



**CS4051NI Fundamentals of Computing**

**60% Individual Coursework**

**2023/24 Spring**

**Student Name: Anugraha Maharjan**

**London Met ID: 23047651**

**College ID: np01cp4a230070**

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

**Assignment Submission Date: Monday, May 6, 2024**

**Word Count: 8774**

**Project File Links:**

|  |  |
| --- | --- |
| **YouTube Link:** | Keep Unlisted YouTube URL of your Project Here |
| **Google Drive Link:** | **https://drive.google.com/drive/folders/1nhiQHyxYJ0TPpwObD0-OFoDlG7cyWMuc?usp=drive\_link** |

*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](#_Toc165931145)

[1.1. Tools Used 2](#_Toc165931146)

[Python 2](#_Toc165931147)

[IDLE 2](#_Toc165931148)

[1.2. Goals 2](#_Toc165931149)

[1.3. Objectives 2](#_Toc165931150)

[2. Discussion and Analysis 3](#_Toc165931151)

[2.1. Algorithm 3](#_Toc165931152)

[2.2. Flowchart 6](#_Toc165931153)

[2.3. Pseudocode 11](#_Toc165931154)

[2.3.1. Main.py 11](#_Toc165931155)

[2.3.2. Read.py 18](#_Toc165931156)

[2.3.3. Operations.py 18](#_Toc165931157)

[2.3.4. Write.py 24](#_Toc165931158)

[2.4. Data Structure 29](#_Toc165931159)

[2.4.1. Primitive data structure: 29](#_Toc165931160)

[2.4.2. Collection data structure: 30](#_Toc165931161)

[3. Program 31](#_Toc165931162)

[4. Testing 51](#_Toc165931163)

[4.1 Test 1: Implementation of try and except 51](#_Toc165931164)

[4.2 Test 2: Invalid input while Rent and Return of land 52](#_Toc165931165)

[4.2.1. Test 2.1: Providing invalid value while renting lands 52](#_Toc165931166)

[4.2.2. Test 2.2: Providing invalid value while returning lands 54](#_Toc165931167)

[4.3 Test 3: Invoice generation after renting multiple lands 56](#_Toc165931168)

[4.4 Test 4: Invoice generation after returning multiple lands 61](#_Toc165931169)

[4.5 Test 5: Update of Availability of Lands 67](#_Toc165931170)

[4.5.1. Test 5.1: After renting land 67](#_Toc165931171)

[4.5.2. Test 5.2: After returning land 70](#_Toc165931172)

[5. Conclusion 73](#_Toc165931173)

[6. Bibliography 74](#_Toc165931174)

[7. Appendix 76](#_Toc165931175)

[7.1. main.py 76](#_Toc165931176)

[7.2. read.py 82](#_Toc165931177)

[7.3. operations.py 83](#_Toc165931178)

[7.4. write.py 87](#_Toc165931179)

**List of Figures**

[Figure 1: Python logo 2](file:///C:\Users\hp\Desktop\College\Second%20sem\Fundamentals%20of%20computing\Coursework\23047651%20Anugraha%20Maharjan.docx#_Toc165931069)

[Figure 2: IDLE logo 2](file:///C:\Users\hp\Desktop\College\Second%20sem\Fundamentals%20of%20computing\Coursework\23047651%20Anugraha%20Maharjan.docx#_Toc165931070)

[Figure 3: Flowchart of system (1) 6](#_Toc165931071)

[Figure 4 : Flowchart of system (2) 7](#_Toc165931072)

[Figure 5: Flowchart of system (3) 8](#_Toc165931073)

[Figure 6: Flowchart of system (4) 9](#_Toc165931074)

[Figure 7: Flowchart of system (5) 10](#_Toc165931075)

[Figure 8: Screenshot of int data type 29](#_Toc165931076)

[Figure 9: Screenshot of boolean data type 29](#_Toc165931077)

[Figure 10: Screenshot of string data type 29](#_Toc165931078)

[Figure 11: Screenshot of list 30](#_Toc165931079)

[Figure 12: Screenshot of start of program 31](#_Toc165931080)

[Figure 13: Screenshot of error message after invalid input in main menu 31](#_Toc165931081)

[Figure 14: Screenshot of selecting view all rental properties in main menu 32](#_Toc165931082)

[Figure 15: Screenshot of selecting rent a land in main menu 32](#_Toc165931083)

[Figure 16: Screenshot of error message after entering invalid kitta while renting 32](#_Toc165931084)

[Figure 17: Screenshot of exiting from renting land 33](#_Toc165931085)

[Figure 18: Screenshot of entering valid kitta while renting land 33](#_Toc165931086)

[Figure 19: Screenshot of asking user duration of rent 34](#_Toc165931087)

[Figure 20: Screenshot of error message after entering invalid duration of rent 34](#_Toc165931088)

[Figure 21: Screenshot after entering valid duration of rent 35](#_Toc165931089)

[Figure 22: Screenshot of bill printed after renting one land 35](#_Toc165931090)

[Figure 23: Screenshot of bill generated in .txt file after renting one land 36](#_Toc165931091)

[Figure 24: Screenshot after selecting rent more land option 36](#_Toc165931092)

[Figure 25: Screenshot of error message after invalid kitta while renting more land 37](#_Toc165931093)

[Figure 26: Screenshot after entering valid kitta 37](#_Toc165931094)

[Figure 27: Screenshot of error message after entering invalid duration while renting more land 38](#_Toc165931095)

[Figure 28: Screenshot after entering valid duration while renting more land 38](#_Toc165931096)

[Figure 29: Screenshot of combined bill printed after renting multiple land 39](#_Toc165931097)

[Figure 30: Screenshot of combined bill .txt file created after renting multiple land 40](#_Toc165931098)

[Figure 31: Screenshot of choosing return land option in main menu 40](#_Toc165931099)

[Figure 32: Screenshot of error message after entering invalid kitta while returning land 41](#_Toc165931100)

[Figure 33: Screenshot after entering valid kitta while returning land 41](#_Toc165931101)

[Figure 34: Screenshot after entering name while returning land 42](#_Toc165931102)

[Figure 35: Screenshot after entering invalid duration as per contract while returning land 42](#_Toc165931103)

[Figure 36: Screenshot after entering valid duration as per contract while returning land 42](#_Toc165931104)

[Figure 37: Screenshot after entering invalid actual duration while returning land 43](#_Toc165931105)

[Figure 38: Screenshot after entering valid actual duration while returning land 43](#_Toc165931106)

[Figure 39: Screenshot of bill printed after returning one land 44](#_Toc165931107)

[Figure 40: Screenshot of .txt file generated after returning one land 45](#_Toc165931108)

[Figure 41: Screenshot after choosing to return more land 45](#_Toc165931109)

[Figure 42: Screenshot after completing all fields while returning multiple land 46](#_Toc165931110)

[Figure 43: Screenshot of bill printed after returning multiple land 47](#_Toc165931111)

[Figure 44: Screenshot of status of lands changed after returning 48](#_Toc165931112)

[Figure 45: Screenshot of .txt file generated after returning multiple lands 49](#_Toc165931113)

[Figure 46: Screenshot after entering invalid reply in main menu 50](#_Toc165931114)

[Figure 47: Screenshot after choosing to exit in main menu 50](#_Toc165931115)

[Figure 48: Screenshot of try except test 51](#_Toc165931116)

[Figure 49: Screenshot of code of try except 51](#_Toc165931117)

[Figure 50: Screenshot of providing negative value while renting 52](#_Toc165931118)

[Figure 51: Screenshot of providing non existing value while renting 53](#_Toc165931119)

[Figure 52: Screenshot of code of checking invalid input 53](#_Toc165931120)

[Figure 53: Screenshot of providing negative value while returning land 54](#_Toc165931121)

[Figure 54: Screenshot of providing non existing value while returning land 55](#_Toc165931122)

[Figure 55: Screenshot of code to check invalid input 55](#_Toc165931123)

[Figure 56: Screenshot of renting land one 56](#_Toc165931124)

[Figure 57: Screenshot of renting multiple land 57](#_Toc165931125)

[Figure 58: Screenshot of bill printed after renting multiple land 57](#_Toc165931126)

[Figure 59: Screenshot of code that prints bill while renting multiple land 58](#_Toc165931127)

[Figure 60: Screenshot of .txt file generated after renting multiple lands 59](#_Toc165931128)

[Figure 61: Screenshot of code that generate .txt file after renting multiple land 60](#_Toc165931129)

[Figure 62: Screenshot of returning one land 61](#_Toc165931130)

[Figure 63: Screenshot of returning multiple land 62](#_Toc165931131)

[Figure 64: Screenshot of bill printed after returning multiple land 63](#_Toc165931132)

[Figure 65: Screenshot of code that print bill after returning multiple land 64](#_Toc165931133)

[Figure 66: Screenshot of .txt file generated after returning multiple land 65](#_Toc165931134)

[Figure 67: Screenshot of code that generate .txt file after returning multiple land 66](#_Toc165931135)

[Figure 68: Screenshot of renting a land 67](#_Toc165931136)

[Figure 69: Screenshot of status changed after renting a land 68](#_Toc165931137)

[Figure 70: Screenshot of status changed after renting a land in .txt file 68](#_Toc165931138)

[Figure 71: Screenshot of code that change status after renting a land 69](#_Toc165931139)

[Figure 72: Screenshot of returning a land 70](#_Toc165931140)

[Figure 73: Screenshot of status changed after returning a land 71](#_Toc165931141)

[Figure 74: Screenshot of status changed after returning land in .txt file 71](#_Toc165931142)

[Figure 75: Screenshot of code to change status after returning land 72](#_Toc165931143)

**List of Tables**

[Table 1: Implementation of try and except test 51](#_Toc165931183)

[Table 2: Invalid value input while renting test 52](#_Toc165931184)

[Table 3: Invalid value input while returning test 54](#_Toc165931185)

[Table 4: Invoice generation after renting multiple lands test 56](#_Toc165931186)

[Table 5: Invoice generation after returning multiple lands test 61](#_Toc165931187)

[Table 6: Update status after renting land test 67](#_Toc165931188)

[Table 7: Update status after returning land test 70](#_Toc165931189)

# Introduction

With the advancement of technology and increased availability of computers, people wish to be able to do everything with just a click of button on screen. Thus, this report is about development of Land Rental System using Python can replace traditional way of renting lands.

The system will consist of 4 main functionalities, view all rental properties, rent a land, return rented land and exit the system. For the development of this system, I will be using procedural-based approach with python programming language. I am going to use functions for major operations so that the code can be reused. For every land rented, a bill will be generated that will always have a unique txt file name and an invoice will be generated for every land returned as well. If the land returned has exceeded contract duration, fine will also be calculated. Try, except will also be used for exception handling.

This system will also contain 4 main modules, ‘main.py’, ‘read.py’, ‘operations.py’ and ‘write.py’. Each module, as name suggest will contain code only in same field. Read module will contain code only of reading a file, operation about operations only and write about writing in file only. The main module contains the main logic of the system where all functions are called from.

The system will be started by running the main.py module. Firstly, the information about the company will be displayed with the 4 options. Using the user’s reply, the system will run different functions according to user’s need. The system will not come to and end without the user exiting from the main menu. The program will only end when the user wishes to. For, this all the exceptions will need to be handled. Different types of tests will also be performed for the system to execute as intended.

## Tools Used

### Python

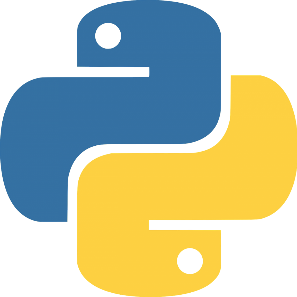
Python is an object-oriented, interpreted, and high-level programming language (Geeks for Geeks, 2024). Programming languages like Python are well-liked. Released in 1991, it was designed by Guido van Rossum (w3schools, 2024). It is a popular computer programming language used for creating software and websites, task automation, and data analysis (Coursera, 2024).

Figure 1: Python logo

### IDLE

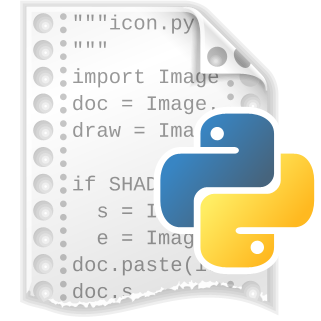
IDLE stands for Integrated Development and Learning Environment. It is used for python programming. With capabilities like syntax highlighting, autocompletion, and smart indent, IDLE offers a feature-rich text editor (Educative, 2024). Its user-friendly interface makes Python development accessible to programmers of all skill levels, be they beginner or expert (Higssoftware, 2024).

Figure 2: IDLE logo

## Goals

The main goal of this project is to create an efficient Land Rental System which can perform all task such as view, rent and return lands. The system must also generate an invoice with each transaction performed.

## Objectives

* Design the logic of system with the help of flowchart and algorithm
* Separation of code into 4 modules on the basis of use i.e, main, read, write and

operations

* Use of function for code reusability
* Creation of .txt file with each transaction
* Perform different test for proper functioning of the system

# Discussion and Analysis

## 2.1. Algorithm

**Step 1:** Start

**Step 2:** Display Press 1 to view all properties, Press 2 to rent land, Press 3 to return land, Press 4 to exit system. What would you like to do?

**Step 3:** Input reply

**Step 4:** If reply is an integer, go to step 6 else go to step 5

**Step 5:** Print Invalid format, Please provide proper values

**Step 6:** Read Data.txt file

**Step 7:** If reply=1, go to step 8 else go to step 9

**Step 8:** Print details of all lands and go to step 2

**Step 9:** If reply=2, go to step 13 else go to step 10

**Step 10:** If reply=3, go to step 46 else go to step 11

**Step 11:** If reply=4, go to step 77 else go to step 12

**Step 12:** Print Invalid number, Please provide valid value. Go to step 2

**Step 13:** Print details of all lands

**Step 14:** Print Enter Kitta no. of the land you would like to rent

**Step 15:** Input kitta

**Step 16:** If kitta is an integer, go to step 18 else go to step 17

**Step 17:** Print Invalid Format, Please provide proper values. Go to step 14

**Step 18:** If kitta exists in land details, go to step 22 else go to step 19

**Step 19:** Print Please enter valid kitta no. or Type exit to return to main menu

**Step 20:** Input reply

**Step 21:** If reply=exit, go to step 2 else go to step 18

**Step 22:** If land is available for rent, go to step 26 else go to step 23

**Step 23:** Print The selected land is not available for rent. Please select another Kitta or Type exit to return to main menu

**Step 24:** Input reply

**Step 25:** If reply=exit, go to step 2 else go to step 18

**Step 26:** Print Enter your full name

**Step 27:** Input buyer’s name

**Step 28:** Print How long would you like to rent this land? (in months)

**Step 29:** Input duration

**Step 30:** If duration is an integer, goto step 32 else go to step 31

**Step 31:** Print Invalid format, Please provide proper values. Go to Step 28

**Step 32:** If duration is greater than 0, go to step 33 else go to step 28

**Step 33:** Print Would you like to rent more lands? Press 1 to rent more lands. Press 2 to complete the renting process

**Step 34:** Input reply

**Step 35:** If rent\_more\_reply is an integer, go to step 37 else go to step 36

**Step 36:** Print Invalid format, Please provide proper values. Go to step 33

**Step 37:** If rent\_more\_reply=1, go to step 40 else go to step 38

**Step 38:** Change the status from Available to Not Available in Data.txt file

**Step 39:** Generation a bill in text file with unique name. Go to step 13

**Step 40:** If rent\_more\_reply=2, go to step 42 else go to step 41

**Step 41:** Print Invalid input, Please provide valid values. Go to step 33

**Step 42:** Change the status from Available to Not Available in Data.txt file

**Step 43:** Generate a bill in text file with unique name

**Step 44:** Print bill in shell

**Step 45:** Print Welcome to Techno Property Nepal Family. Go to step 2

**Step 46:** Print details of all lands

**Step 47:** Print Enter Kitta no. of the land you would like to return

**Step 48:** Input kitta

**Step 49:** If return\_land\_no is integer, go to step 51 else go to step 50

**Step 50:** Print Invalid format, Please provide proper values. Go to Step 47

**Step 51:** If return\_land\_no exists in land details, go to step 55 else go to 52

**Step 52:** Print Enter valid kitta no. or Type exit to return to main menu

**Step 53:** Input reply

**Step 54:** If reply=exit go to step 2 else go to step 51

**Step 55:** If status of return\_land\_no is Not Available, go to step 59 else go to step 56

**Step 56:** Print The selected land has not been rented. Please select another kitta no. or Type exit to return to main menu

**Step 57:** Input reply

**Step 58:** If reply= exit, go to step 2 else go to step 51

**Step 59:** Print Enter your full name

**Step 60:** Input name

**Step 61:** Print Enter the duration of rent as per rent. Enter the actual duration of rent

**Step 62:** Input duration\_ini and duration\_fin

**Step 63:** If duration\_ini and duration\_fin are integer, go to step 65 else go to step 64

**Step 64:** Print Invalid format, Please provide proper values

**Step 65:** Print Would you like to return more lands? Press 1 to rent more lands. Press 2 to complete the returning process

**Step 66:** Input reply

**Step 67:** If return\_more is an integer, go to Step 69 else go to step 68

**Step 68:** Print Invalid format, Please provide proper values

**Step 69:** If return\_more=1, go to step 71 else go to step 70

**Step 70:** Change the status from Not Available to Available in Data.txt. Go to Step 46

**Step 71:** If return\_more=2, go to Step 73 else go to step 72

**Step 72:** Print Invalid Input, Please provide valid values. Go to step 65

**Step 73:** Change the status from Not Available to Available in Data.txt

**Step 74:** Generate a bill in text file with unique name and fine appropriate amount for rent duration exceeding contract duration

**Step 75:** Print Bill in shell

**Step 76:** Print Land returned successfully. Go to Step 2

**Step 77:** Print Thank you for choosing us

**Step 78:** Stop

## 2.2. Flowchart

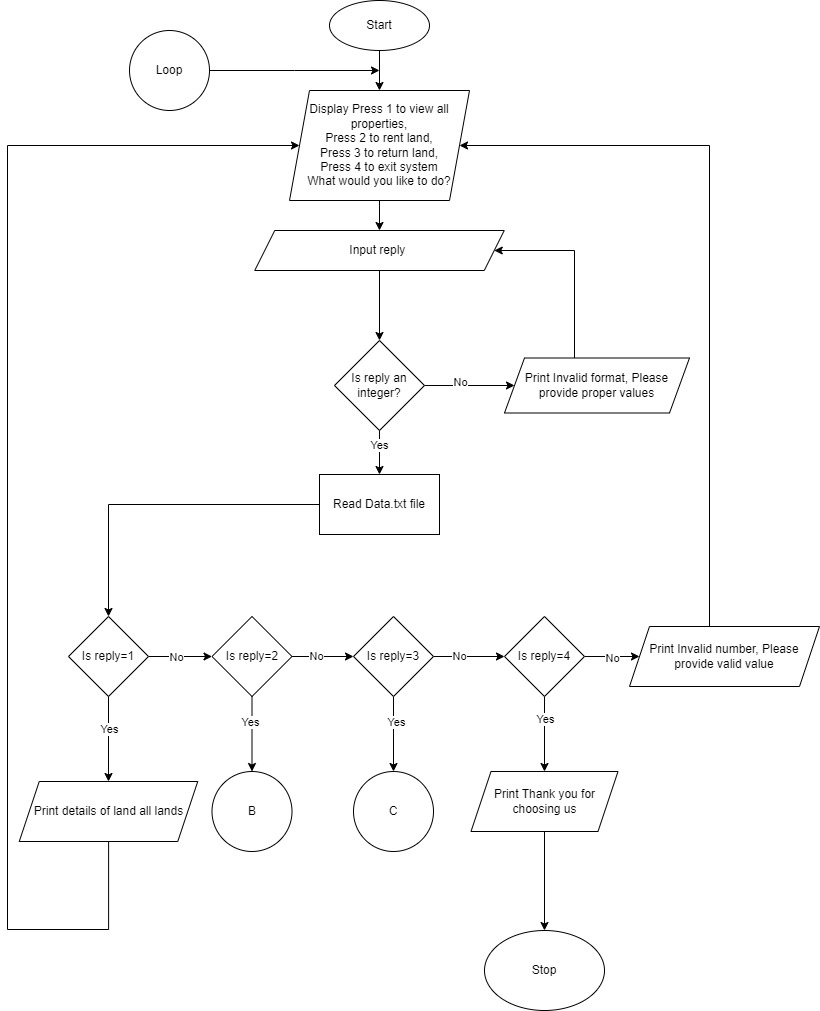


Figure 3: Flowchart of system (1)

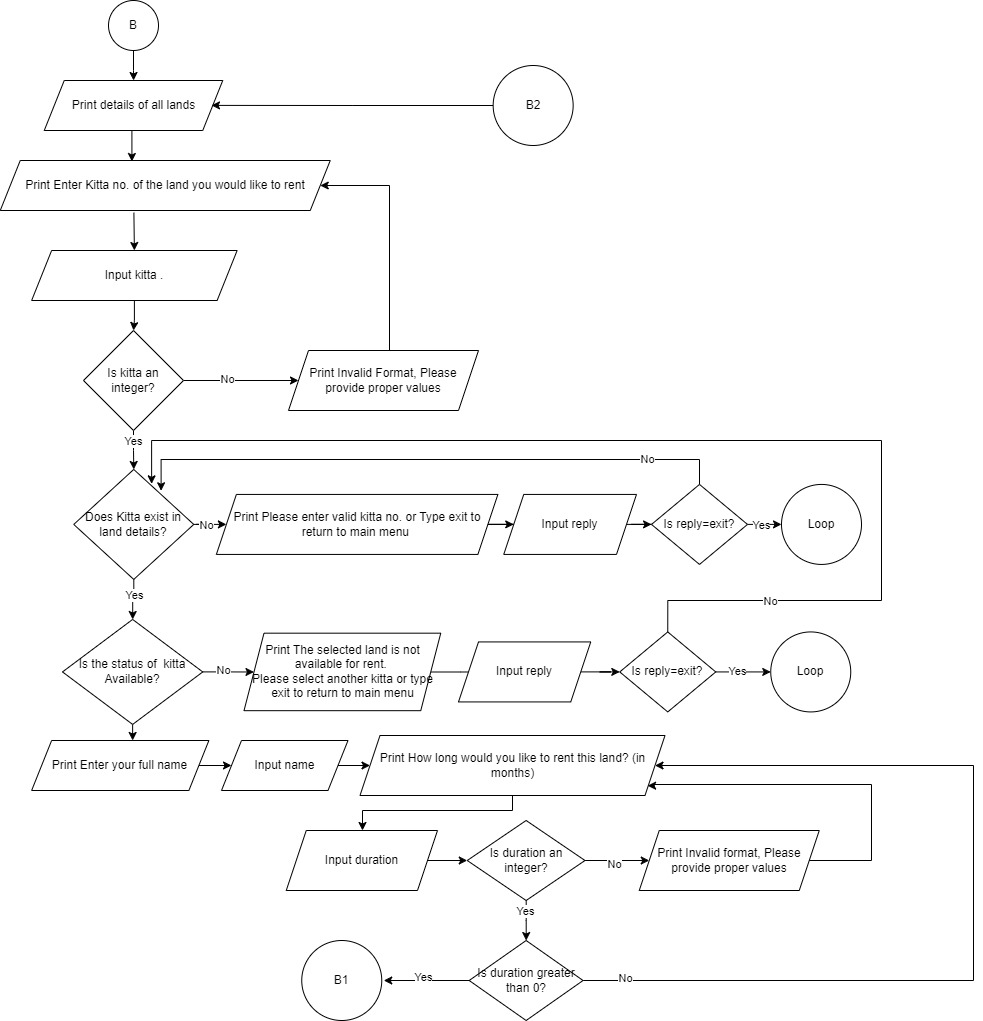


Figure 4 : Flowchart of system (2)

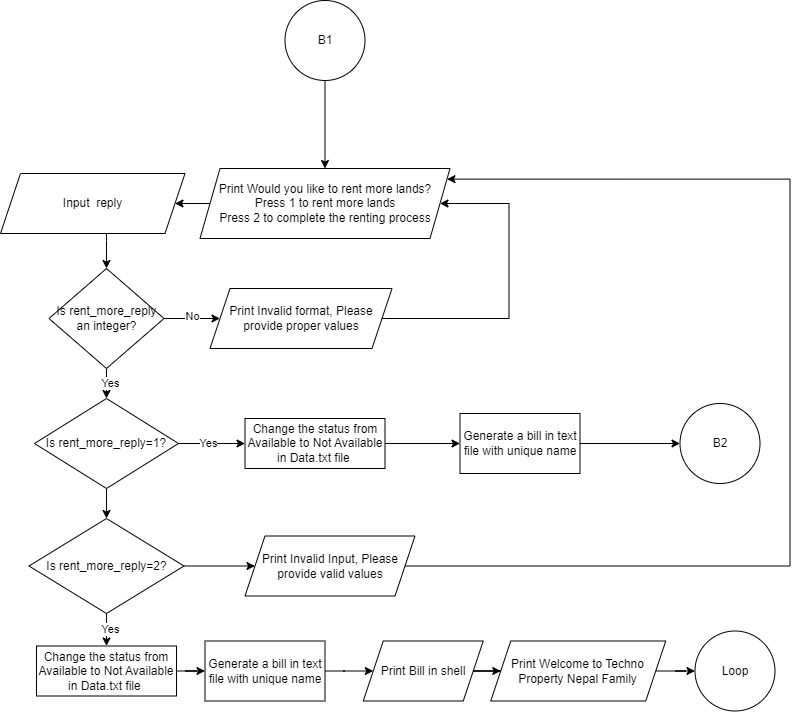


Figure 5: Flowchart of system (3)

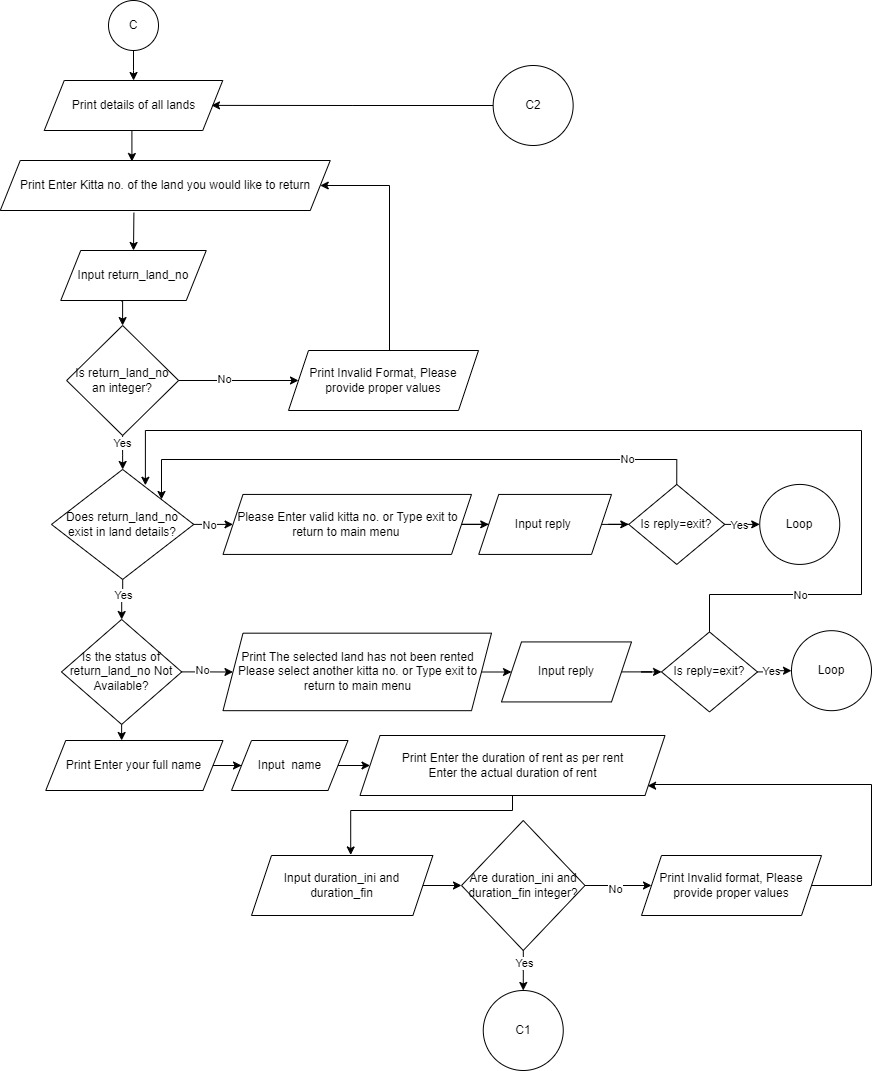


Figure 6: Flowchart of system (4)

