m-
%d_%H-%M-%S')}.txt"
            with open(techno_prop_invoice_name, 'w') as techno_prop_invoice:
                techno_prop_invoice.write("Return Invoice\n\n")
                techno_prop_invoice.write(f"Date and Time of Return:
{return_datetime}\n")
                techno_prop_invoice.write("Returned Lands:\n")
                for land in returned_lands:
                    techno_prop_invoice.write(f"- Kitta Number:
{land['kitta_number']}\n")
                    techno_prop_invoice.write(f"  City/District: {land['city']}\n")
                    techno_prop_invoice.write(f"  Direction: {land['direction']}\n")
                    techno_prop_invoice.write(f"  Area: {land['area']} anna\n")
                techno_prop_invoice.write(f"Total Amount: {total_amount}\n")
                if total_fine > 0:
                    techno_prop_invoice.write(f"Late Return Fine: {total_fine}\n")
                    techno_prop_invoice.write(f"Amount with Fine: {total_amount +
total_fine}\n")
            print("Invoice generated!")
            return
```

```
            else:
                print("This land was never rented.")
                land['status'] = 'Available'
                return
        if not found_land:
            print("This land does not exist in the system.")
            return
```

7.3 Appendix for Read.py:

```
def read_land_data(file_name):
    """
    This function reads the text file and returns a list of dictionary
    that contains information about lands.
    """

    land_data = []
    try:
        with open(file_name, 'r') as file:
            for line in file:
                land = line.strip().split(',')
                land_data.append({
                    'kitta_number': int(land[0]),
                    'city': land[1].strip(),
                    'direction': land[2].strip(),
                    'area': int(land[3]),
                    'price': int(land[4]),
                    'status': land[5].strip()
                })
    except FileNotFoundError:
        print("Error: File not found.")
    except Exception as e:
        print(f"An error occurred: {str(e)}")
    return land_dat
```

7.4 Appendix for write.py:

```
def write_land_data(file_name, land_data):
    """
    -file_name and land_data are passed as parameters
    -file_name is opened in write mode and land data will be written to the file_name
    -handels exception which might occur during file handling
    """
    try:
        with open(file_name, 'w') as file:
            for land in land_data:

file.write(f"{land['kitta_number']},{land['city']},{land['direction']},{land['area']},{land['price']},
{land['status']}\n")
    except Exception as e:
        print(f"An error occurred: {str(e)}")
```