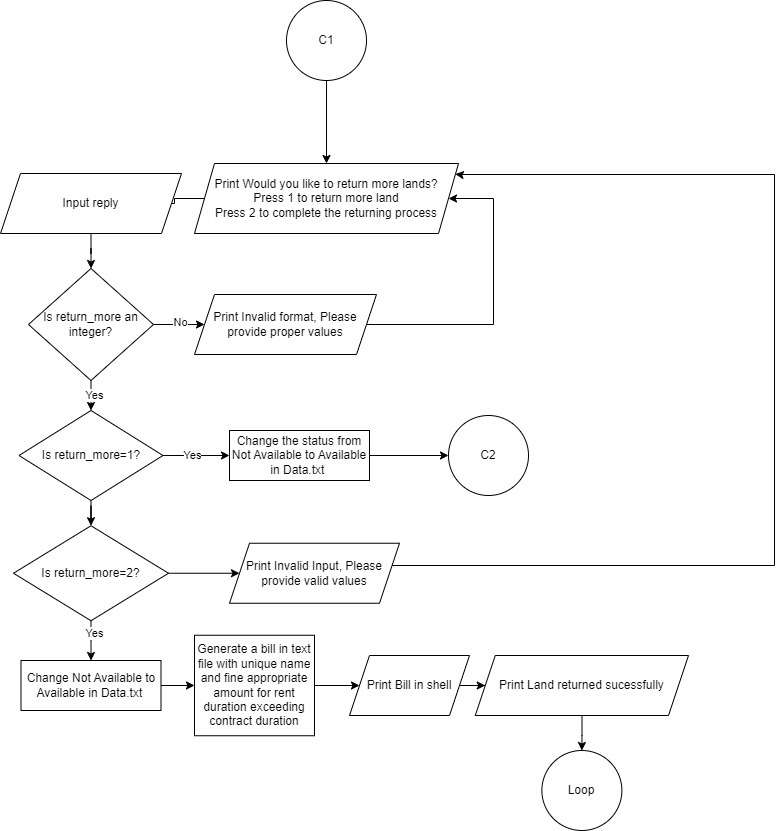


Figure 7: Flowchart of system (5)

## 2.3. Pseudocode

### 2.3.1. Main.py

**IMPORT** all from read

**IMPORT** all from write

**IMPORT** all from operations

**IMPORT** datetime

**PRINT** Techno Property Nepal

**PRINT** Kamalpokhari, Kathmandu

**PRINT** Twenty one dash symbol

**INITIALIZE** unique\_bill to concatenation of convert to string (datetime.datetime.now ().year), convert to string (datetime.datetime.now().month), convert to string (datetime.datetime.now().day), convert to string (datetime.datetime.now().hour), convert to string (datetime.datetime.now().minute), convert to string (datetime.datetime.now().second)

**INITIALIZE** date\_bill to convatenation of convert to string (datetime.datetime.now().year), ".", convert to string (datetime.datetime.now().month),, ".", convert to string (datetime.datetime.now().day)

**INITIALIZE** loop=True

**WHILE** loop equals True

**PRINT** To view all rental properties, press 1

**PRINT** To rent a land, press 2

**PRINT** To return the land, press 3

**PRINT** To exit, press 4

**WHILE** True

**TRY**

**PRINT** What would you like to do? ->

**ASSIGN** user reply to reply and convert to int

**BREAK** the loop

**EXCEPT**

**PRINT** Invalid Format, Please provide proper values

**END WHILE**

**INITIALIZE** newlist to return value of read function

**IF** reply equals to one

**CALL** viewAllLands function with newlist as arguments

**END IF**

**ELIF** reply equals to two

**INITIALIZE** rent\_count to zero

**INITIALIZE** rented\_list to empty list

**INITIALIZE** rent\_more\_outer to zero

**INITIALIZE** duration\_list to empty list

**WHILE** rent\_more\_outer equals to 0

**CALL** viewAllLands function with newlist as argument

**INITIALIZE** valid\_kitta to True

**WHILE** valid\_kitta is True

**TRY**

**PRINT** Enter kitta no. of the land you would like to rent ->

**INITIALIZE** kitta to entered kitta and convert to int

**ASSIGN** valid\_kitta to False

**EXCEPT**

**PRINT** Invalid Format, Please provide proper values

**END WHILE**

**INITIALIZE** in\_loop to zero

**WHILE** in\_loop equals to zero

**INITIALZIE** i to return value of check\_kitta function with arguments newlist

and kitta

**IF** i equals to negative one

**ASSIGN** rent\_more\_outer to one

**BREAK**

**END IF**

**INITIALIZE** available to return value of check\_availability function and pass

newlist and I as argument

**IF** available equals to negative one

**ASSIGN** rent\_more\_outer to one

**BREAK**

**END IF**

**ELIF** available equals to zero

**IF** rent\_count equals to zero

**PRINT** Enter your full name ->

**INITIALIZE** name to entered name

**ENDIF**

**ASSIGN** valid\_duration to True

**WHILE** valid\_duration is True

**TRY**

**PRINT** How long would you like to rent this land? in months ->

**ASSIGN** duration to entered month and convert to int

**EXCEPT**

**PRINT** Invalid Format, Please provide proper values

**IF** duration is less than one

**PRINT** Please enter valid duration for renting

**END IF**

**ELSE**

**APPEND** durationtoduration\_list

**BREAK**

**END IF**

**END WHILE**

**INITIALIZE** rent\_more to zero

**WHILE** rent\_more equals to zero

**WHILE** True

**TRY**

**PRINT** Would you like to rent more lands? Press 1 to rent more lands Press 2 to complete the renting process ->

**ASSIGN** rent\_more\_reply to user reply and convert to int

**BREAK**

**EXCEPT**

**PRINT** Invalid Format, Please provide proper values

**END WHILE**

**IF** rent\_more\_reply equals to one

**ASSIGN** in\_loop to one

**APPEND** ito rented\_list

**CALL** bill function with arguments i, newlist, name, duration,

rent\_count, unique\_bill and date\_bill

**INCREMENT** rent\_count by one

**CALL** change\_data function with arguments newlist and i

**BREAK**

**END IF**

**ELIF** rent\_more\_reply equals to two

**APPEND** i to rented\_list

**CALL** bill function with arguments i, newlist, name, duration,

rent\_count, unique\_bill and date\_bill

**CALL** change\_data function with arguments newlist and i

**CALL** bill\_total function with arguments newlist, name, rented\_list,

and unique\_bill

**CALL** print\_bill function with arguments newlist, name, duration\_list,

rented\_list, unique\_bill and date\_bill

**PRINT** Welcome to the Techno Property Nepal family

**ASSIGN** in\_loop to one

**ASSIGN** rent\_more\_outer to one

**BREAK**

**END IF**

**ELSE**

**PRINT** Invalid input detected, Please provide valid number

**END IF**

**END** **IF**

**ELSE**

**ASSIGN** kitta to available

**END** **WHILE**

**END** **IF**

**ELIF** reply equals to three

**INITIALIZE** return\_list to empty list

**INITIALIZE** duration\_ini\_list to empty list

**INITIALIZE** duration\_fin\_list to empty list

**INITIALIZE** fine\_list to empty list

**INITIALIZE** return\_loop to True

**INITIALIZE** return\_count to zero

**WHILE** return\_loop is True

**CALL** viewAllLands function with argumet newlist

**WHILE** True

**TRY**

**PRINT** Enter kitta number of the land you would like to return ->

**INITIALIZE** return\_land\_no to entered kitta and convert to int

**BREAK**

**EXCEPT**

**PRINT** Invalid format, Please provide proper values

**END WHILE**

**INITIALIZE** not\_available to zero

**WHILE** not\_available equals to zero

**INITIALIZE** I to return value of check\_kitta function passing newlist and

return\_land\_no as arguments

**IF** i equals to negative one

**ASSIGN** return\_loop to one

**BREAK**

**END** **IF**

**INITIALIZE** unavailable to return value of check\_unavailability function

passing newlist and I as arguments

**IF** unavailable equals to negative one

**ASSIGN** return\_loop to one

**BREAK**

**END IF**

**ELIF** unavailable equals to zero

**IF** return\_count equals to zero

**PRINT**  Enter your full name ->

**INITIALIZE** name with customer’s name

**END IF**

**WHILE** True

**TRY**

**PRINT** Enter the duration of rent as per the contract ->

**INITIALIZE** duration\_ini with contract duration and convert to int

**PRINT** Enter the actual duration of rent ->

**INITIALIZE** duration\_final with actual duration and convert to int

**BREAK**

**EXCEPT**

**PRINT** Invalid Format, Please provide proper values

**END WHILE**

**INITIALIZE** fine to return value of check\_fine function with arguments

duration\_ini and duration\_final

**APPEND** fine to fine\_list

**WHILE** True

**TRY**

**PRINT** Would you like to return more lands? Press 1 to return more

land Press 2 to complete the returning process ->

**INITIALIZE** return\_more to user reply and convert to int

**BREAK**

**EXCEPT**

**PRINT** Invalid Format, Please provide proper values

**END WHILE**

**WHILE** True

**APPEND** itoreturn\_list

**APPEND** duration\_initoduration\_ini\_list

**APPEND** duration\_final to duration\_fin\_list

**IF** return\_more equals to one

**CALL** change\_data\_return function with arguments newlist and i

**INCREMENT** return\_count by one

**ASSIGN** not\_available to one

**BREAK**

**END IF**

**ELIF** return\_more equals to two

**CALL** return\_note function with arguments newlist, return\_list, name,

duration\_ini\_list, duration\_fin\_list, unique\_bill, date\_bill and fine\_list

**CALL** change\_data\_return function with arguments newlist and i

**CALL** print\_ret\_bill function with arguments newlist, return\_list, name,

duration\_ini\_list, duration\_fin\_list, unique\_bill, date\_bill and fine\_list

**PRINT** Land returned sucessfully

**ASSIGN** return\_loop to False

**ASSIGN** not\_available to one

**BREAK**

**END IF**

**ELSE**

**PRINT** Invalid number, Please provide valid number

**END IF**

**END WHILE**

**ELSE**

**UPDATE** return\_land\_no to unavailable

**END** **IF**

**END** **WHILE**

**END WHILE**

**ELIF** reply equals to four

**PRINT** Thank you for choosing us

**UPDATE** loop to False

**END** **IF**

**ELSE**

**PRINT** Invalid number, Please provide valid value.

**END IF**

**END WHILE**

### 2.3.2. Read.py

**DEFINE** function read()

**INITIALIZE** newlist as an empty list

**READ** Data.txt file

**STORE** the data in newlist

**RETURN** newlist

### 2.3.3. Operations.py

**IMPORT** all from read

**DEFINE** funtion viewAllLands with parameter newlist

**PRINT** empty line

**FOR** line in newlist

**FOR** item in line

**PRINT** item

**END FOR**

**PRINT** emptyline

**END** **FOR**

**DEF** function check\_kitta with parameters newlist and kitta

**FOR** i in range length of newlist

**IF** newlist[i][0] equals to kitta converted to string

**RETURN** i

**END** **IF**

**PRINT** Please enter valid kitta no. or Type exit to return to main menu:\n->

**INITIALIZE** reply to user reply

**IF** reply converted to lower case equals to exit

**RETURN** negative one

**END** **IF**

**ELSE**

**RETURN** function check\_kitta with arguments newlist and reply

**END** **IF**

**DEF** function check\_availability with parameters newlist and i

**INITIALIZE** check\_avai\_list to parameter newlist

**INITIALIZE** check\_avai\_list[i][-1]to check\_avai\_list[i][-1] where all one space is

removed

**IF** check\_avai\_list[i][-1] converted to lower case is equal to available

**RETURN** zero

**END** **IF**

**ELSE**

**PRINT** The selected land is not available to rent

**PRINT** Please select another kitta no. OR Type exit to return to main menu:\n->

**INITIALIZE** reply to enter user reply

**IF** reply converted to lower case equals to exit

**RETURN** negative one

**END** **IF**

**ELSE**

**RETURN** reply

**END** **IF**

**DEF** function print\_bill with parameters newlist, name, duration, rented, unique and date

**INITIALIZE** dur to zero

**PRINT** Techno Property Nepal

**PRINT** Kamalpokhari, Kathmandu

**PRINT** “-“ twenty three times

**PRINT** Date: date parameter

**PRINT** Bill no.: unique parameter

**PRINT** "=" hundred thirty one times

**PRINT** Name: name parameter converted to upper case

**FOR** i in rented

**PRINT** Kitta Number: newlist[i][0]

**PRINT** City/District: newlist[i][1]

**PRINT** Direction of Land: newlist[i][2]

**PRINT** Area of Land (Anna): "+newlist[i][3]

**PRINT** Duration of rent: duration[dur]

**PRINT** Amount: newlist[i][4]

**PRINT** "=" hundred thirty one times

**INCREMENT** dur by one

**END FOR**

**INITIALIZE** total to zero

**PRINT** Total Amount: and end with empty space

**INITIALIZE** count to zero

**FOR** i in rented

**IF** count equal to zero

**PRINT** newlist[i][4] end with empty space

**END** **IF**

**ELSE**

**PRINT** newlist[i][4] end with empty space

**END IF**

**INCREMENT** count by one

**ASSIGN** total to total plus newlist[i][4] converted to int

**END FOR**

**IF** count is greater than one

**PRINT** total

**DEF** function check\_unavailability with parameters newlist and i

**INITIALIZE** check\_unavai\_list to parameter newlist

**UPDATE** check\_unavai\_list[i][-1] to check\_unavai\_list[i][-1] where all one space is

removed

**IF** check\_unavai\_list[i][-1] converted to lower case is equal to notavailable

**RETURN** zero

**END** **IF**

**ELSE**

**PRINT** The selected land has not been rented

**PRINT** Please select another kitta no. OR Type exit to return to main menu:\n->

**INITIALIZE** reply to user reply

**IF** reply converted to lower case equals to exit

**RETURN** negative one

**END IF**

**ELSE**

**RETURN** reply

**END IF**

**DEF** function print\_ret\_bill with parameters newlist, return\_list, name, ini, fin, unique, date and fine\_list

**INITIALIZE** dur to zero

**INITIALIZE** total\_amt to zero

**PRINT** Techno Property Nepal

**PRINT** Kamalpokhari, Kathmandu

**PRINT** "-" twenty three times

**PRINT** Date: date parameter

**PRINT** Bill no.: unique parameter

**PRINT** "=" hundred thirty one times

**PRINT** Name: name parameter converted to upper case

**FOR** i in return\_list

**PRINT** Kitta Numer: newlist[i][0]

**PRINT** City/District: newlist[i][1]

**PRINT** Direction of Land: newlist[i][2]

**PRINT** Area of Land (Anna): newlist[i][3]

**PRINT** Duration of rent according to contract: ini[dur] converted to string

**PRINT** Actual duration of rent : fin[dur] converted to string

**IF** fine\_list[dur] is not equal to zero

**ASSIGN** amount to newlist[i][4] converted to integet multiplied to fin[dur]

**PRINT** Amount: newlist[i][4] converted to string x fin[dur] converted to string is

equal to amount converted to string

**END IF**

**ELSE**

**ASSIGN** amount to newlist[i][4] converted to int multiplied to ini[dur]

**PRINT** Amount: newlist[i][4] converted to string x ini[dur] converted to string

equal to Amount converted to string

**END** **IF**

**UPDATE** total\_amt to amount plus itself

**PRINT** empty line

**PRINT** "=" one thirty one times

**INCREMENT** dur by one

**END** **FOR**

**IF** fine\_list[0] is not equal to zero

**PRINT** Post contract termination fine:

**INITIALIZE** TOTAL\_fine to zero

**INITIALIZE** total to zer

**INITIALIZE** dur to zero

**FOR** i in return\_list

**PRINT** For Kitta no.: newlist[i][0]

**ASSIGN** fine to newlist[i][4] converted to int times point two zero

**FOR** j in range one to fine\_list[dur] plus one

**PRINT** Month j converted to string: fine converted to string

**UPDATE** total to fine plus itself

**UPDATE** fine to itself times point one

**PRINT** Fine: total converted to string

**UPDATE** total\_fine to itself plus total

**INCREMENT** dur by one

**END FOR**

**PRINT** "=" one thirty one times

**PRINT** Total Amount: total\_amt converted to string

**PRINT** Total Fine: total\_fine converted to string

**PRINT** Grand Total: total\_amt converted to string total\_fine converted to string equals

to (total\_amt plus total\_fine) converted to string

**PRINT** "=" one thirty one times

**PRINT** Note: The initial fine is calculated as 20% of the monthly rent. It is compounded monthly at a rate of 10%.

**DEF** function check\_fine with parameter ini and fin

**IF** ini is less than fin

**RETURN** ini subtracted from fin

**END** **IF**

**ELSE**

**RETURN** zero

**END** **IF**

### 2.3.4. Write.py

**IMPORT** all from operations

**DEF** function change\_data with parameters newlist and i

**OPEN** Data.txt in write mode as file

**UPDATE** newlist[i][-1] to Not Available

**FOR** line in range length of newlist

**FOR** item in range length of newlist[line]

**IF** item equal to zero

**WRITE** newlist[line][item]

**END IF**

**ELSE**

**WRITE** comma and newlist[line][item]

**END** **IF**

**END FOR**

**WRITE** in new line

**END FOR**

**DEF** function bill with parameters i, newlist, name, duration, count, unique, date

**IF** count equals to zero

**OPEN** Rent\_name\_unique.txt in write mode as file

**WRITE** Techno Property Nepal

**WRITE** Kamalpokhari, Kathmandu

**WRITE** "-" twenty three times

**WRITE** Date: date

**WRITE** Bill no.: unique

**WRITE** "=" one thirty one times

**WRITE** Name: name converted to uppercase

**WRITE** Kitta Number: newlist[i][0]

**WRITE** City/District: newlist[i][1]

**WRITE** Direction of Land: newlist[i][2]

**WRITE** Area of Land (Anna): newlist[i][3]

**WRITE** Duration of rent: duration converted to string

**WRITE** Amount: newlist[i][4]

**END** **IF**

**ELSE**

**OPEN** Rent\_name\_unique.txt in append mode as file

**WRITE** "=" one thirty one times

**WRITE** Kitta Number: newlist[i][0]

**WRITE** City/District: newlist[i][1]

**WRITE** Direction of Land: newlist[i][2]

**WRITE** Area of Land (Anna): newlist[i][3]

**WRITE** Duration of rent: duration convert to string

**WRITE** Amount: newlist[i][4]

**END** **IF**

**DEF** function bill\_total with parameters newlist, name, rented and unique

**OPEN** Rent\_name\_unique.txt in append mode as file

**INITIALIZE** total to zero

**WRITE** "=" one thirty one times

**WRITE** Total Amount:

**INITIALIZE** count to zero

**FOR** i in rented

**IF** count equals to zero

**WRITE** newlist[i][4]

**END** **IF**

**ELSE**

**WRITE** plus newlist[i][4]

**END** **IF**

**INCREMENT** count by one

**UPDATE** total to itself plus newlist[i][4] convert to int

**END FOR**

**IF** count is greater than one

**WRITE** total convert to string

**END** **IF**

**DEF** function change\_data\_return with parameters newlist and i

**OPEN** Data.txt in write mode as file

**UPDATE** newlist[i][-1] to Available

**FOR** line in range length of newlist

**FOR** item in range length of newlist[line]

**IF** item equals to zero

**WRITE** newlist[line][item]

**END** **IF**

**ELSE**

**WRITE** comma plus newlist[line][item]

**END** **FOR**

**WRITE** new line

**END** **FOR**

**DEF** function return\_note with parameters newlist, return\_list, name, ini, fin, unique, date, fine\_list

**OPEN** Return\_name\_unique.txt in write mode as file

**INITIALIZE** dur to zero

**INITIALIZE** total\_amt to zero

**WRITE** Techno Property Nepal

**WRITE** Kamalpokhari, Kathmandu

**WRITE** "-" twenty three times

**WRITE** Date: date

**WRITE** Bill no.: unique

**WRITE** “=” one thirty one times

**WRITE** Name: name converted to upper case

**FOR** i in return\_list

**WRITE** Kitta Numer: newlist[i][0]

**WRITE** City/District: newlist[i][1]

**WRITE** Direction of Land: newlist[i][2]

**WRITE** Area of Land (Anna): newlist[i][3]

**WRITE** Duration of rent according to contract: ini[dur] convert to string

**WRITE** Actual duration of rent : fin[dur] convert to string

**If** fine\_list[dur] is not equal to zero

**UPDATE** amount to newlist[i][4] converted to int times fin[dur]

**WRITE** Amount: newlist[i][4] convert to string x fin[dur] convert to string equal

to amount convert to string

**END IF**

**ELSE**

**UPDATE** amount to newlist[i][4] convert to int times ini[dur]

**WRITE** Amount: newlist[i][4] convert to string x ini[dur] convert to string equal

to amount convert to string

**END IF**

**UPDATE** total\_amt to itself plus amount

**WRITE** two empty line

**WRITE** “=" one thirty one times

**INCREMENT** dur by one

**END FOR**

**IF** fine\_list[0] is not equal to zero

**WRITE** Post contract termination fine

**END IF**

**INITIALIZE** total\_fine to zero

**INTIALIZE** total to zero

**INITIALIZE** dur to zero

**FOR** i in return\_list

**WRITE** For Kitta no.: newlist[i][0]

**UPDATE** fine= newlist[i][4] convert to int times point two

**FOR** j in range one to fine\_list[dur] plus one

**WRITE** Month j convert to string : fine convert to string

**UPDATE** total to itself plus fine

**UPDATE** Fine to itself times point one

**END** **FOR**

**WRITE** Fine: total convert to string

**UPDATE** total\_fine to itself plus total

**INCREMENT** dur by one

**END FOR**

**WRITE** "=" times one thirty one

**WRITE** Total Amount: "+str(total\_amt)

**WRITE** Total Fine: "+str(total\_fine)

**WRITE** Grand Total: total\_amt convert to string plus total\_fine convert to string

equal to (total\_amt+total\_fine) convert to string

**WRITE** "=" one thirty one times

**WRITE** Note: The initial fine is calculated as 20% of the monthly rent. It is

compounded monthly at a rate of 10%.

## 2.4. Data Structure

A data structure is a way to store data (W3schools, 2024). There are two types of data structure:

### 2.4.1. Primitive data structure:

It is a fundamental data structure that allows the value of only one type of data (Gupta, 2024). They offer an easy way to store data in its unprocessed state and are included in the majority of computer languages (Prepbytes, 2023).

* Int

Int data type is a primitive data structure used to represent integer value (Prepbytes, 2023). For ex: 0, 1, 30, etc.



Figure 8: Screenshot of int data type

* Boolean

Boolean data type is a primitive data structure used to represent logical value (Prepbytes, 2023). It is either True or False.



Figure 9: Screenshot of boolean data type

* String

String is collection of alphabets, characters or words (Mitchell, 2022). It is enclosed inside “” or ‘’. For ex: “ABCD”, “Hello”, ”Blue”, etc.



Figure 10: Screenshot of string data type

* Float

The numbers written with a decimal point are called float numbers (Mitchell, 2022).

For ex: 50.12, 20.0, 2.5, etc.

### 2.4.2. Collection data structure:

It is type of data structure that allows the value of multiple type of data. They are created with the help of primitive data structure (Gupta, 2024).

* List

It is a ordered collection of items (Mitchell, 2022). It can be of 1D type or 2D type.

For ex: main\_list=[1,”asd”,50], sub\_list=[“Man”,12,”Status”], etc.



****

Figure 11: Screenshot of list

* Tuple

They are basically list but immutable (Mitchell, 2022).

For ex: lists\_new=(1,2,”abcd”), lists=(1,”sleep”,90), etc.

* Set

They are unorganized collection of unique items (Mitchell, 2022).

For ex: set\_a={“a”,1,”qwe”}, set\_1={1,4,6,”anu”}, etc.

* Dictionary

It is a type of colletion data structure that store items in key-value pair (Simplilearn, 2024). Each key or value can be used to access eachother.

For ex: new\_dict={1:”Python”, 2:”Java”,3:”HTML”}, my\_dictionary={1:”Apple”, 2:”Ball”}, etc.

# Program



Figure 12: Screenshot of start of program

Firstly, the user is provided with 4 choices and an input line for replying.

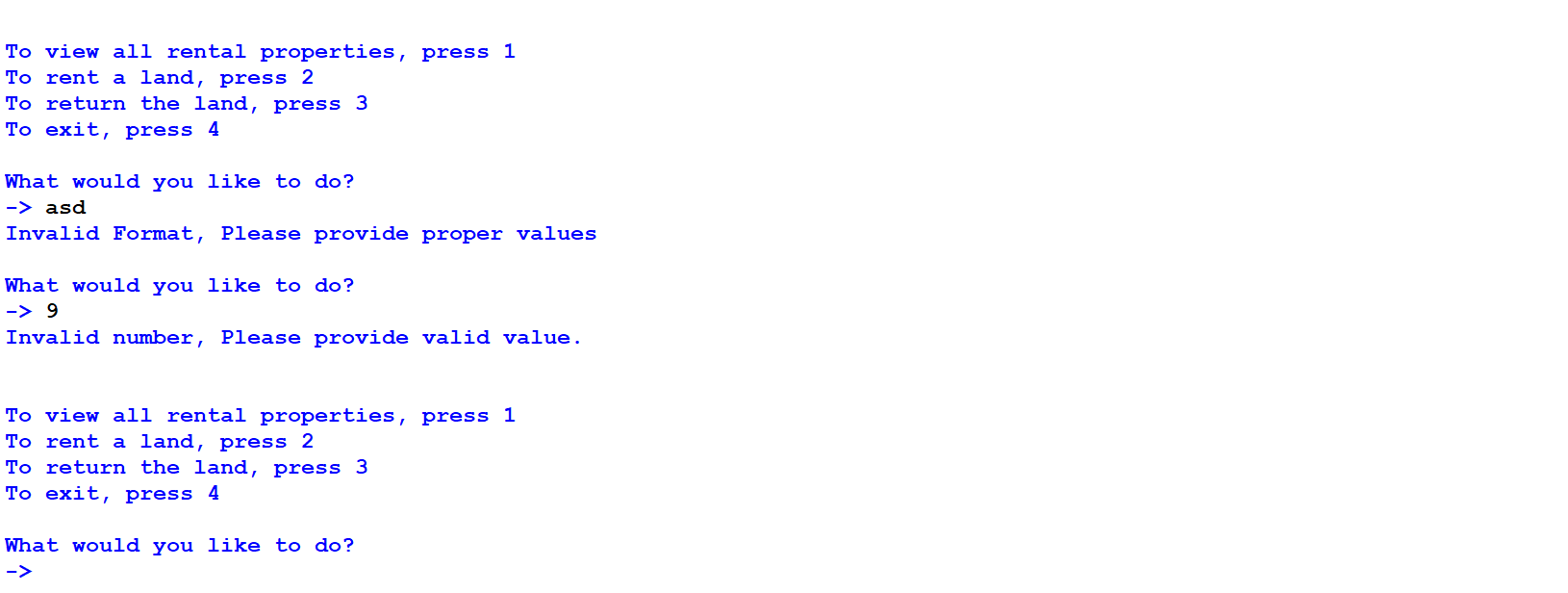


Figure 13: Screenshot of error message after invalid input in main menu

Any invalid input will print an error message and will ask the user to provide valid answer.

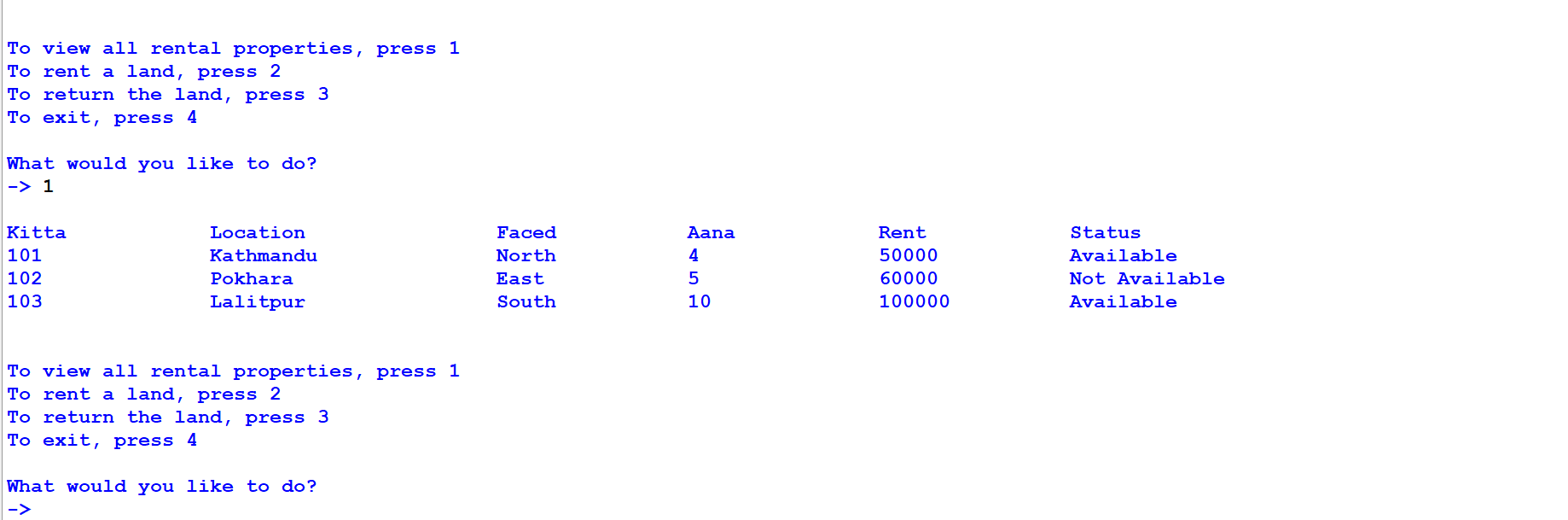


Figure 14: Screenshot of selecting view all rental properties in main menu

If the user enters 1, details of all lands are shown and again the 4 choices are shown.

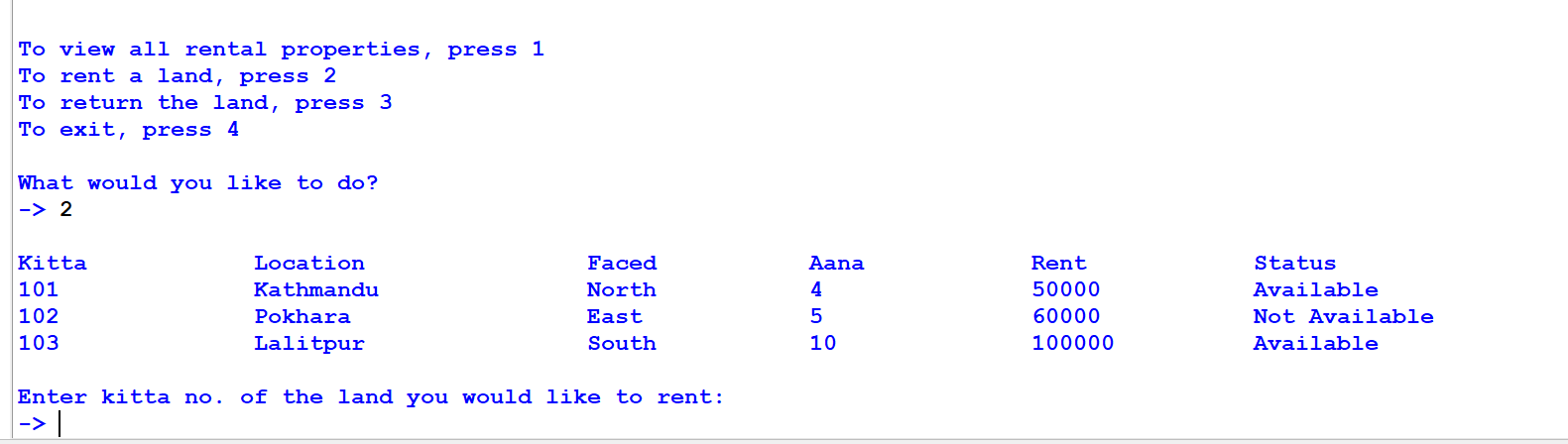


Figure 15: Screenshot of selecting rent a land in main menu

If the user enters 2, all the land’s details are shown and kitta no. of the land to be rented is asked.

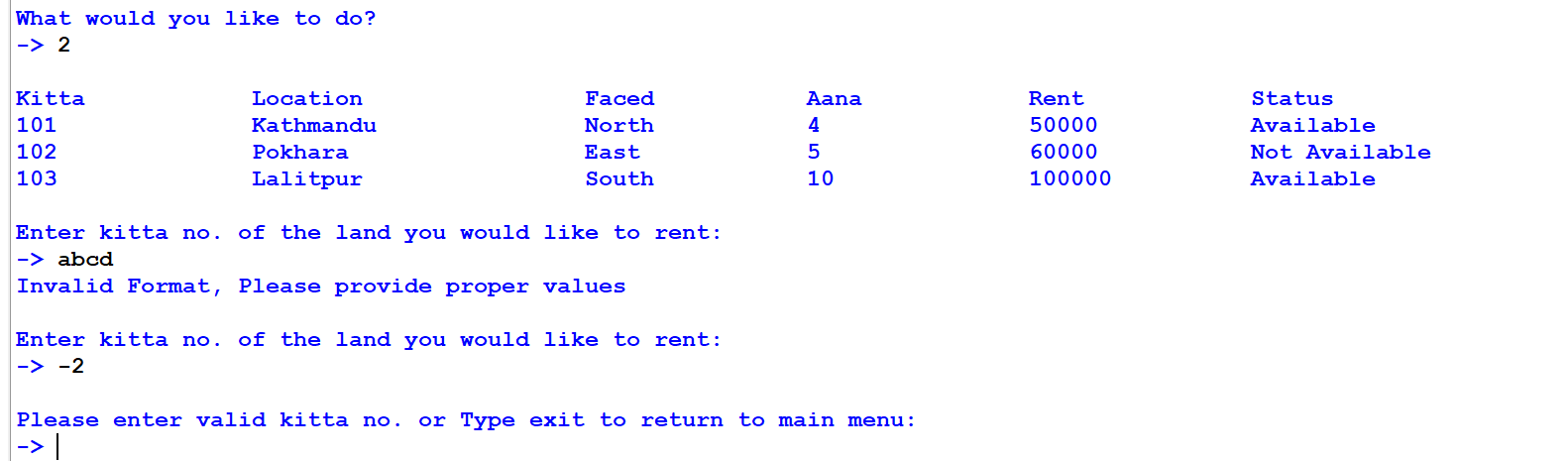


Figure 16: Screenshot of error message after entering invalid kitta while renting

Invalid input will print error and ask the user to enter kitta no. again or type exit to return to main menu.

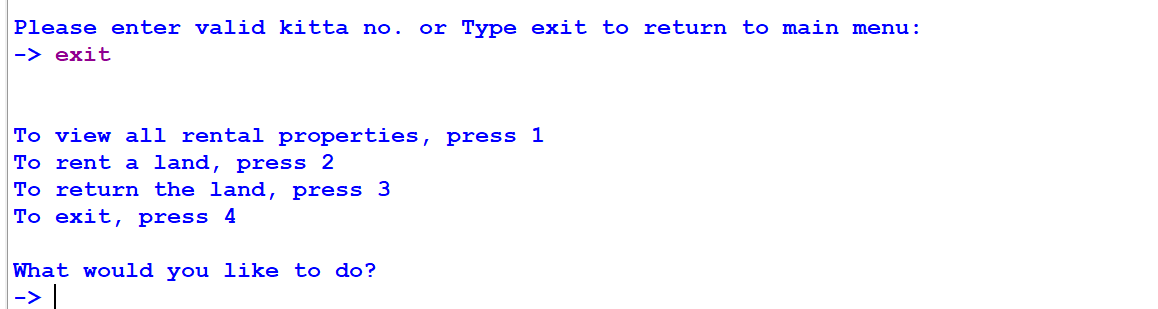


Figure 17: Screenshot of exiting from renting land

If the user types exit, the user is taken back to main menu.

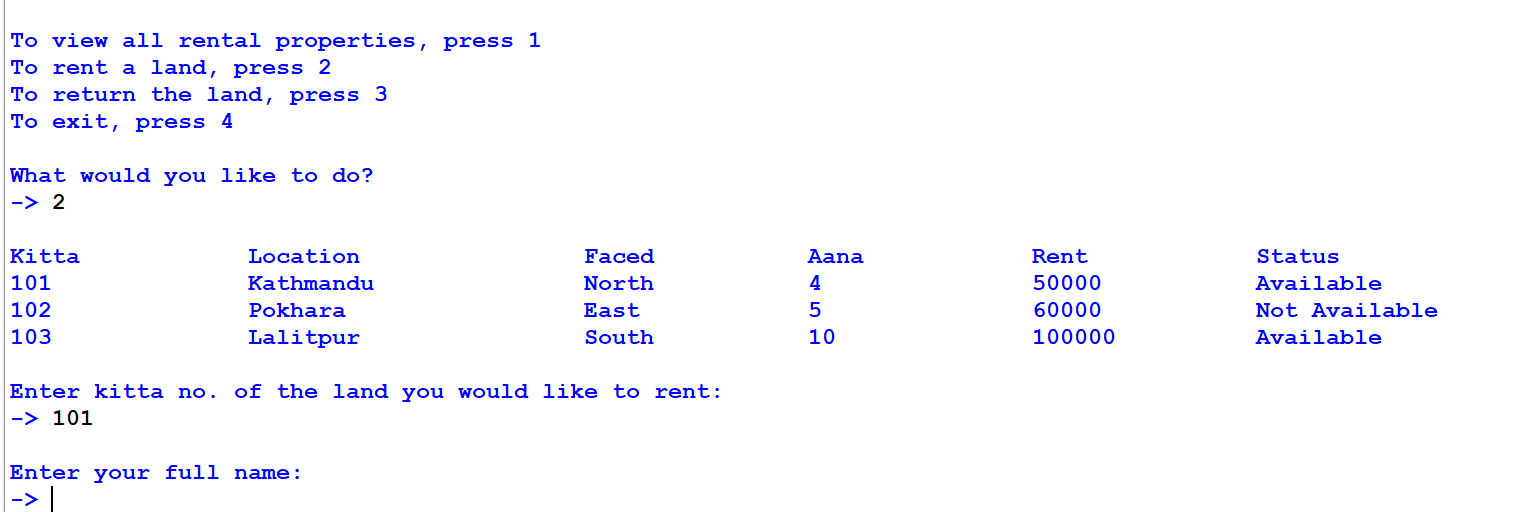
****

Figure 18: Screenshot of entering valid kitta while renting land

When user enters valid kitta no., the user is asked to enter his/her full name.

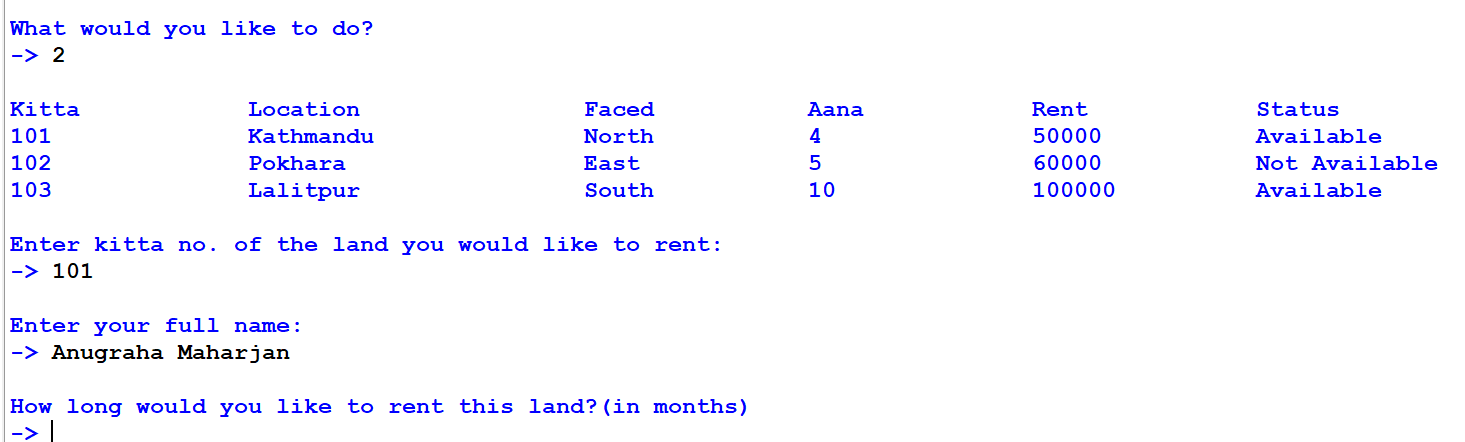


Figure 19: Screenshot of asking user duration of rent

Then, the user is asked the duration of rent

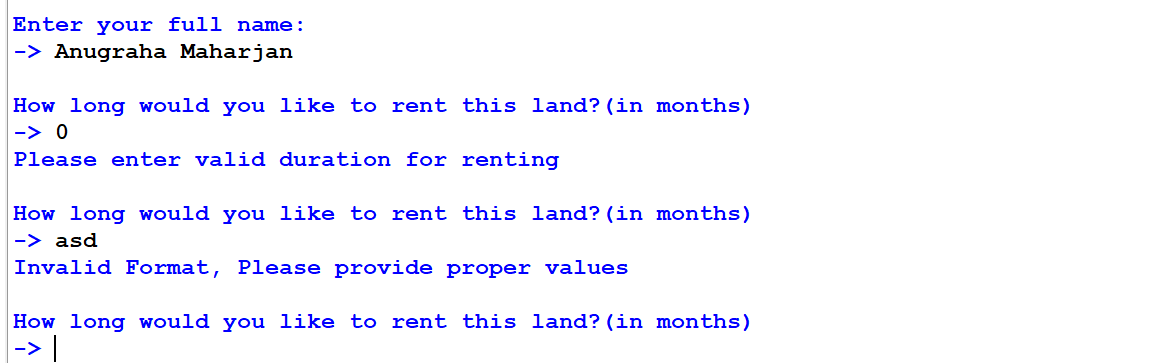


Figure 20: Screenshot of error message after entering invalid duration of rent

Invalid input will generate error message and again ask the duration.

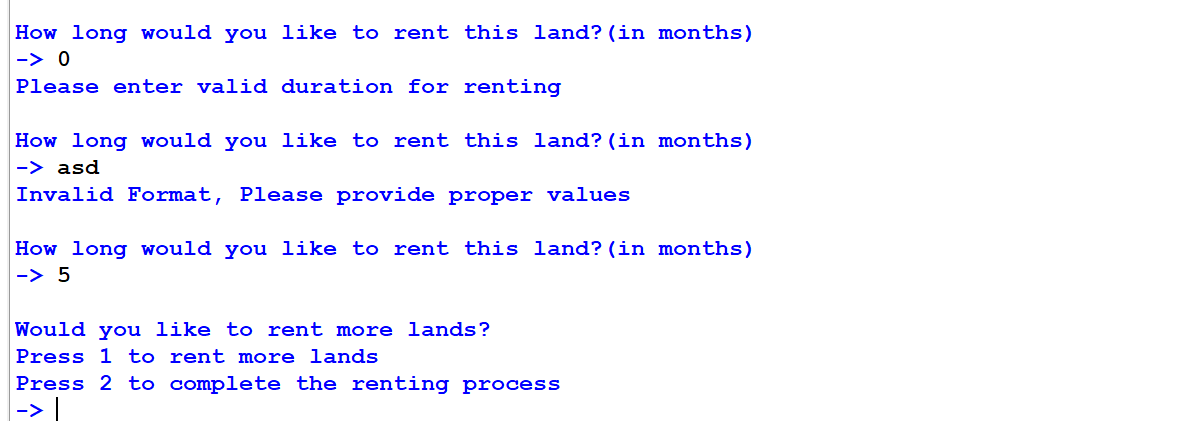


Figure 21: Screenshot after entering valid duration of rent

When a valid duration is entered, the user is asked whether to rent more lands or complete the renting process.

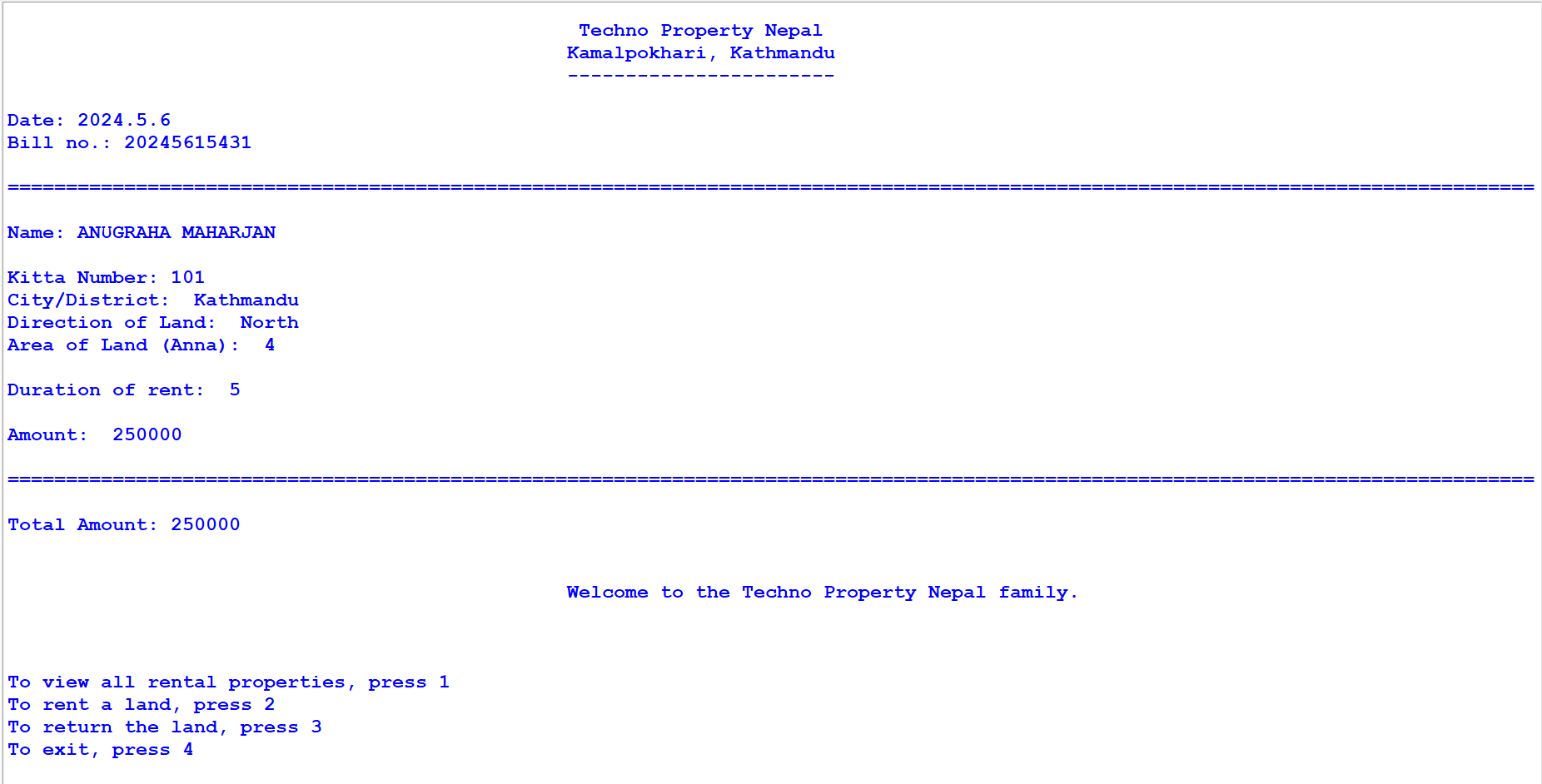


Figure 22: Screenshot of bill printed after renting one land

If the user enters 2, a bill is generated and the 4 choices are displayed again.

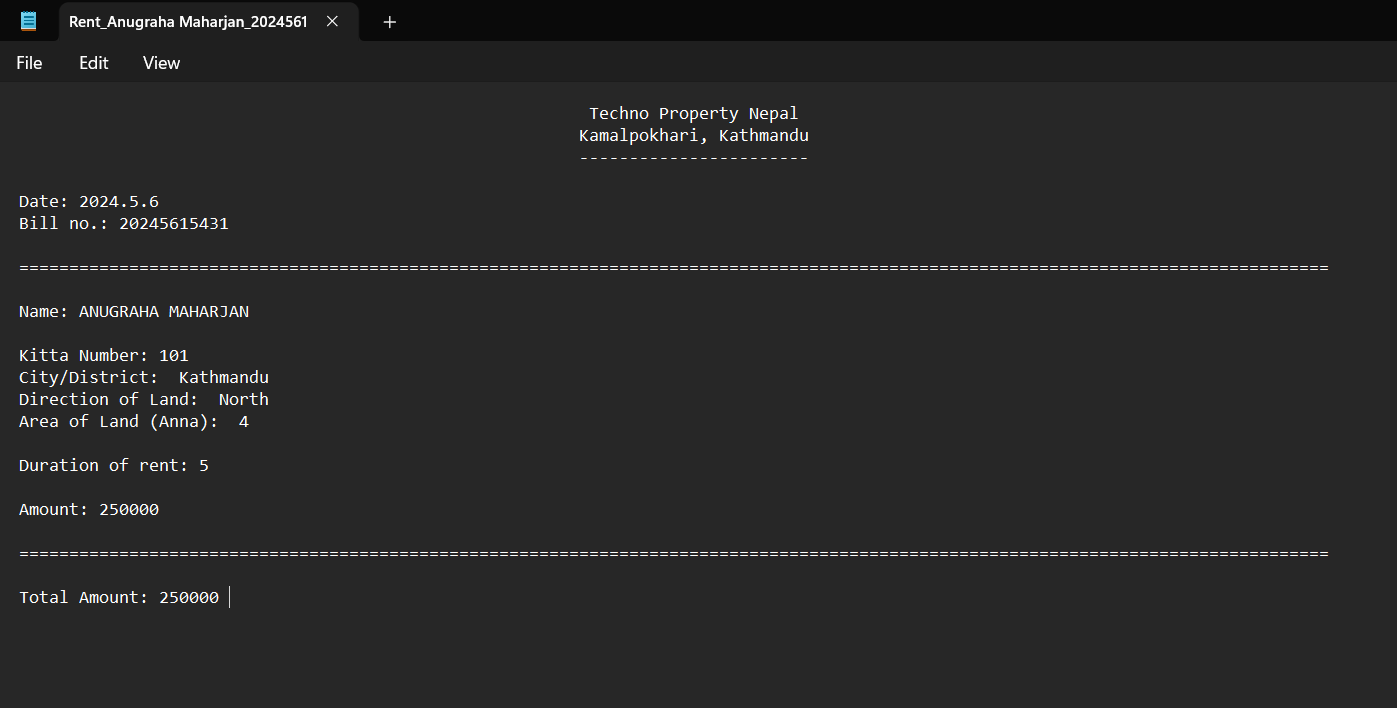


Figure 23: Screenshot of bill generated in .txt file after renting one land

A text file of bill is also generated with a unique name.

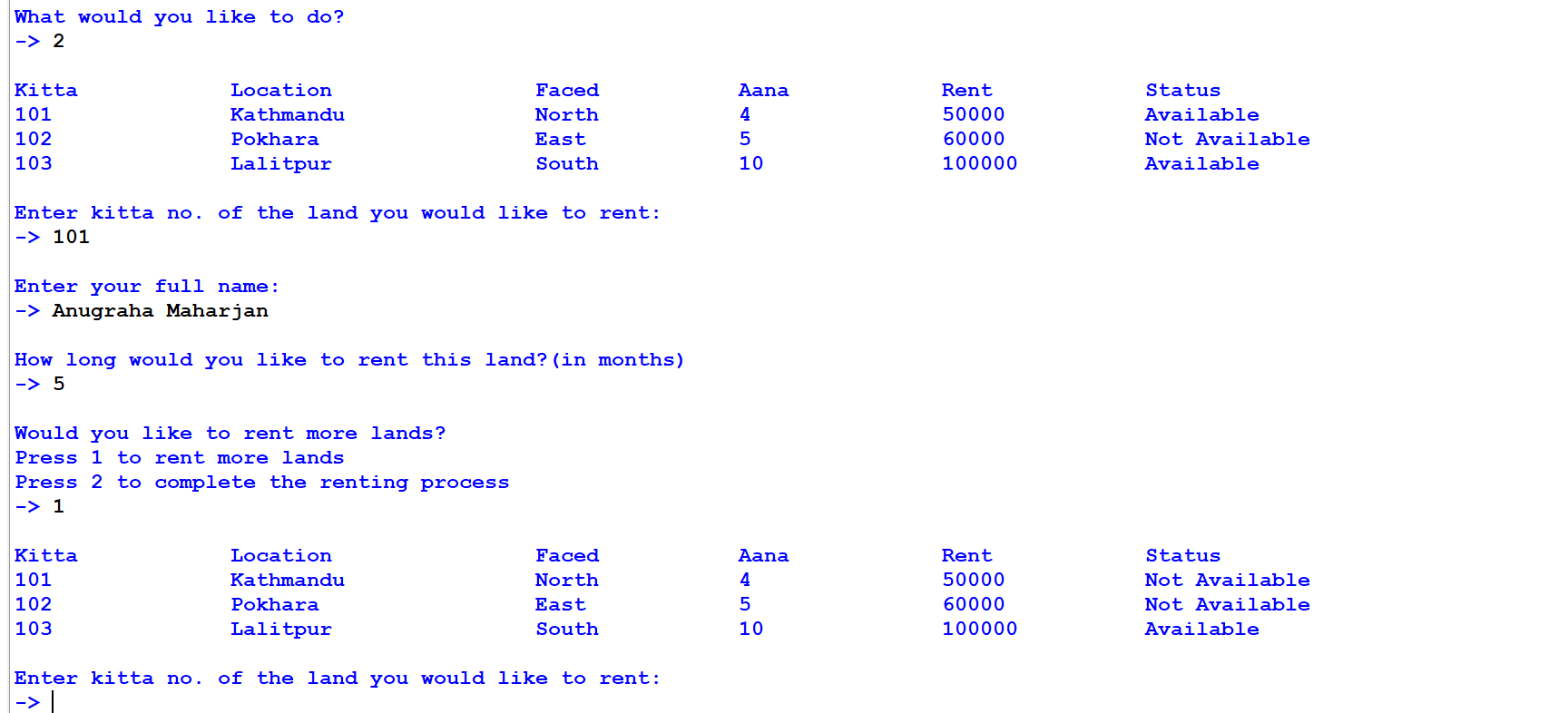


Figure 24: Screenshot after selecting rent more land option

If the user chooses to rent more land, kitta no. is asked.

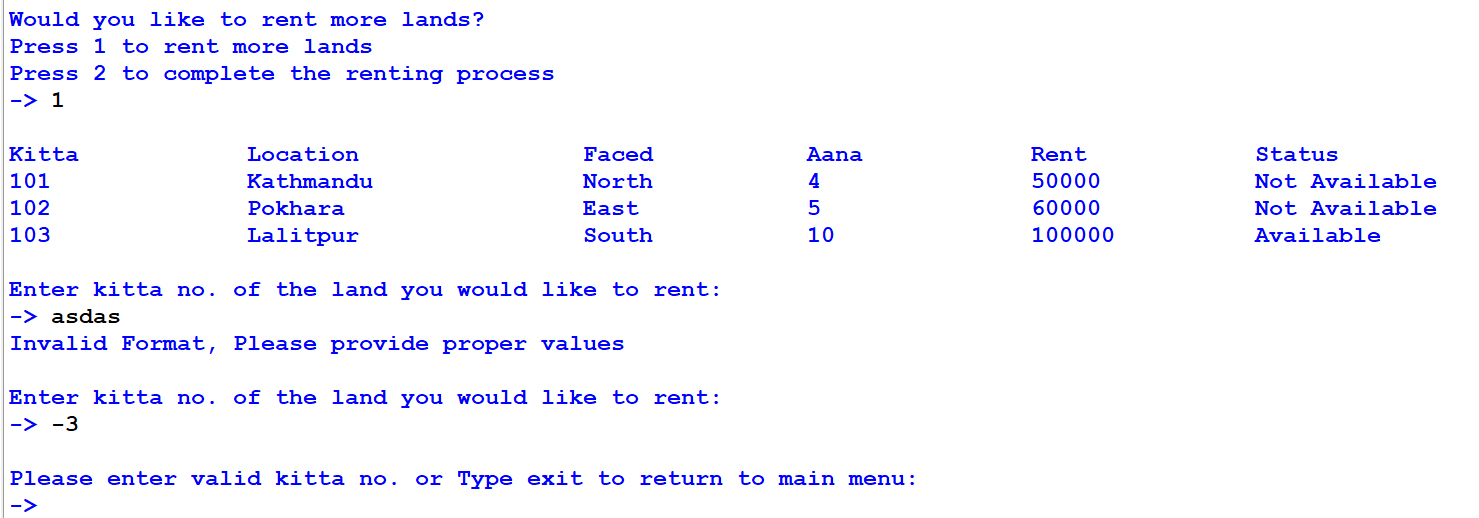


Figure 25: Screenshot of error message after invalid kitta while renting more land

Invalid input will result in error message and kitta no. is asked again.

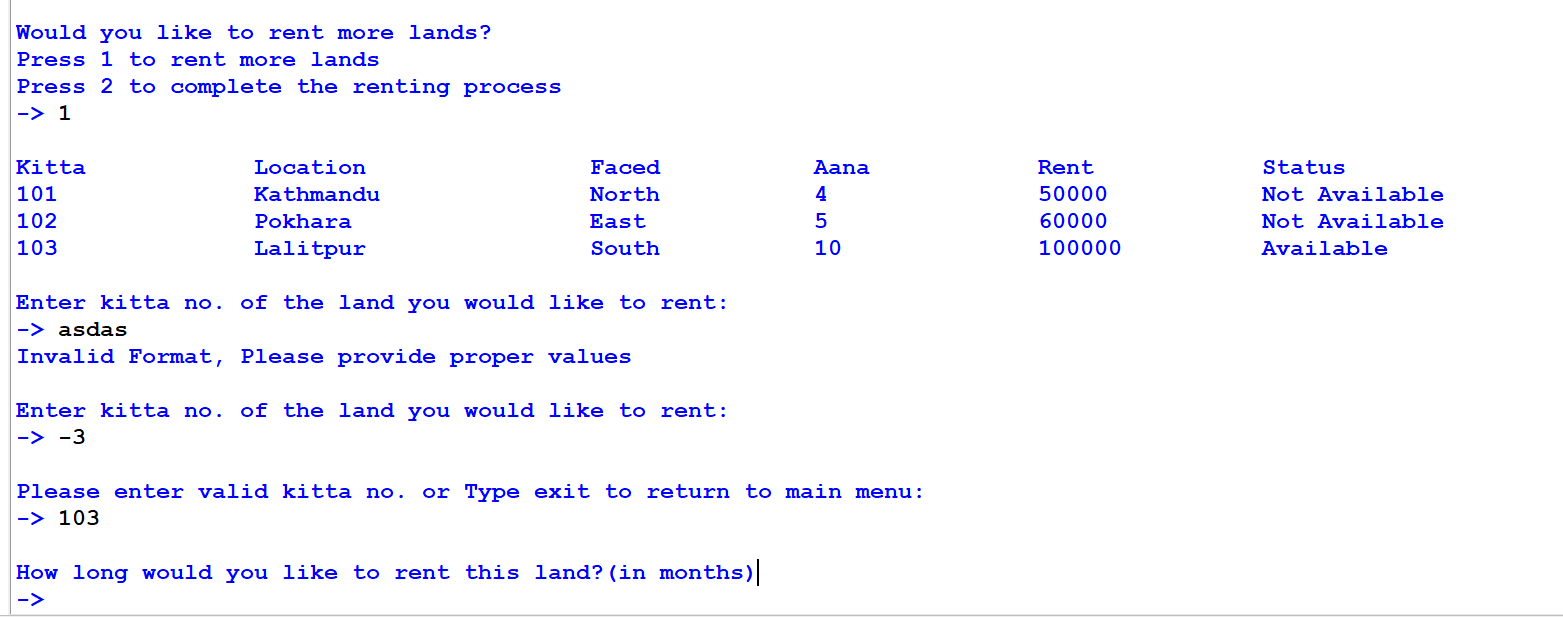


Figure 26: Screenshot after entering valid kitta

When a valid kitta no. is entered, duration of rent is asked.

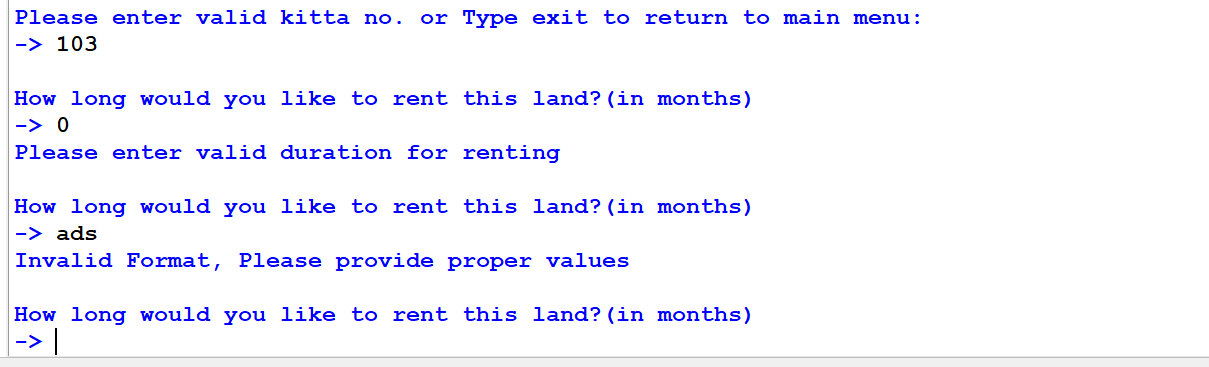


Figure 27: Screenshot of error message after entering invalid duration while renting more land

Invalid input will display error and duration is asked again.

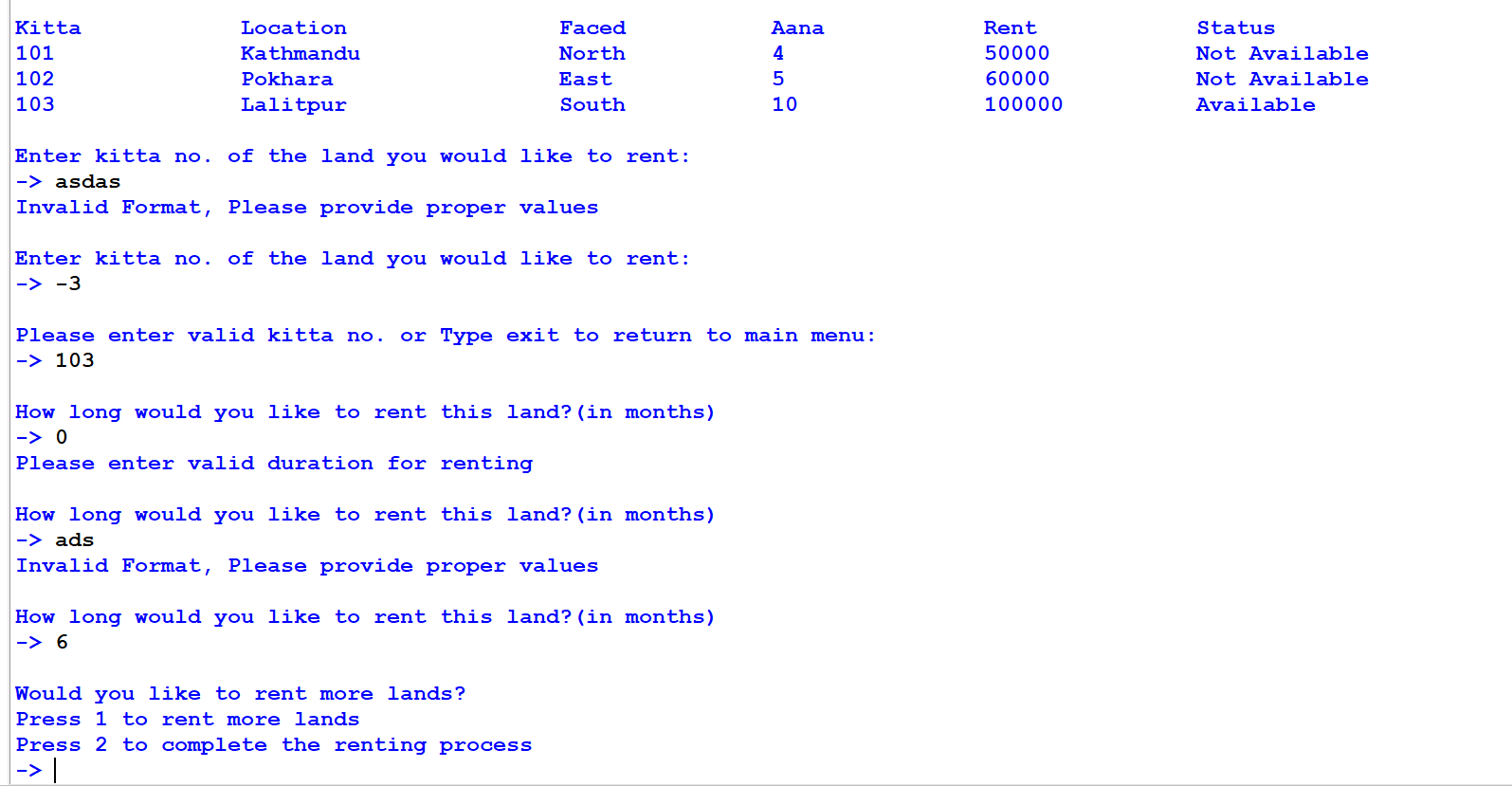


Figure 28: Screenshot after entering valid duration while renting more land

When valid duration is entered, the two choices are displayed

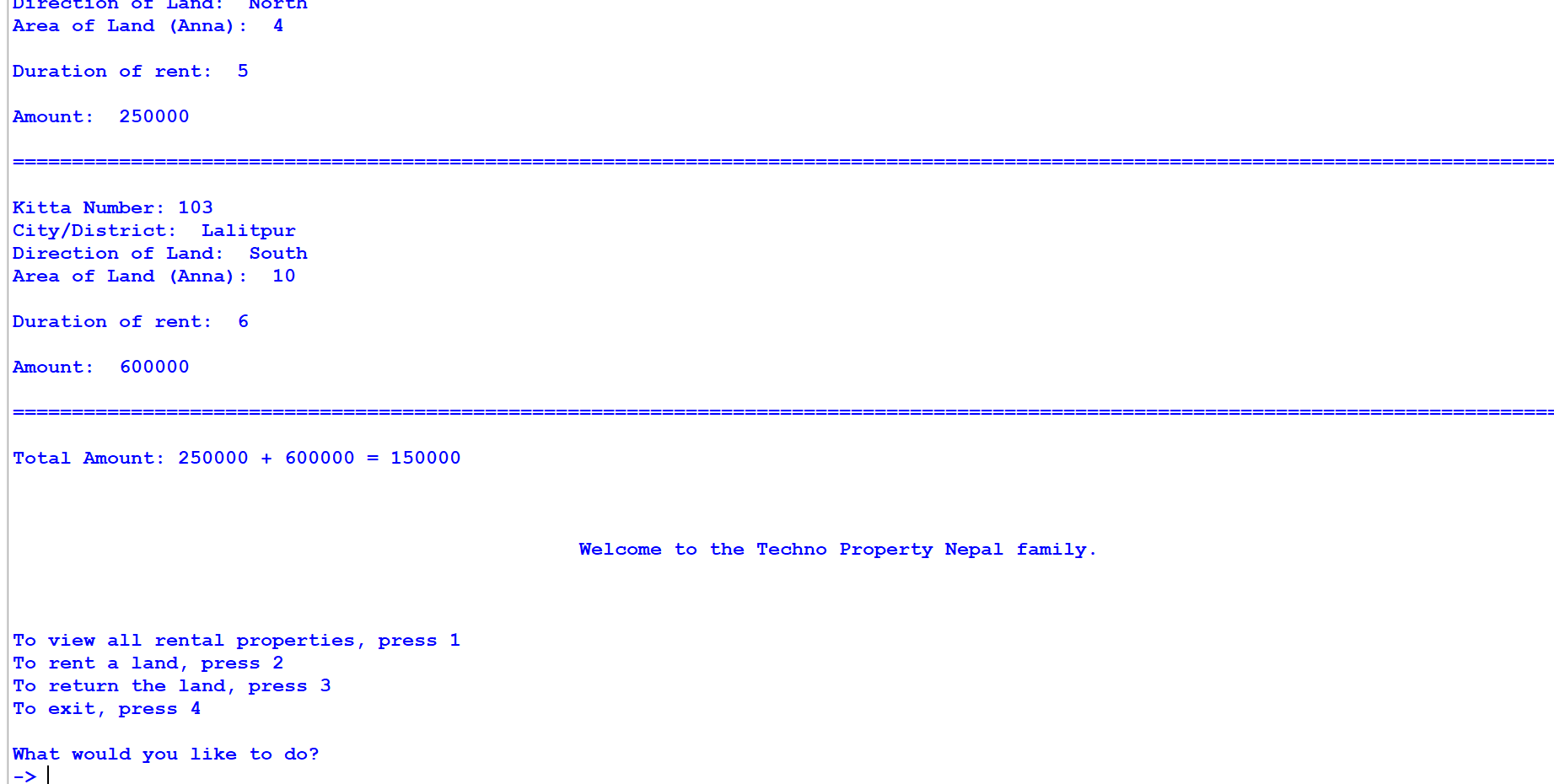
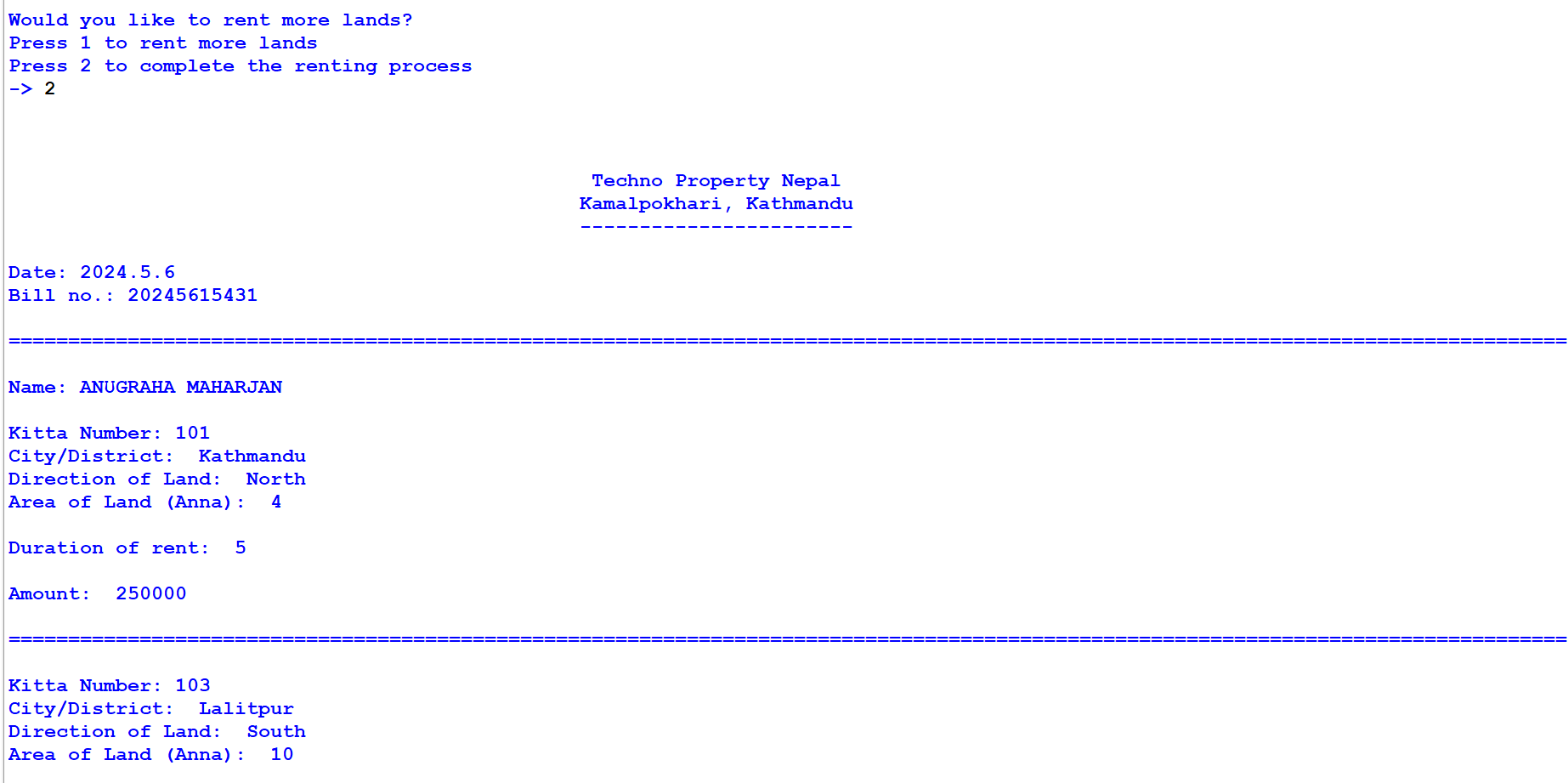


Figure 29: Screenshot of combined bill printed after renting multiple land

When 2 is entered, the combined bill of both lands is printed and user is returned to main menu.

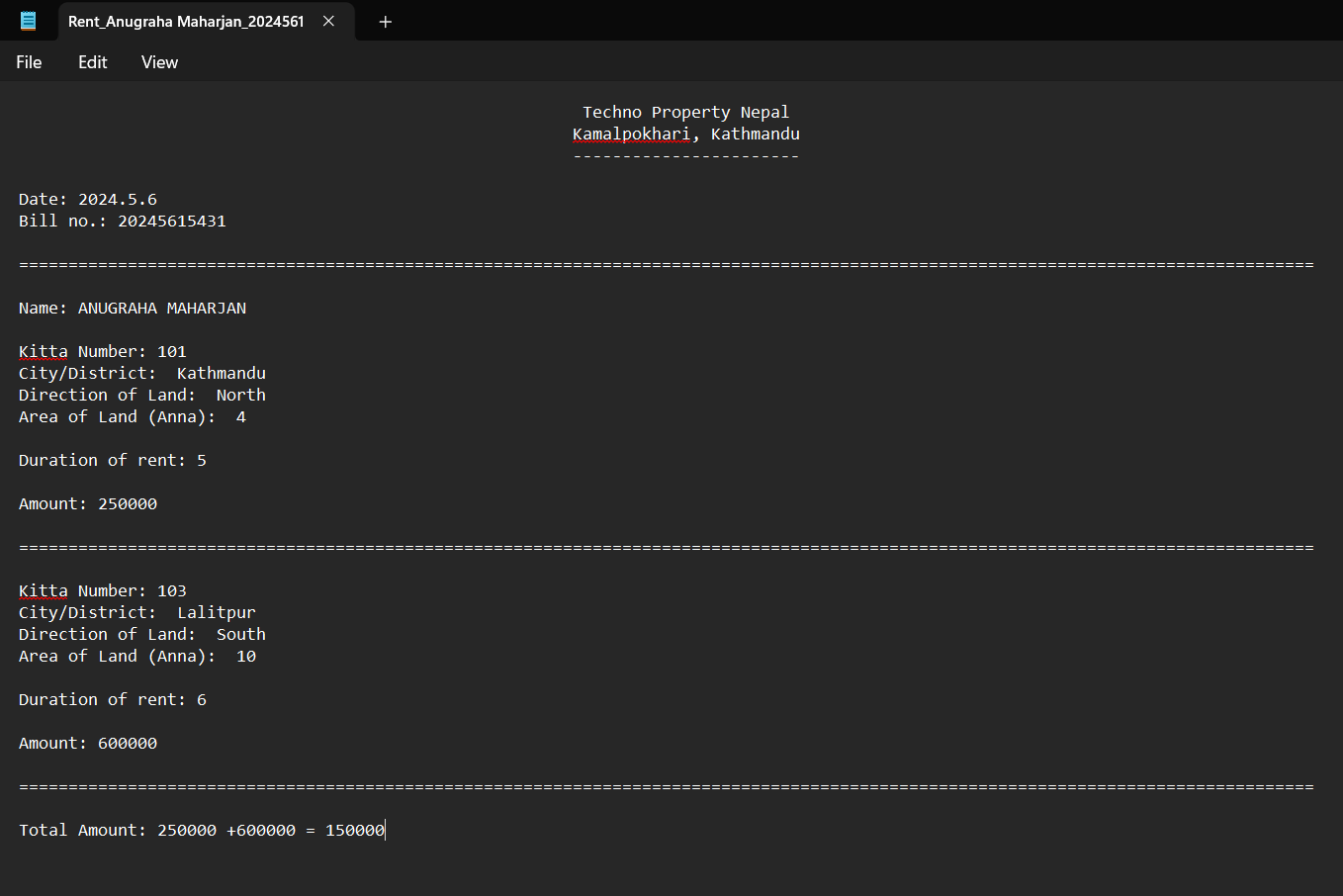


Figure 30: Screenshot of combined bill .txt file created after renting multiple land

A text file of combined bill is also created.

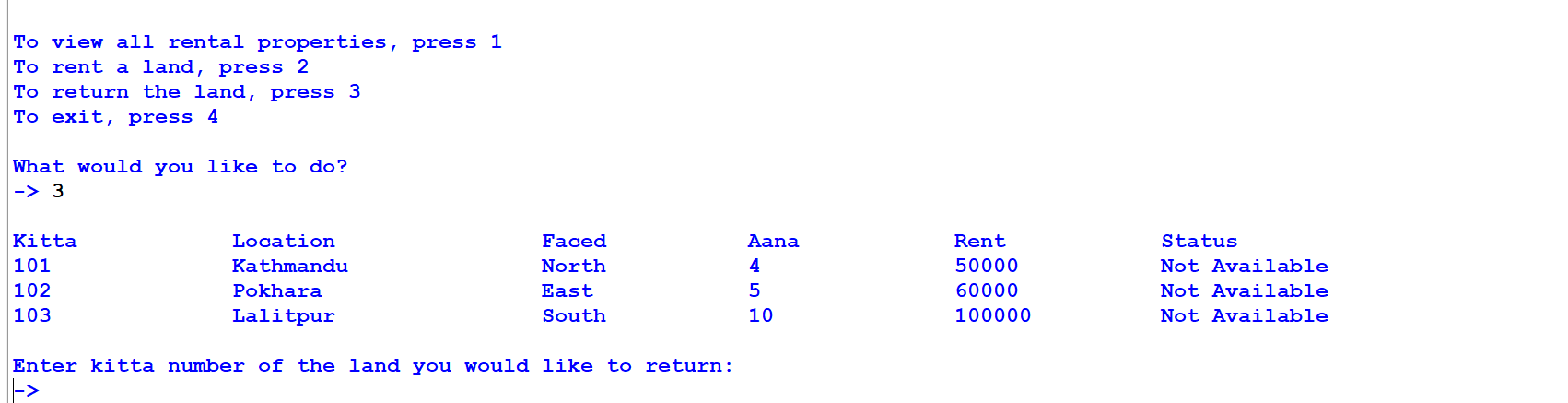


Figure 31: Screenshot of choosing return land option in main menu

Here, the status of the lands are changed from available to not available. If the user enters 3 in main menu, all land’s details are shown and the user is asked to enter kitta no. of the land to be returned.

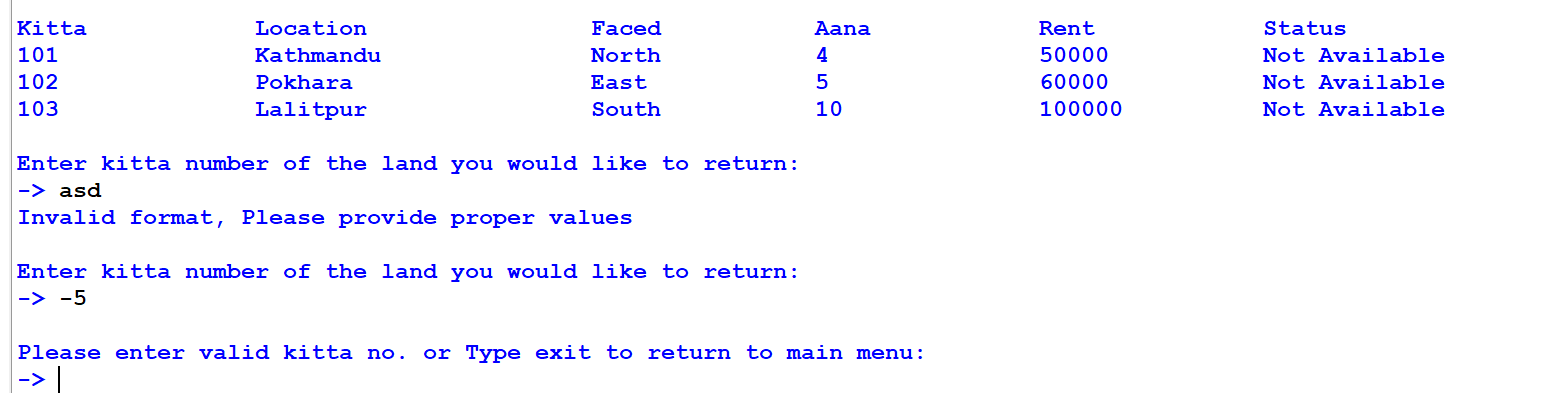


Figure 32: Screenshot of error message after entering invalid kitta while returning land

Invalid input will display error and asks user to enter valid kitta no. or type exit to return to main menu

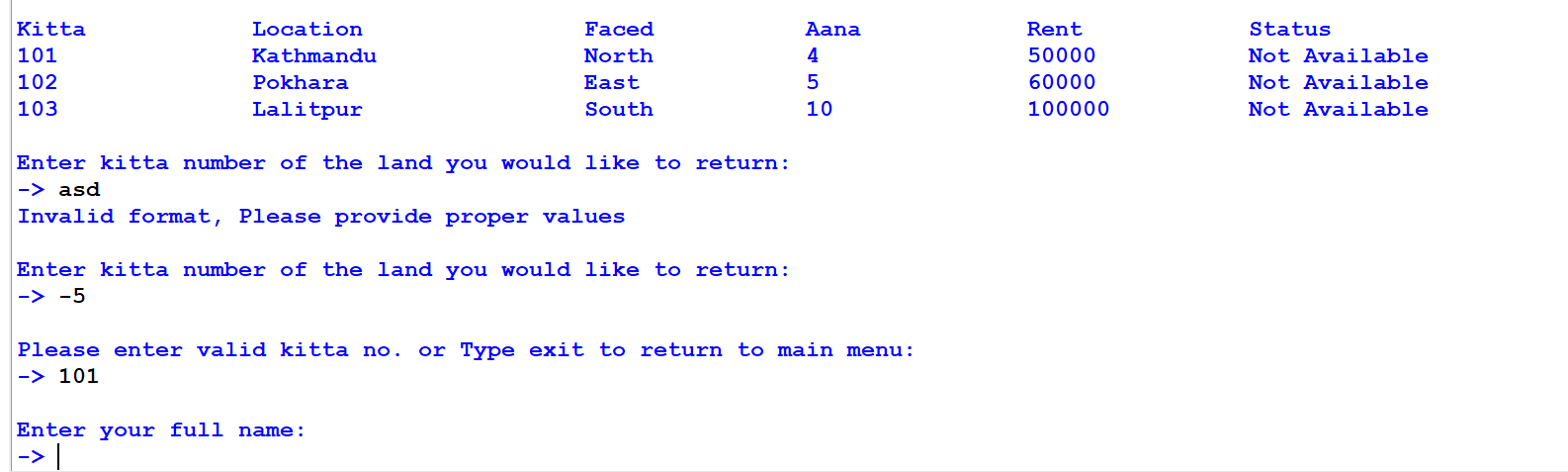


Figure 33: Screenshot after entering valid kitta while returning land

If user enters exit, the system will return to main menu. If the user enters valid kitta number with status unavailable. Full name of the user is asked.

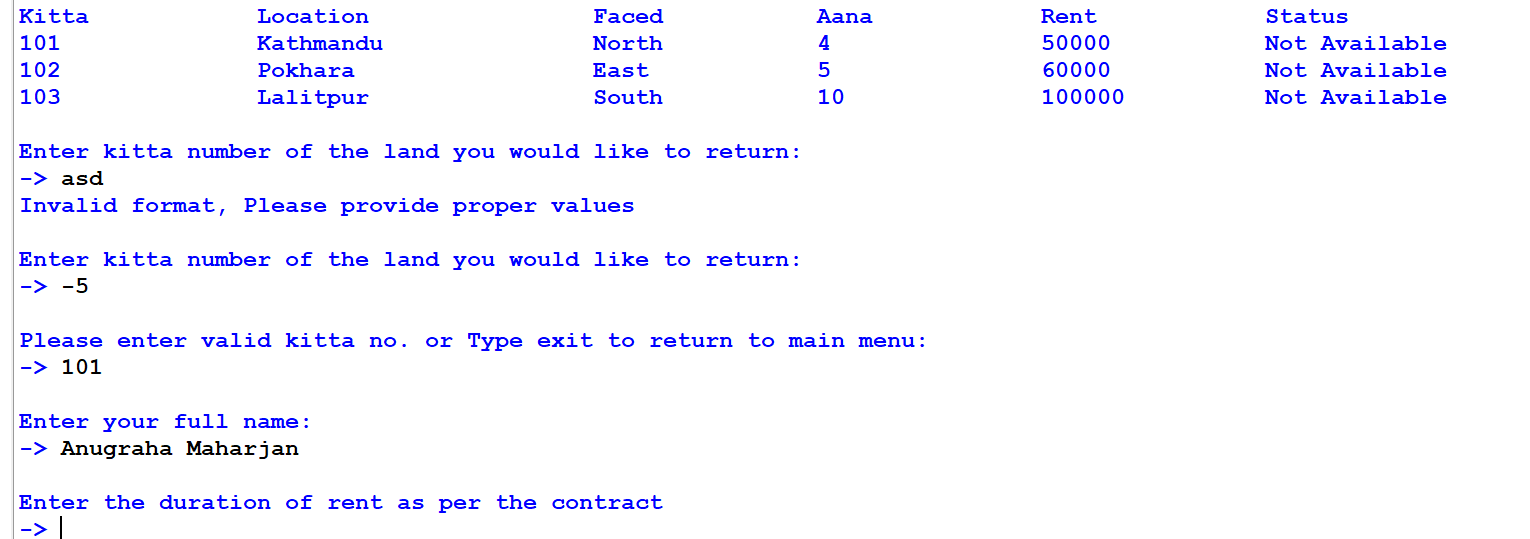


Figure 34: Screenshot after entering name while returning land

Now, duration of rent according to contract is asked.

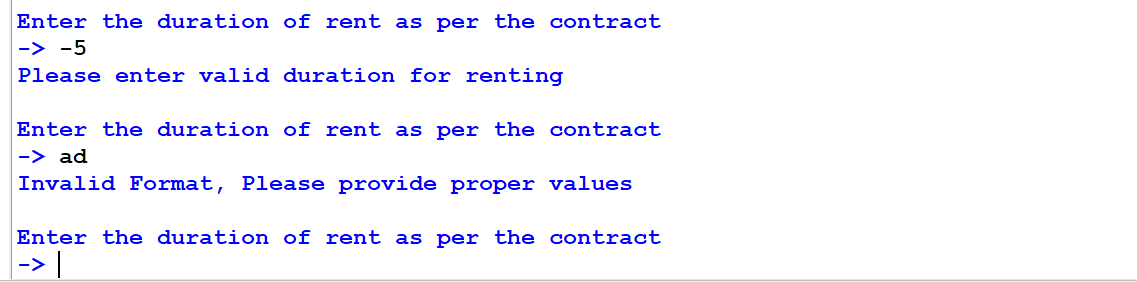


Figure 35: Screenshot after entering invalid duration as per contract while returning land

Invalid input will display error message and ask duration again.

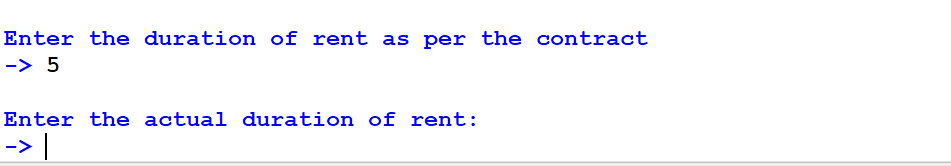


Figure 36: Screenshot after entering valid duration as per contract while returning land

When valid initial duration is entered, actual duration of rent is asked.

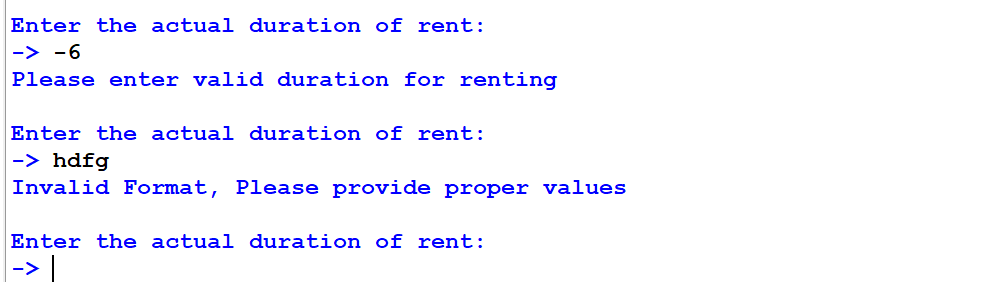


Figure 37: Screenshot after entering invalid actual duration while returning land

Invalid input will display error message and ask actual duration again.

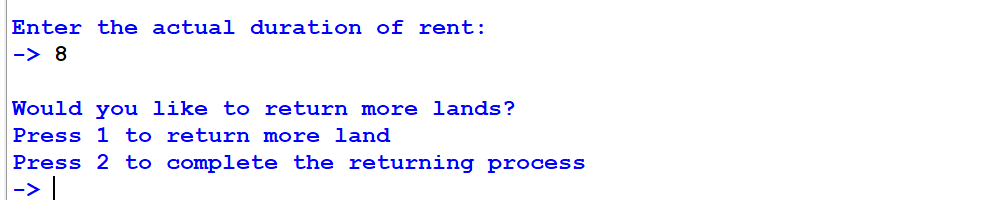


Figure 38: Screenshot after entering valid actual duration while returning land

When valid actual duration is entered, two choices are displayed.

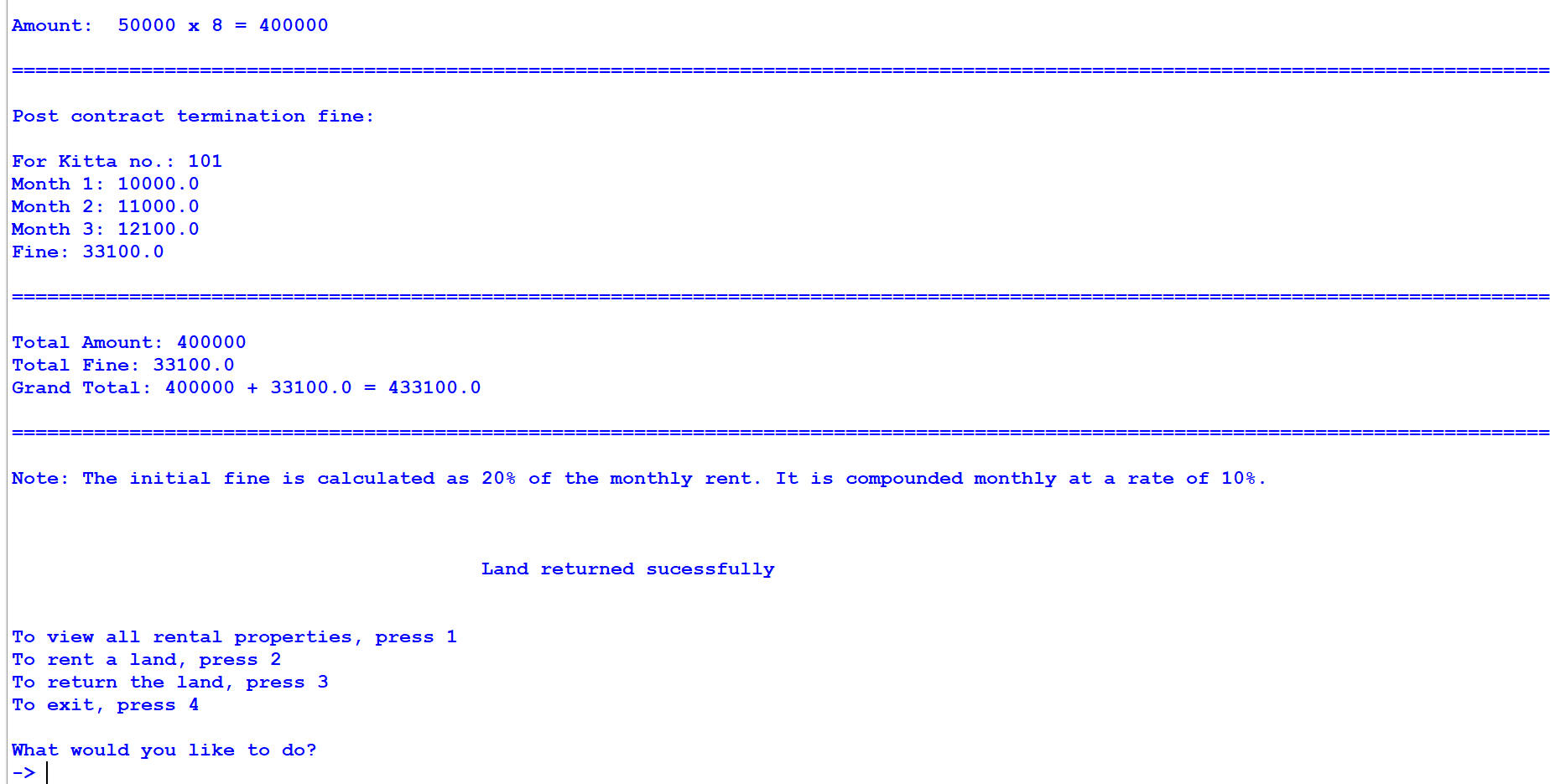
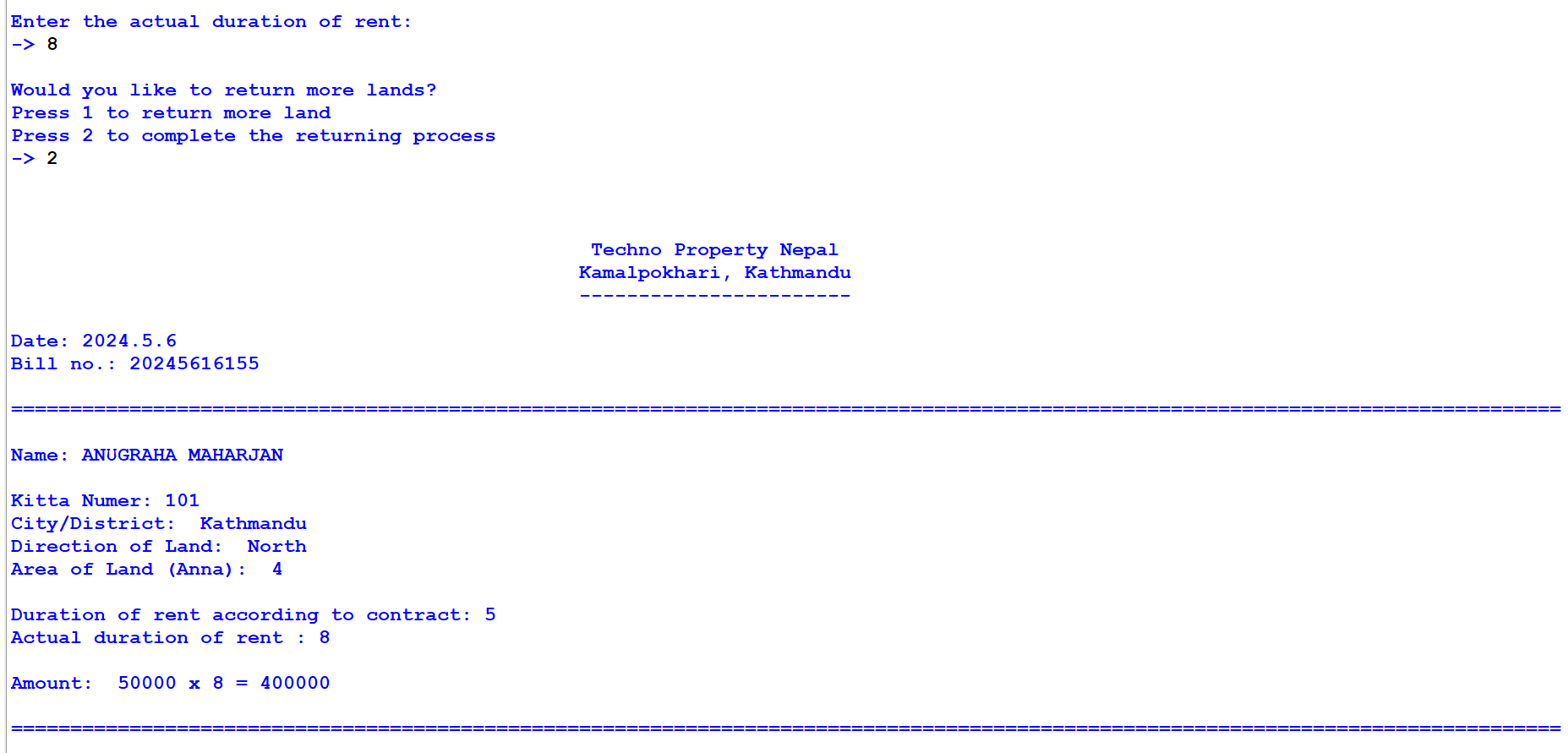


Figure 39: Screenshot of bill printed after returning one land

If the user chooses 2, a bill is printed and system returns to main menu. The bill contains fine which is calculated on monthly basis if the actual duration is more than contract duration.

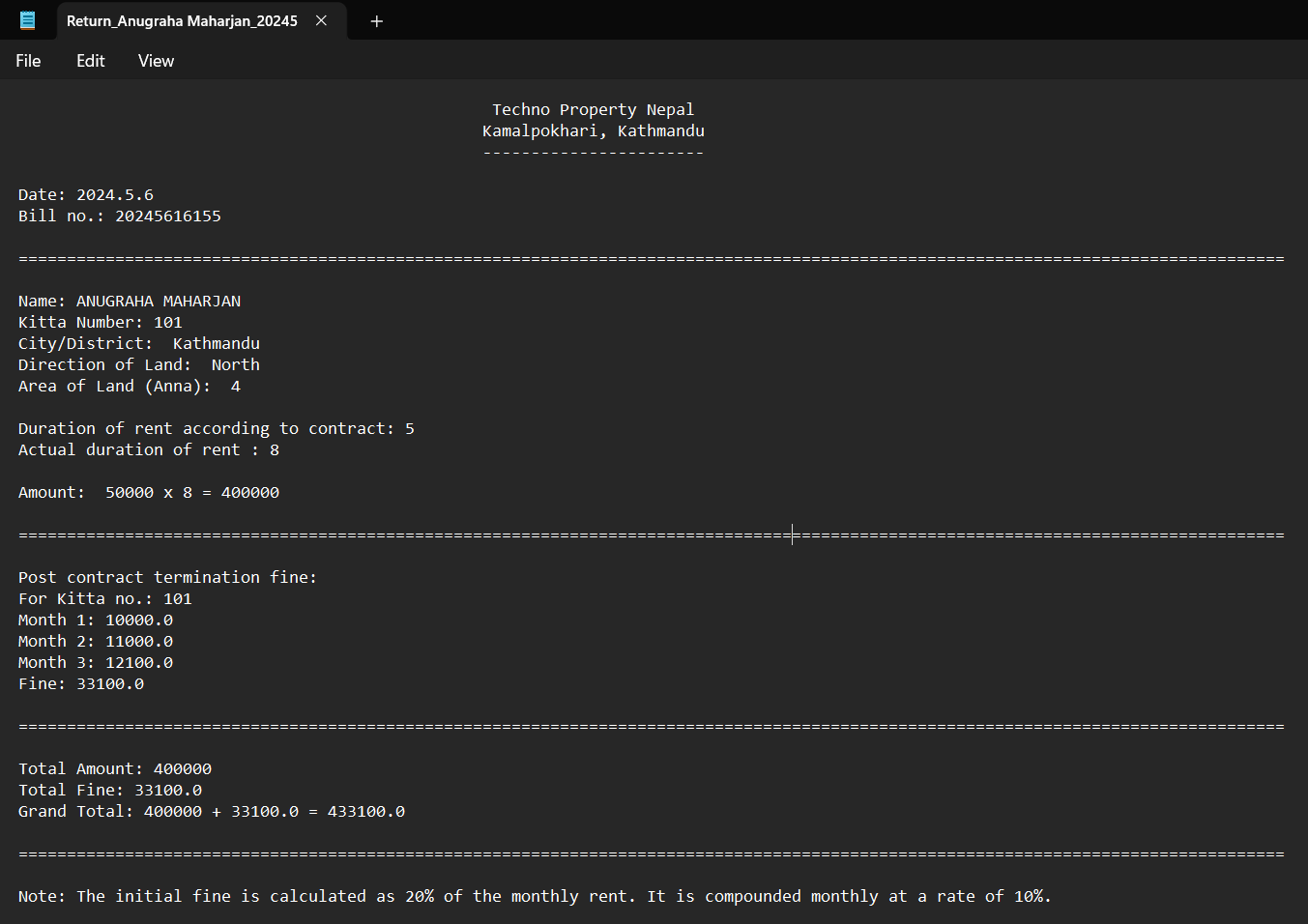


Figure 40: Screenshot of .txt file generated after returning one land

A text file is generated with a unique name.

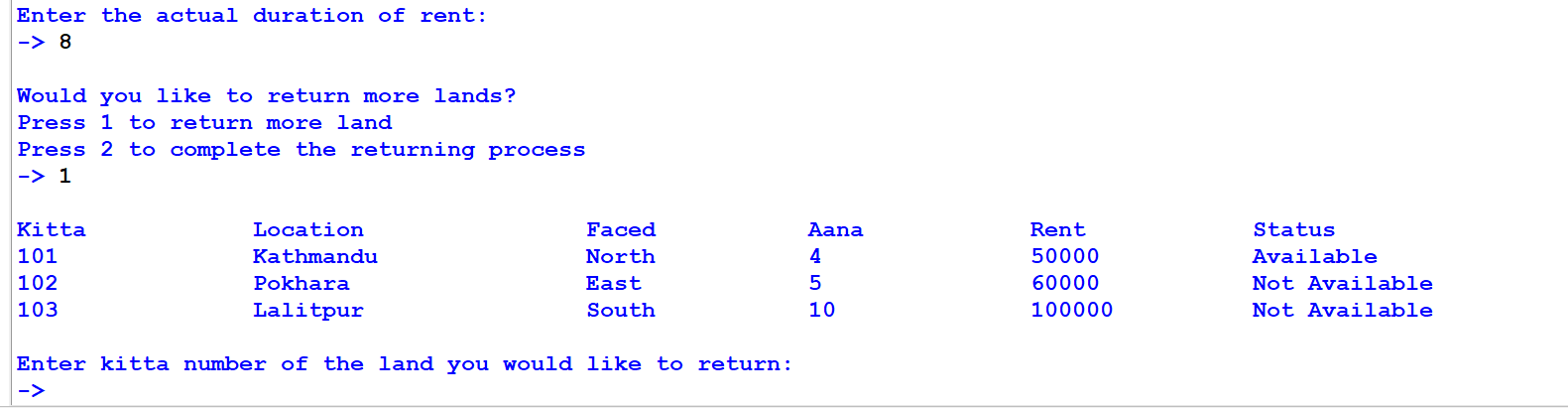


Figure 41: Screenshot after choosing to return more land

If the user chooses 1, the system displays all land’s details and asks to enter kitta no.

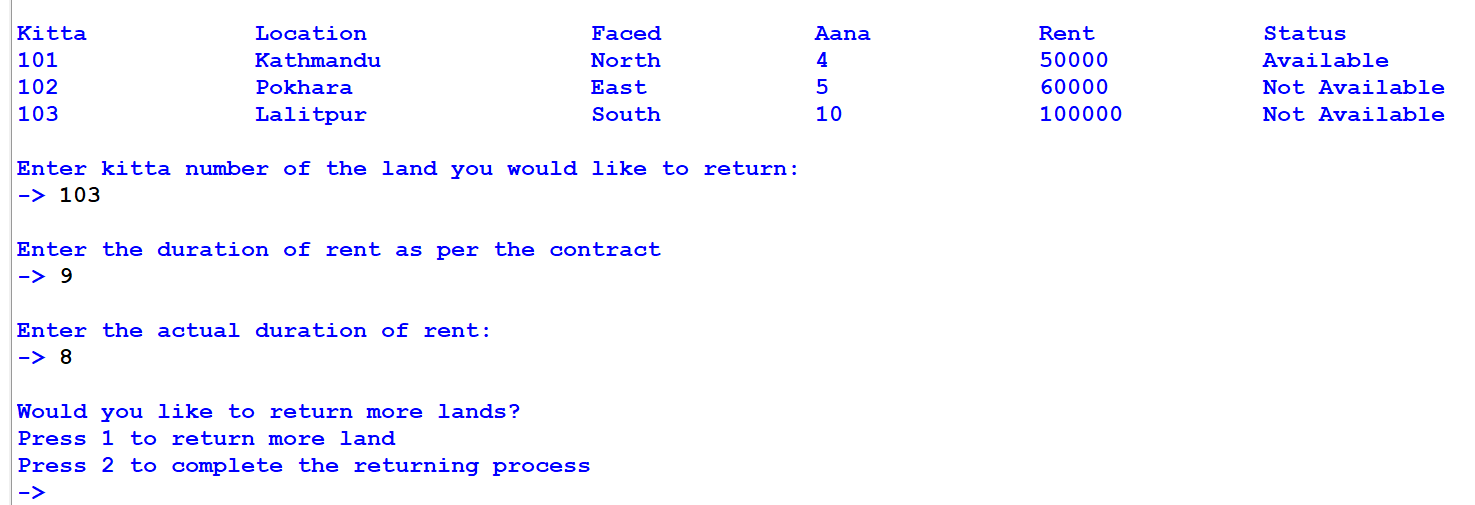


Figure 42: Screenshot after completing all fields while returning multiple land

When all fields are completed with valid values. The system again shows two choices.

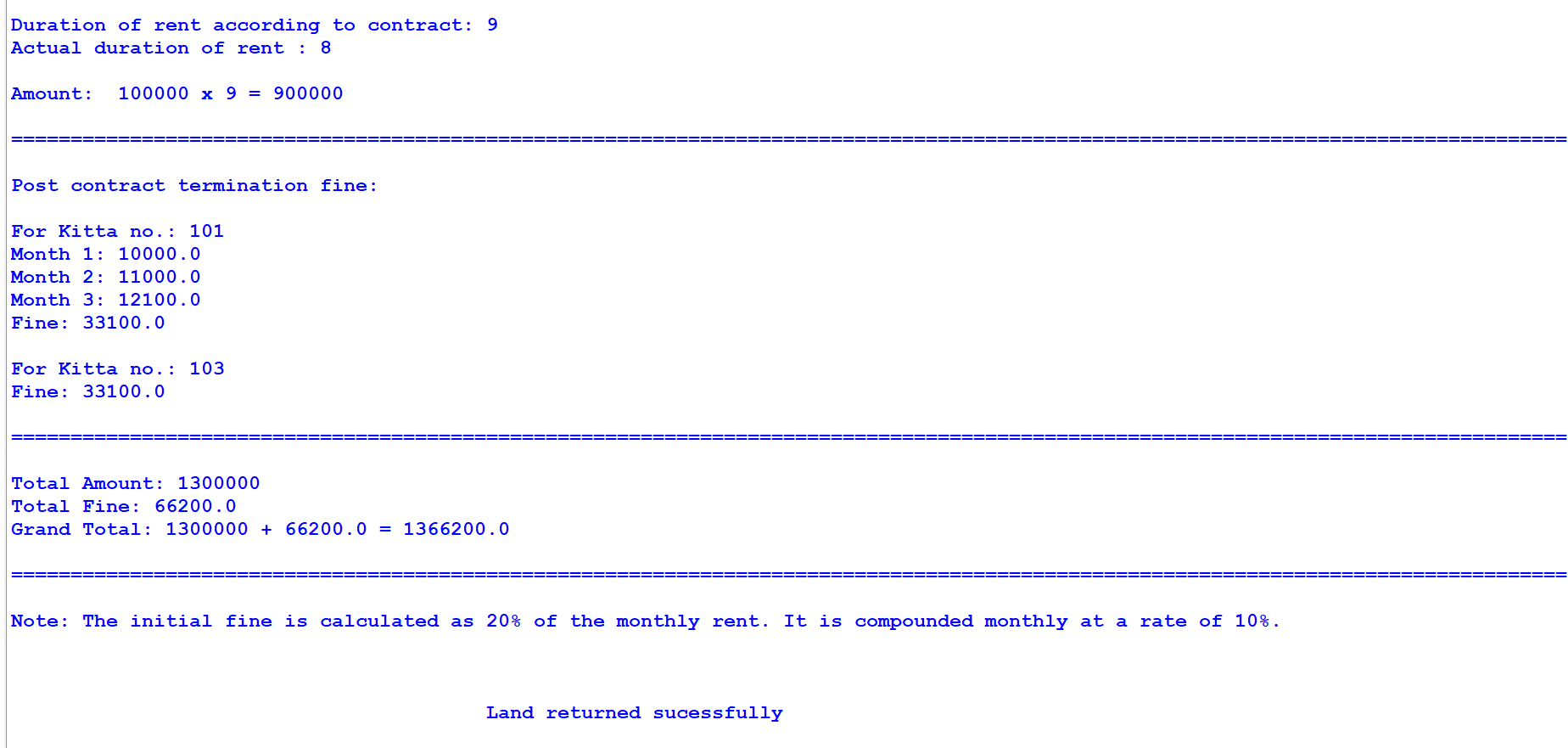
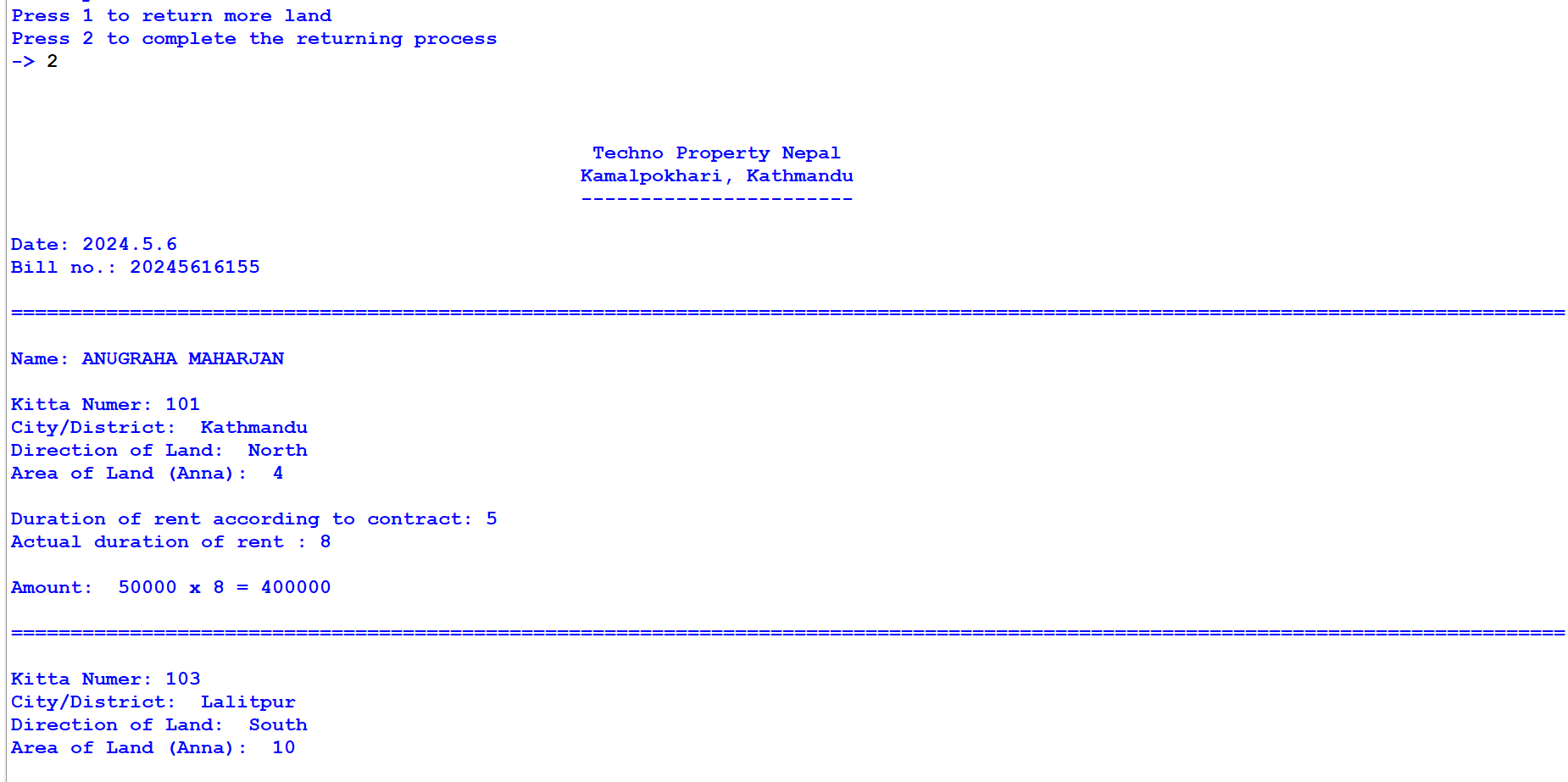


Figure 43: Screenshot of bill printed after returning multiple land

If the user chooses 2, a combined bill for both lands returned is printed and user is returned back to main menu.

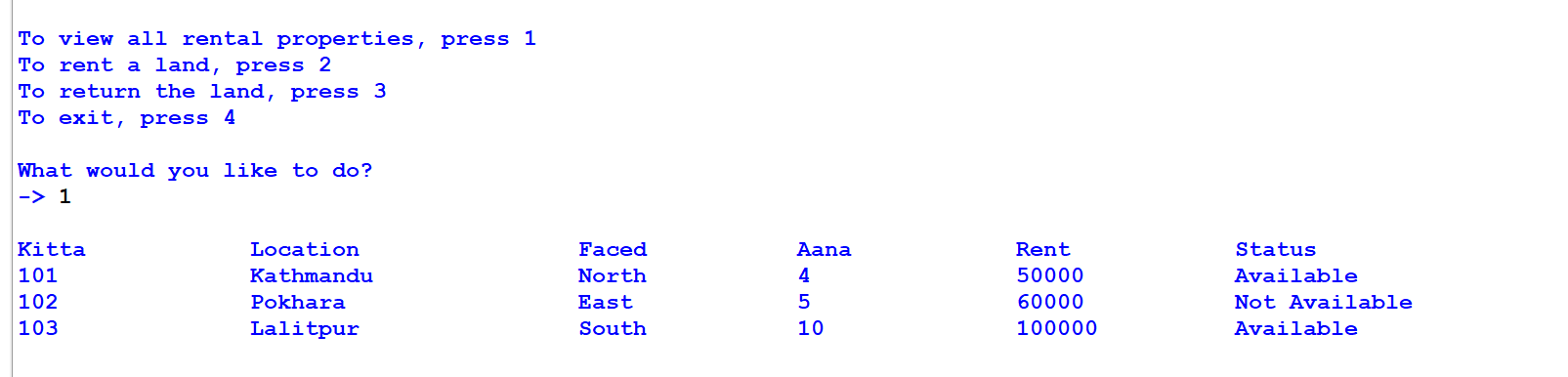


Figure 44: Screenshot of status of lands changed after returning

The status is also changes from not available to available.

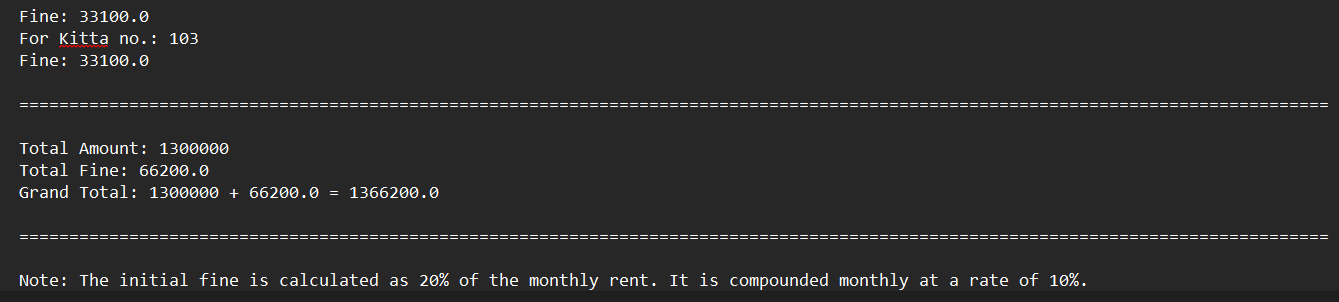
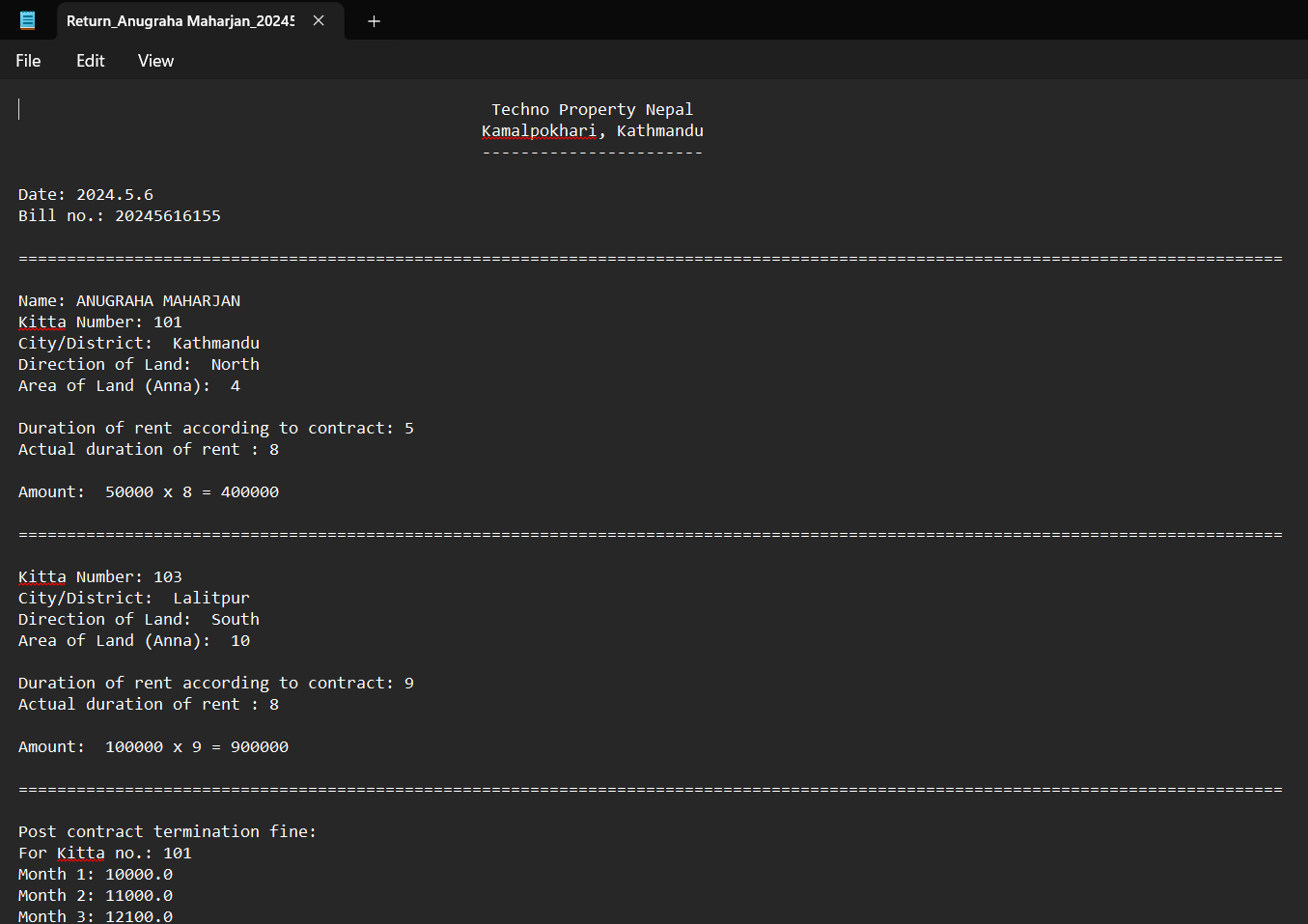


Figure 45: Screenshot of .txt file generated after returning multiple lands

A text file is also created for combined bill.

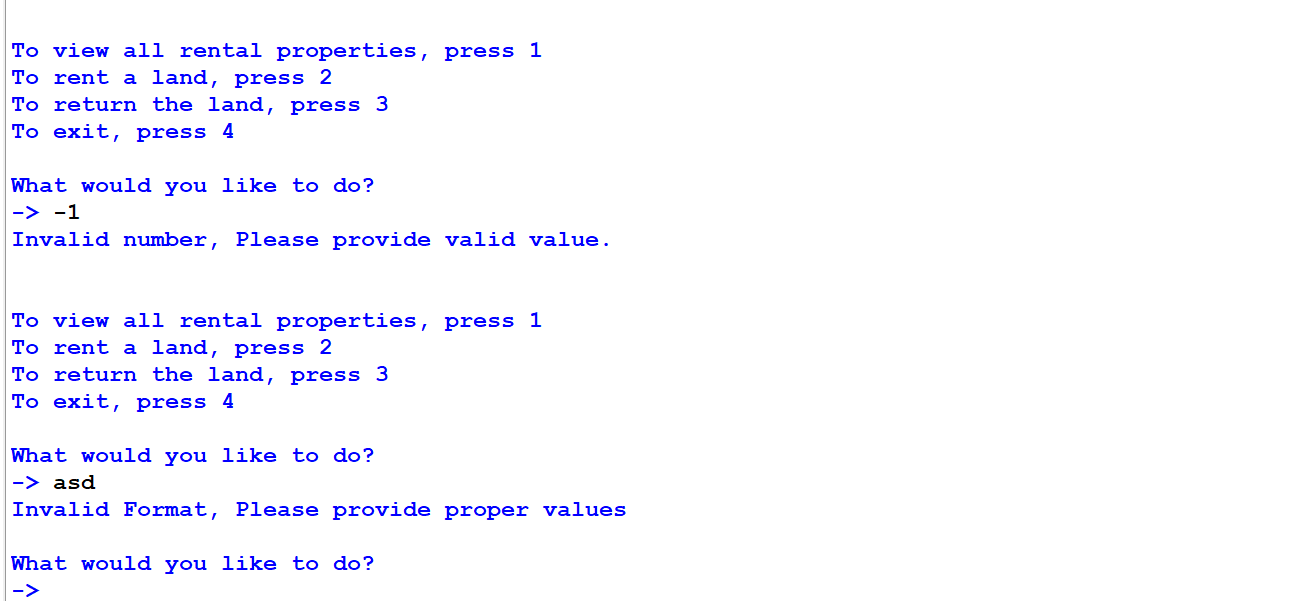


Figure 46: Screenshot after entering invalid reply in main menu

In main menu, if user enters invalid reply. Error message is shown and user is asked to enter reply again.

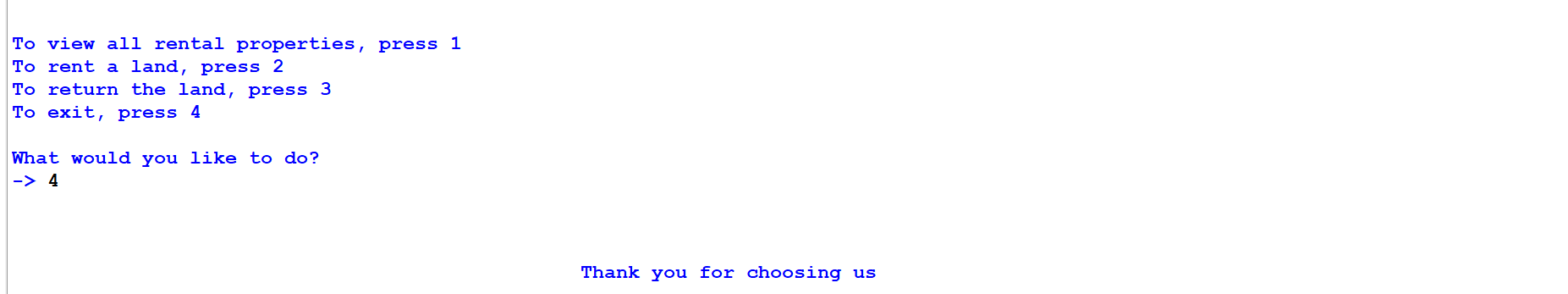


Figure 47: Screenshot after choosing to exit in main menu

If user enters 4, a short message is displayed and the system is closed.

# Testing

## Test 1: Implementation of try and except

|  |  |
| --- | --- |
| **Test no:** | 1 |
| **Objective:** | To test implementation of try and except. |
| **Action:** | String value is entered where kitta no. is to be entered. |
| **Expected Result:** | An error message will be displayed and user will be asked to enter kitta no. again. |
| **Actual Result:** | An error message was displayed and user was asked to enter kitta no. again. |
| **Conclusion:** | The test was successful. |

Table 1: Implementation of try and except test

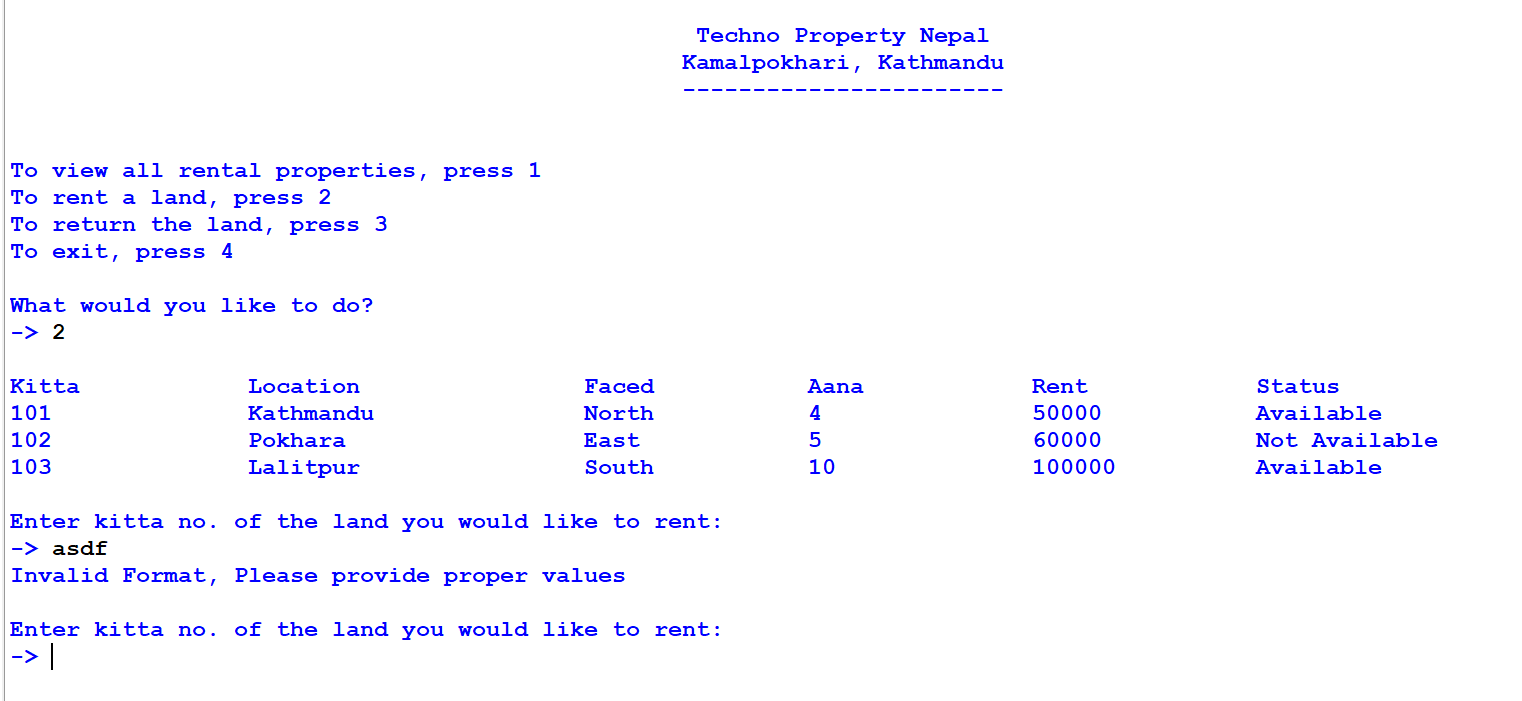


Figure 48: Screenshot of try except test

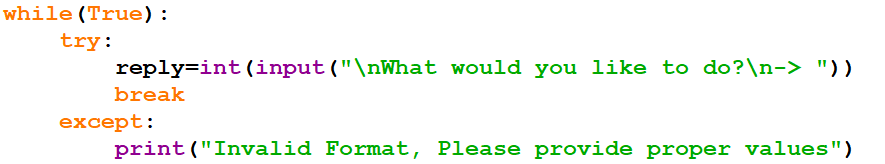


Figure 49: Screenshot of code of try except

## Test 2: Invalid input while Rent and Return of land

### 4.2.1. Test 2.1: Providing invalid value while renting lands

|  |  |
| --- | --- |
| **Test no:** | 2.1 |
| **Objective:** | To test working of renting land process with invalid value |
| **Action:** | * Negative value was provided as input. * Not existing value was provided as input |
| **Expected Result:** | An error message will be displayed and user will be asked to enter kitta no. again. |
| **Actual Result:** | An error message was displayed and user was asked to enter kitta no. again. |
| **Conclusion:** | The test was successful. |

Table 2: Invalid value input while renting test

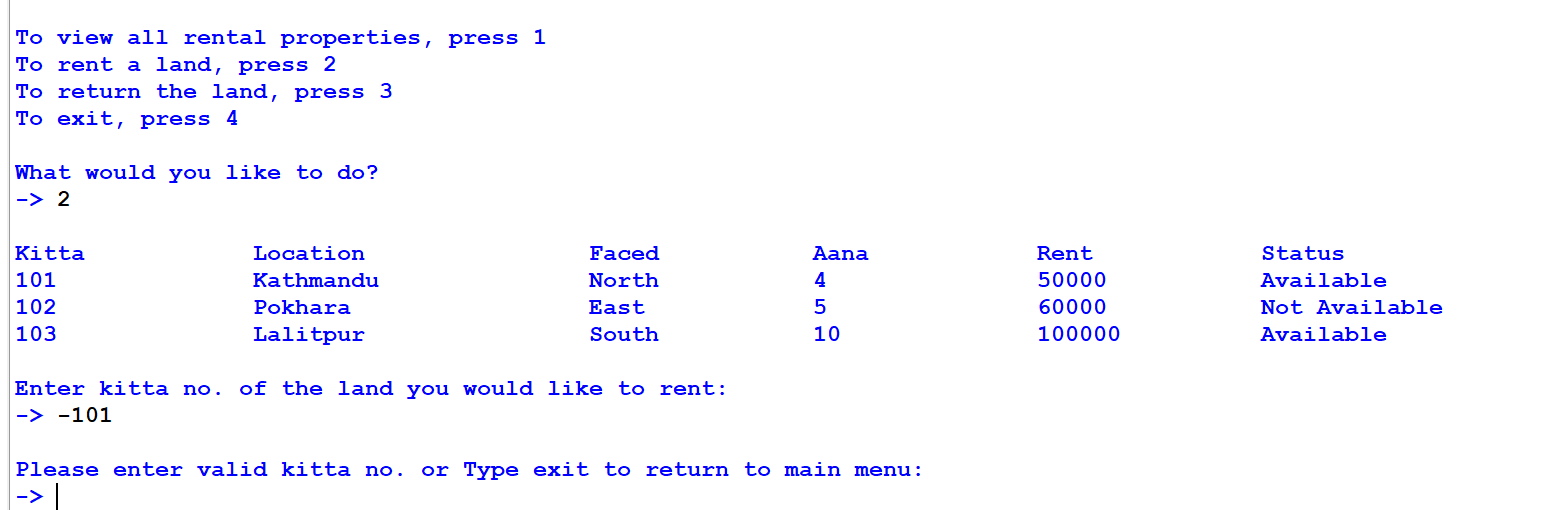


Figure 50: Screenshot of providing negative value while renting

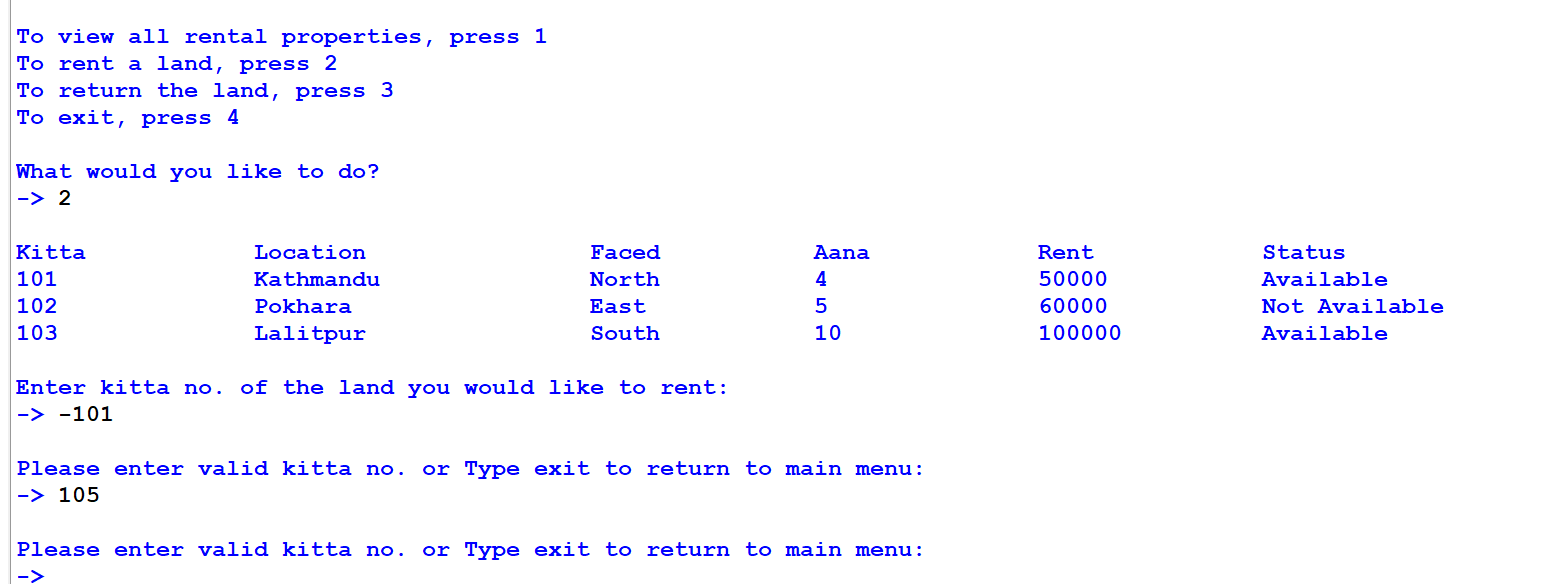


Figure 51: Screenshot of providing non existing value while renting

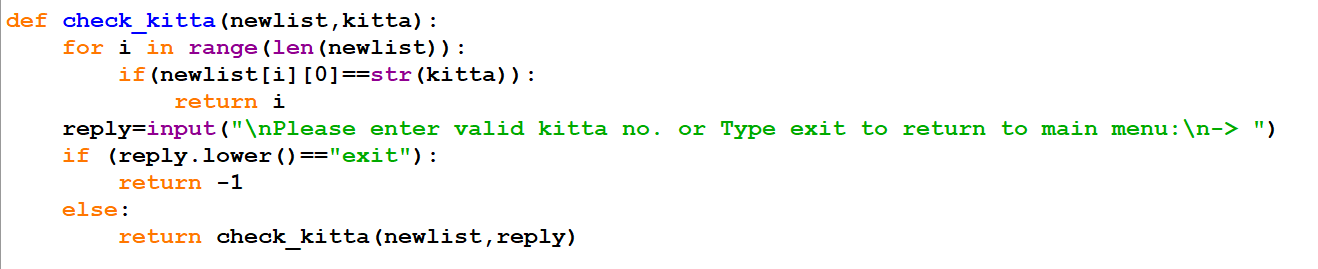


Figure 52: Screenshot of code of checking invalid input

### 4.2.2. Test 2.2: Providing invalid value while returning lands

|  |  |
| --- | --- |
| **Test no:** | 2.2 |
| **Objective:** | To test working of returning land process with invalid value |
| **Action:** | * Negative value was provided as input. * Not existing value was provided as input |
| **Expected Result:** | An error message will be displayed and user will be asked to enter kitta no. again. |
| **Actual Result:** | An error message was displayed and user was asked to enter kitta no. again. |
| **Conclusion:** | The test was successful. |

Table 3: Invalid value input while returning test

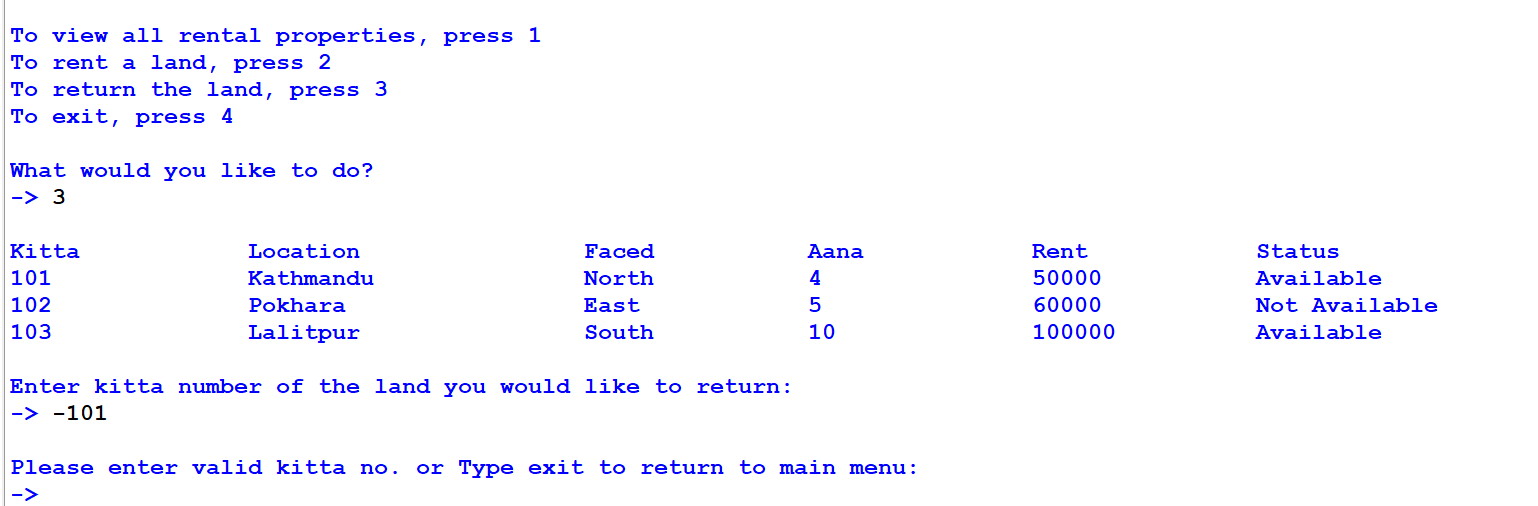


Figure 53: Screenshot of providing negative value while returning land

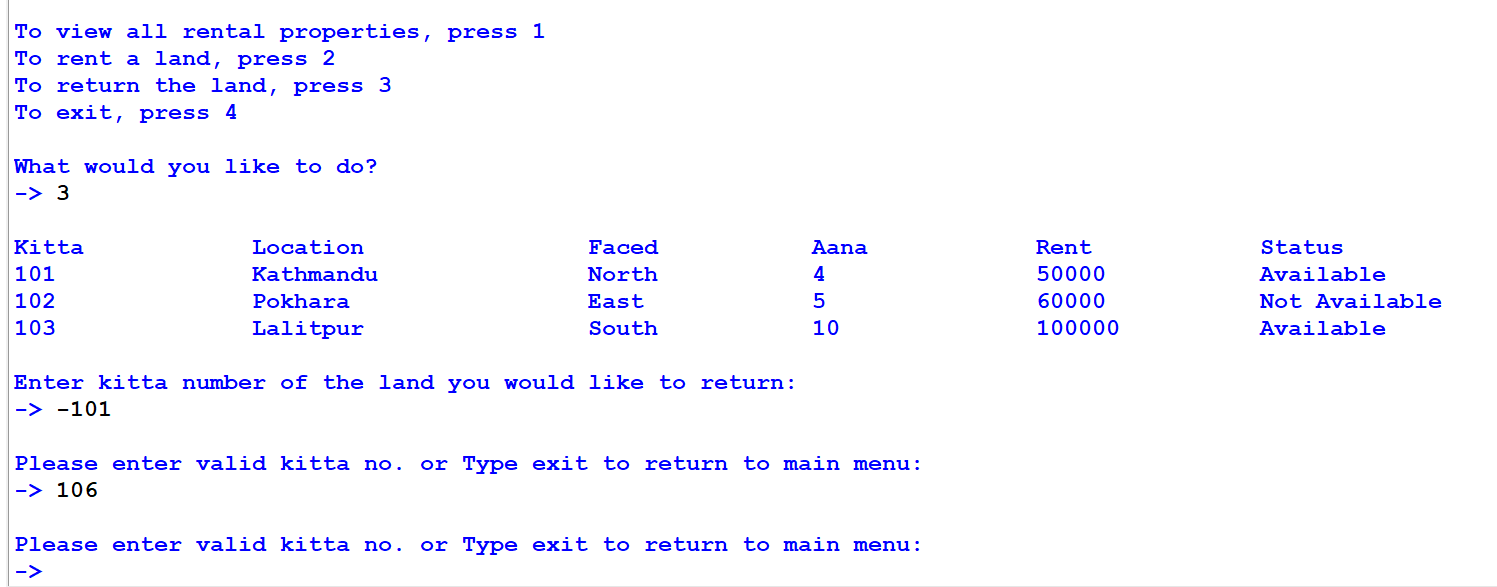


Figure 54: Screenshot of providing non existing value while returning land

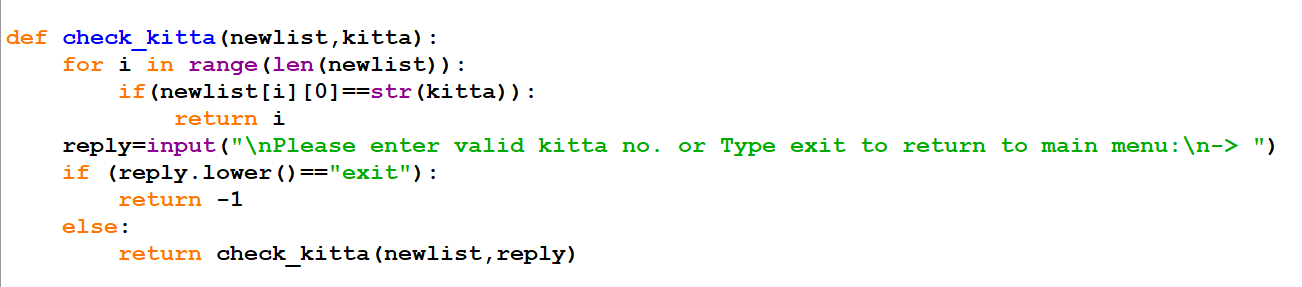


Figure 55: Screenshot of code to check invalid input

## Test 3: Invoice generation after renting multiple lands

|  |  |
| --- | --- |
| **Test no:** | 3 |
| **Objective:** | To test the generation of invoice after renting multiple lands. |
| **Action:** | * Provide all necessary information and rent a land. * Enter 1 and rent another land. * Choose 2 and complete the renting process. |
| **Expected Result:** | A combined bill for both lands will be printed and a .txt file will also be generated for it. |
| **Actual Result:** | A combined bill for both lands was printed and a .txt file was also generated for it. |
| **Conclusion:** | The test was successful. |

Table 4: Invoice generation after renting multiple lands test

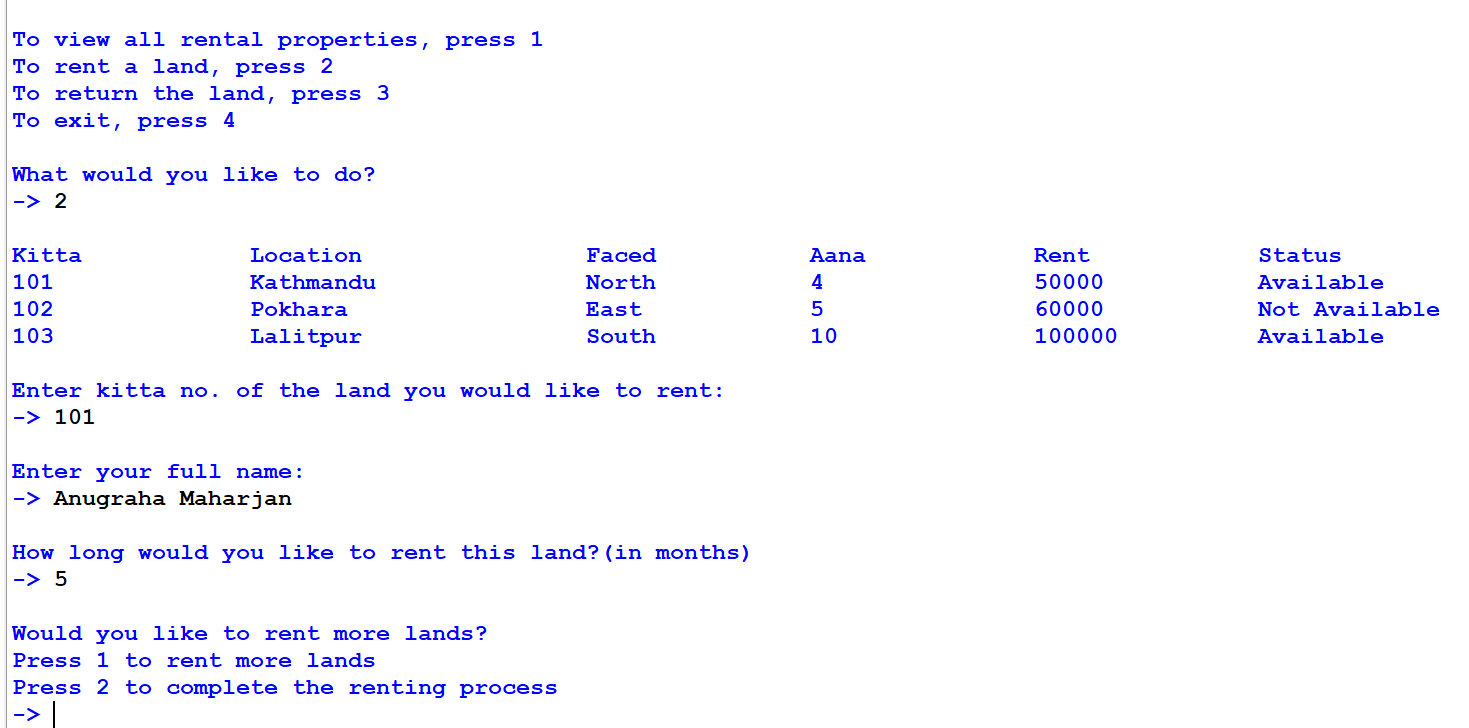


Figure 56: Screenshot of renting land one

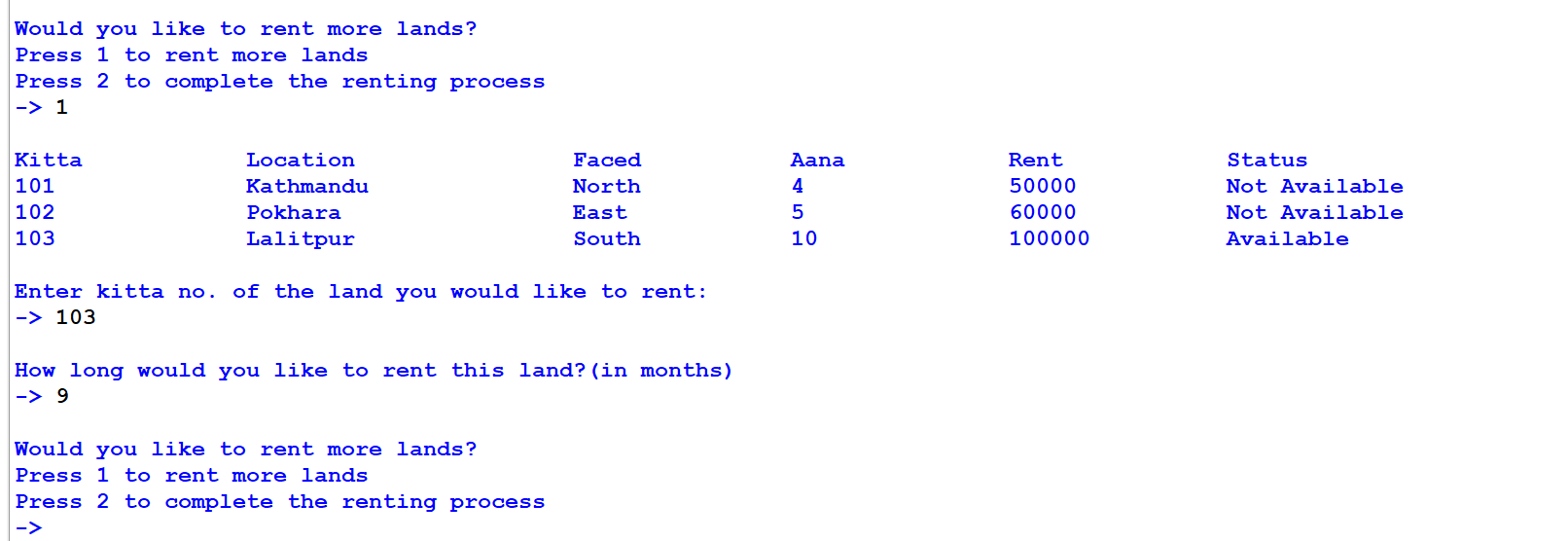
****

Figure 57: Screenshot of renting multiple land

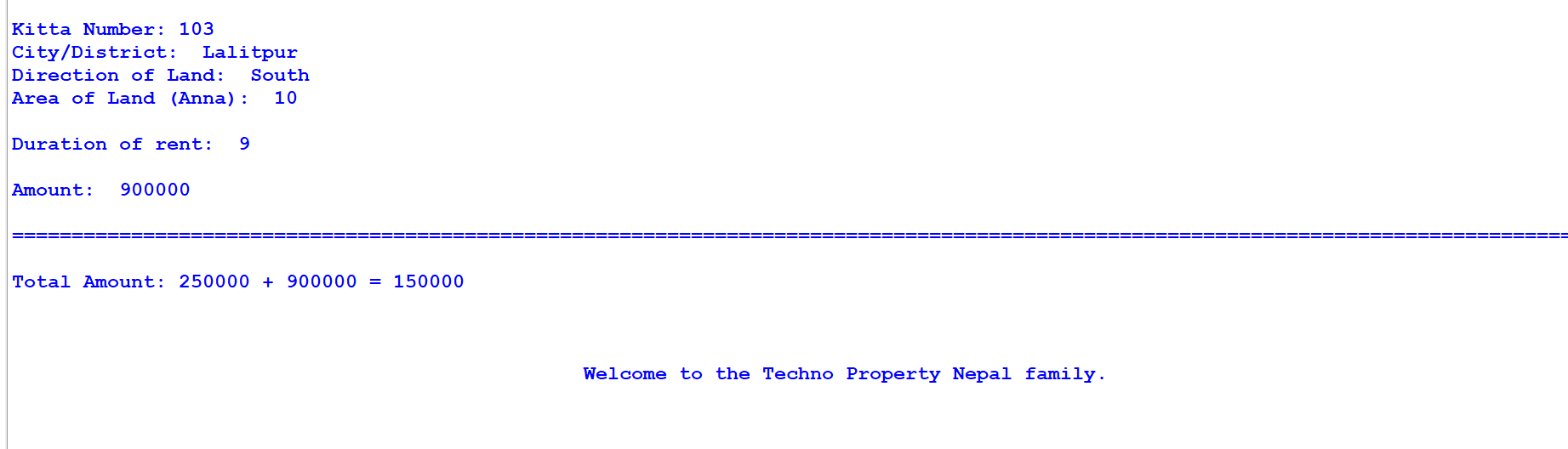
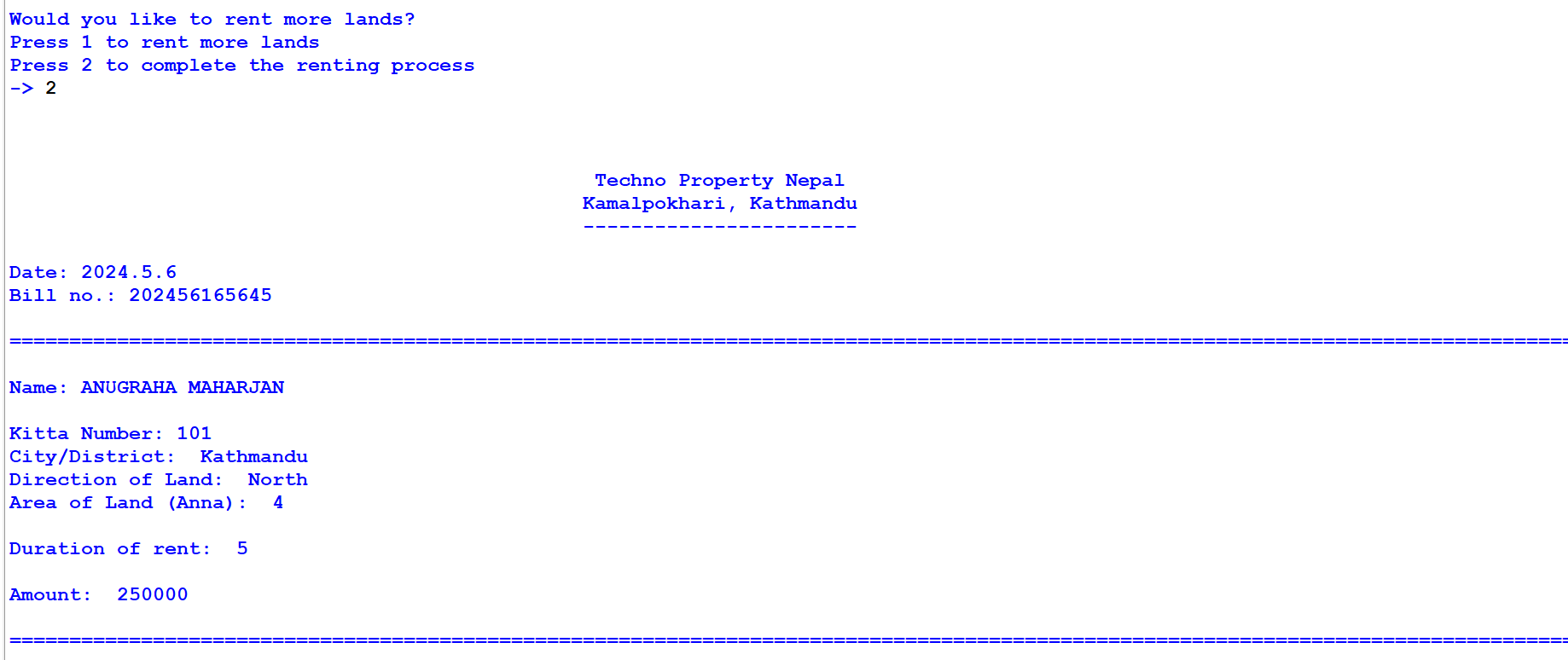


Figure 58: Screenshot of bill printed after renting multiple land



Figure 59: Screenshot of code that prints bill while renting multiple land

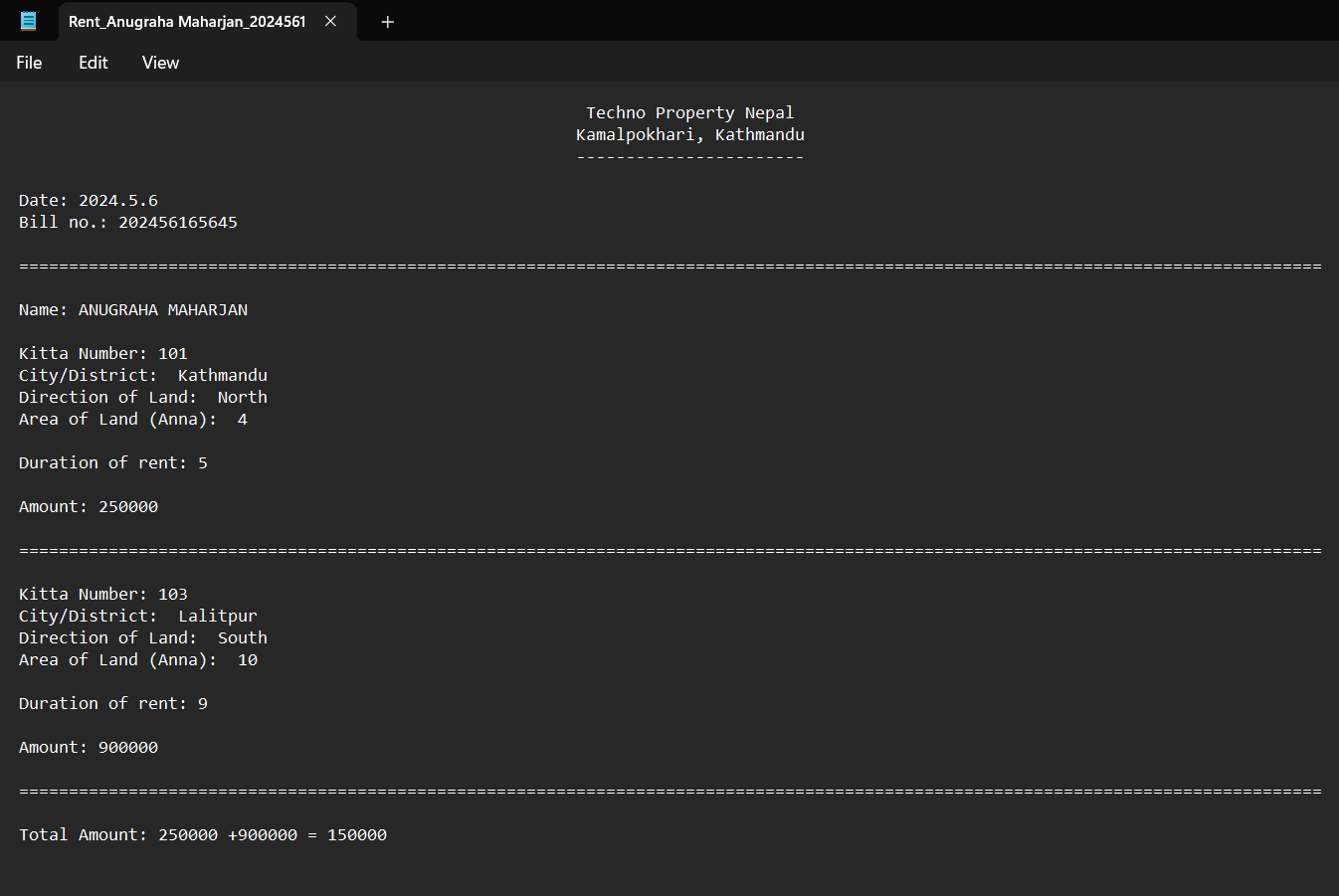


Figure 60: Screenshot of .txt file generated after renting multiple lands

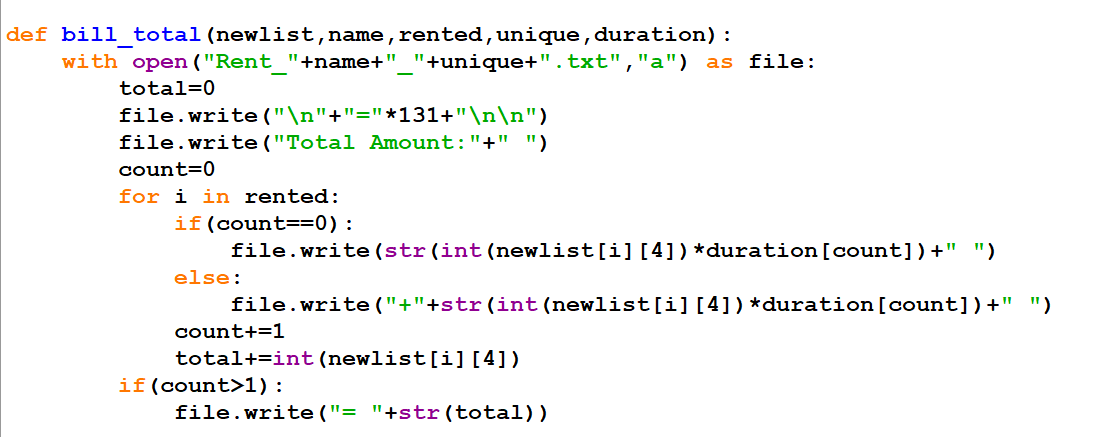
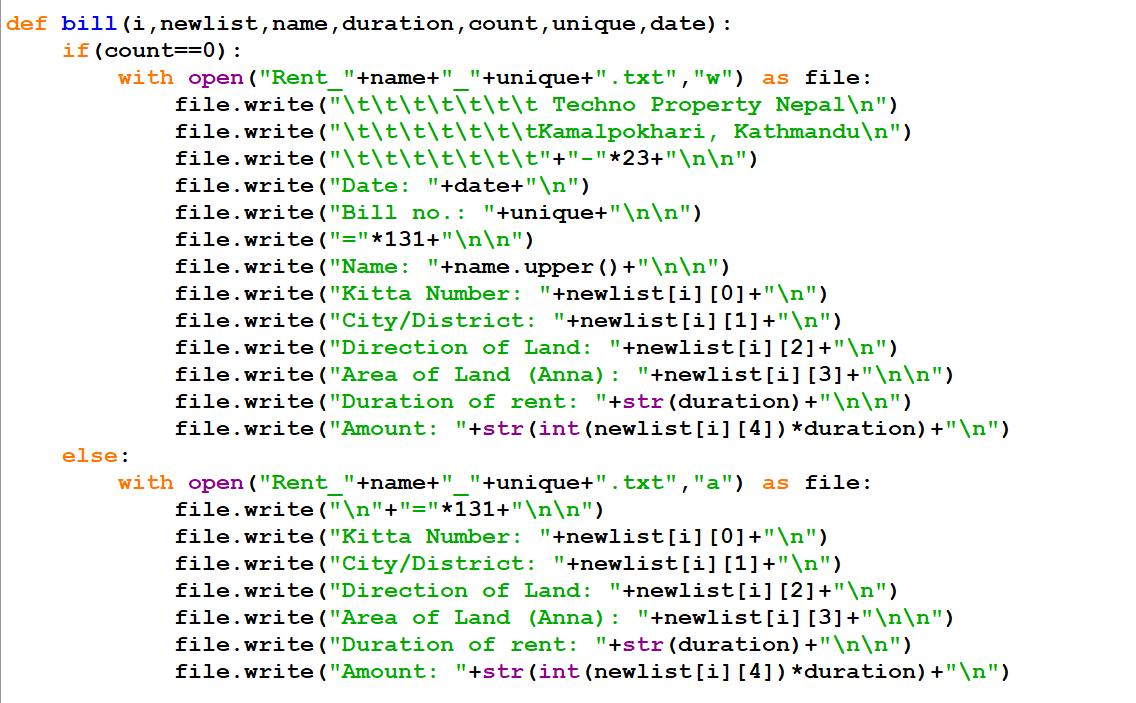


Figure 61: Screenshot of code that generate .txt file after renting multiple land

## Test 4: Invoice generation after returning multiple lands

|  |  |
| --- | --- |
| **Test no:** | 4 |
| **Objective:** | To test the generation of Invoice after returning multiple lands |
| **Action:** | * Provide all necessary information and return a land. * Enter 1 and return another land. * Choose 2 and complete the returning process. |
| **Expected Result:** | A combined bill for both lands will be printed and a .txt file will also be generated for it. |
| **Actual Result:** | A combined bill for both lands was printed and a .txt file was also generated for it. |
| **Conclusion:** | The test was successful. |

Table 5: Invoice generation after returning multiple lands test

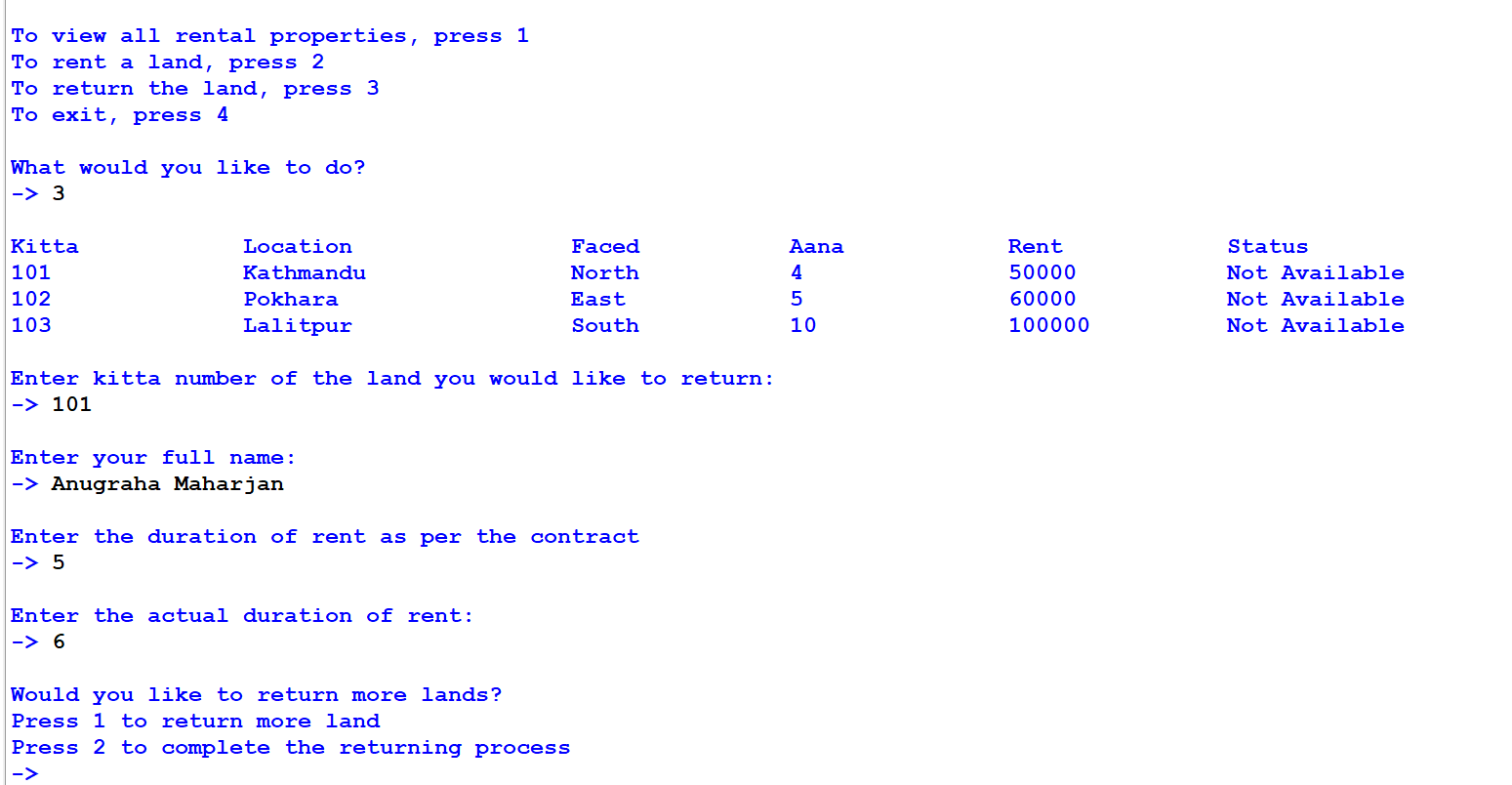


Figure 62: Screenshot of returning one land

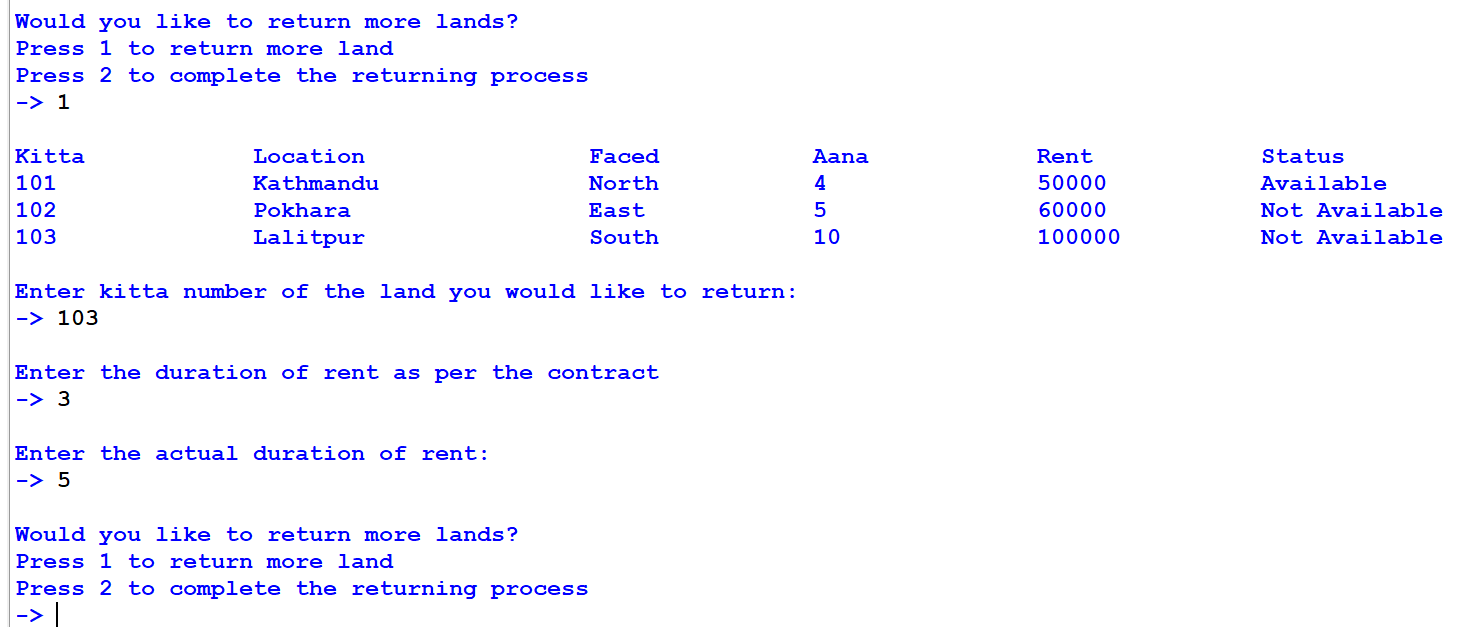


Figure 63: Screenshot of returning multiple land

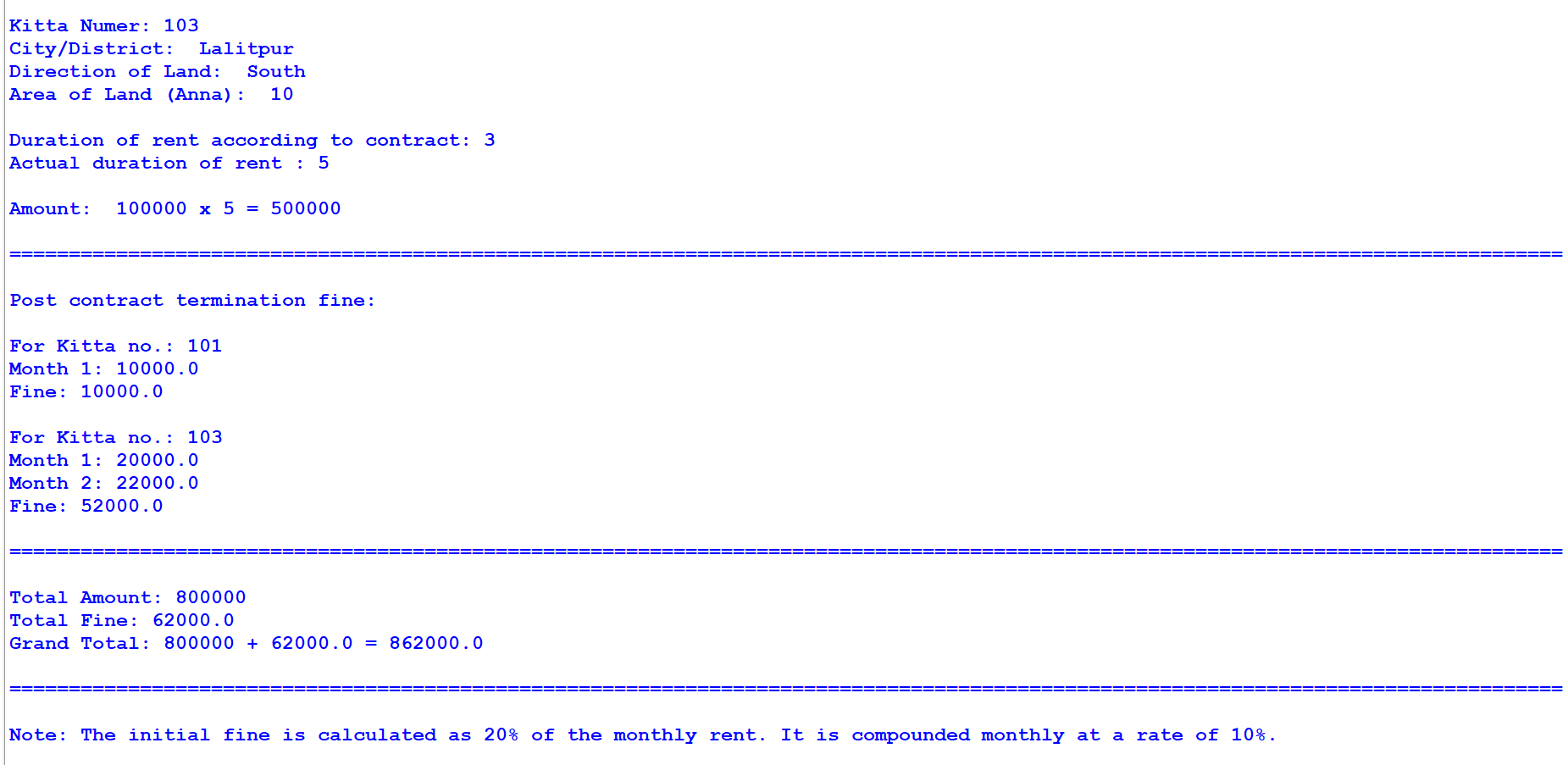
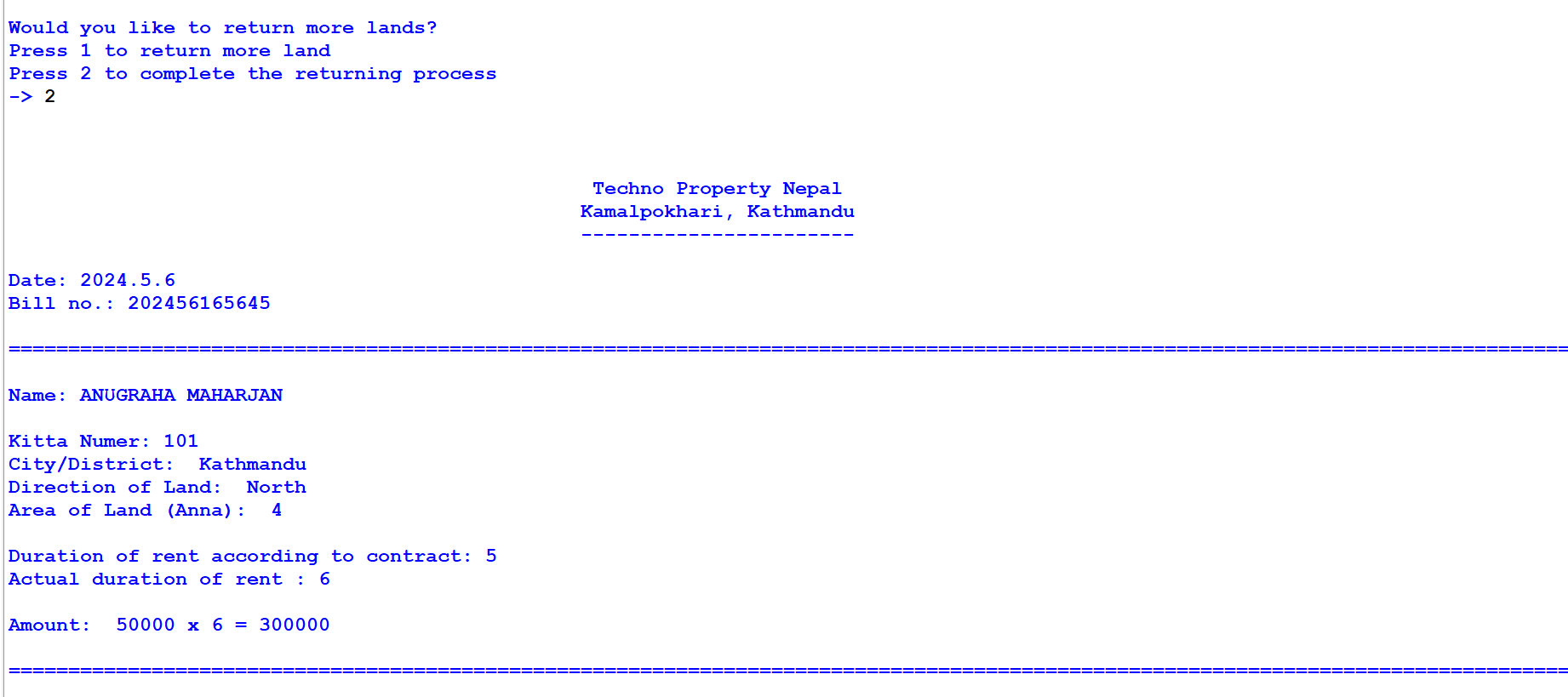


Figure 64: Screenshot of bill printed after returning multiple land



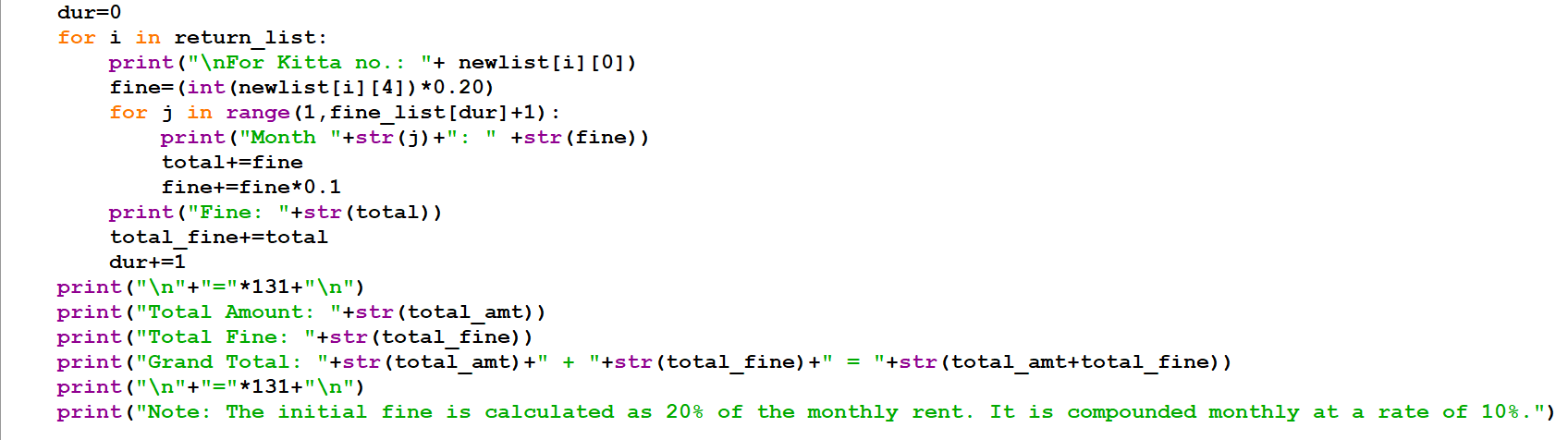


Figure 65: Screenshot of code that print bill after returning multiple land

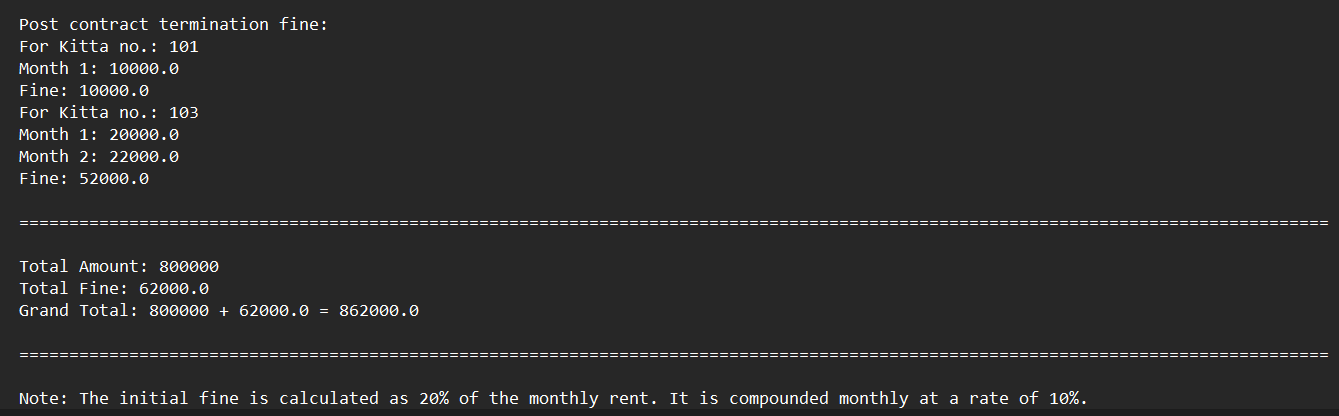
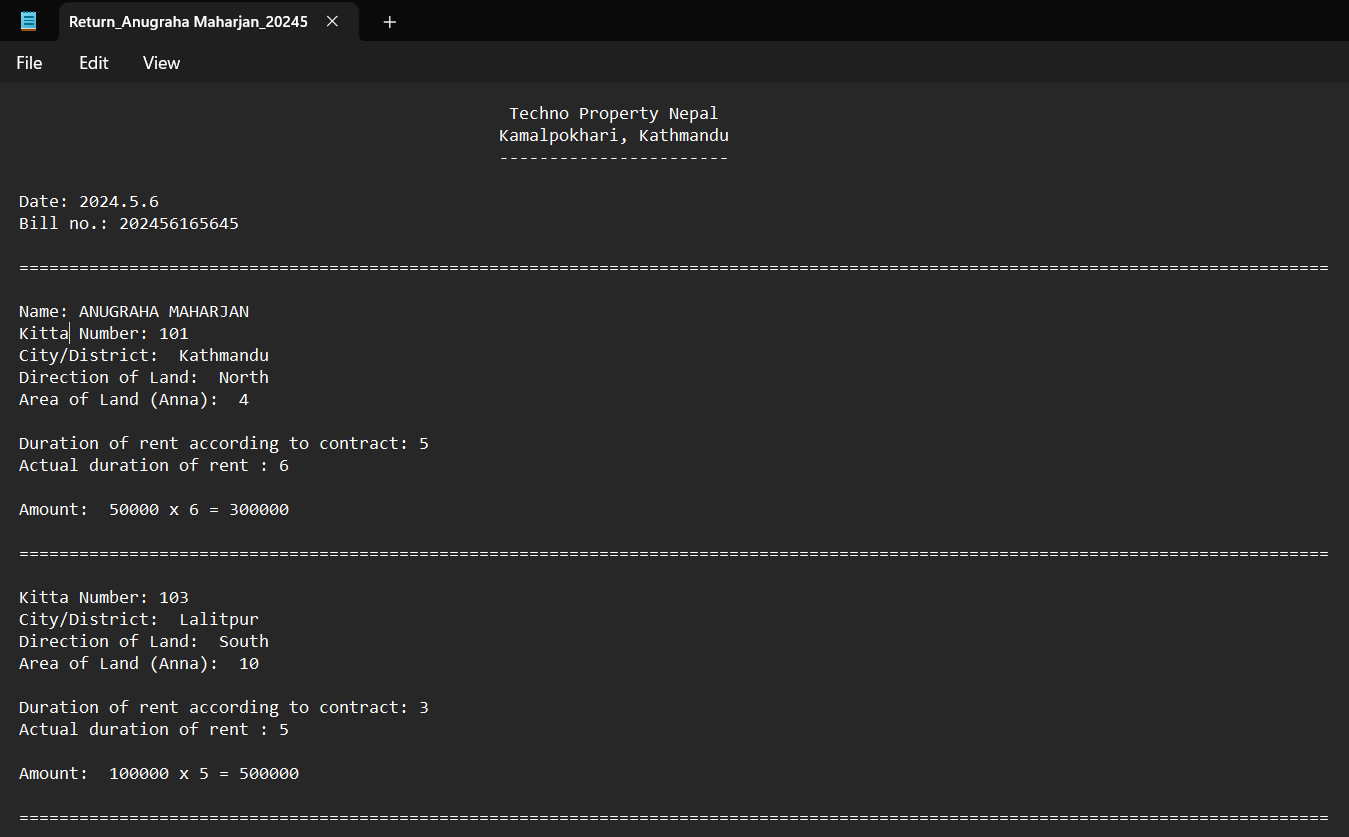
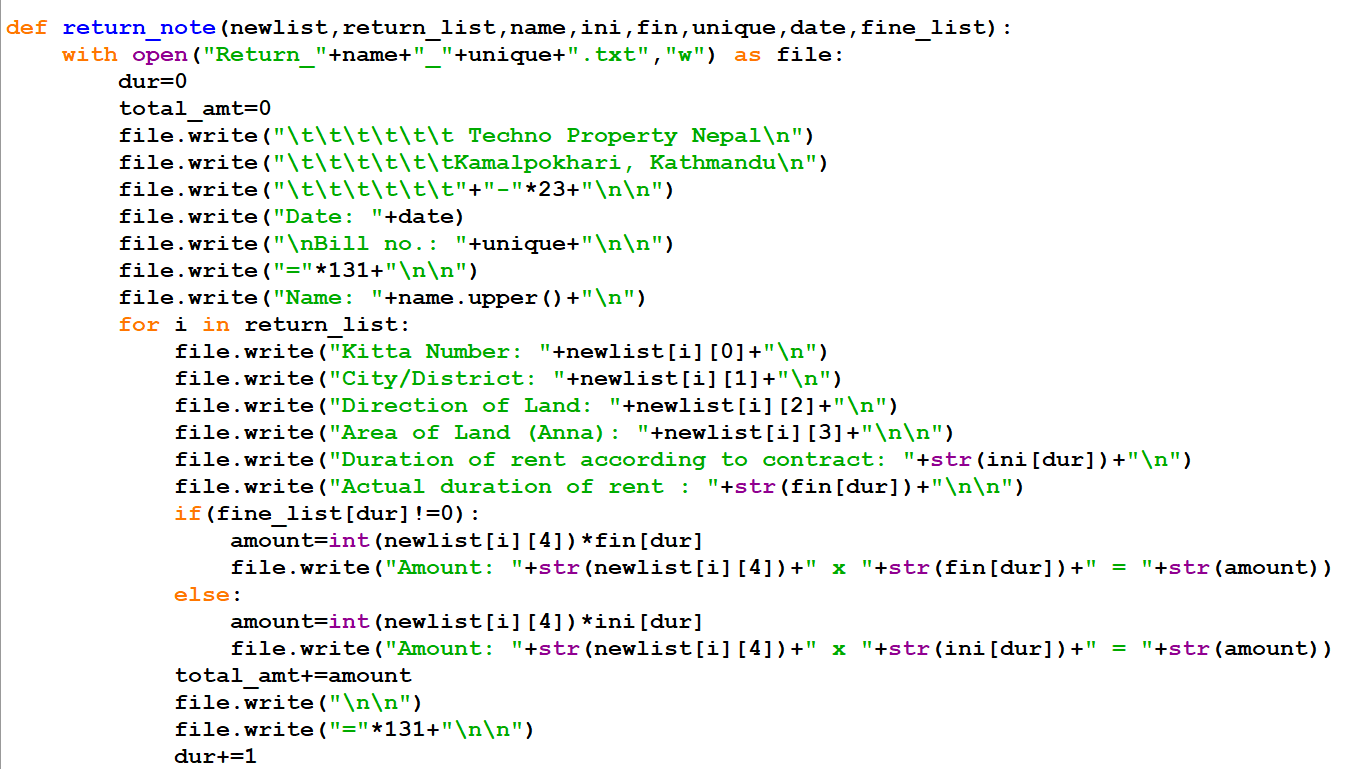


Figure 66: Screenshot of .txt file generated after returning multiple land



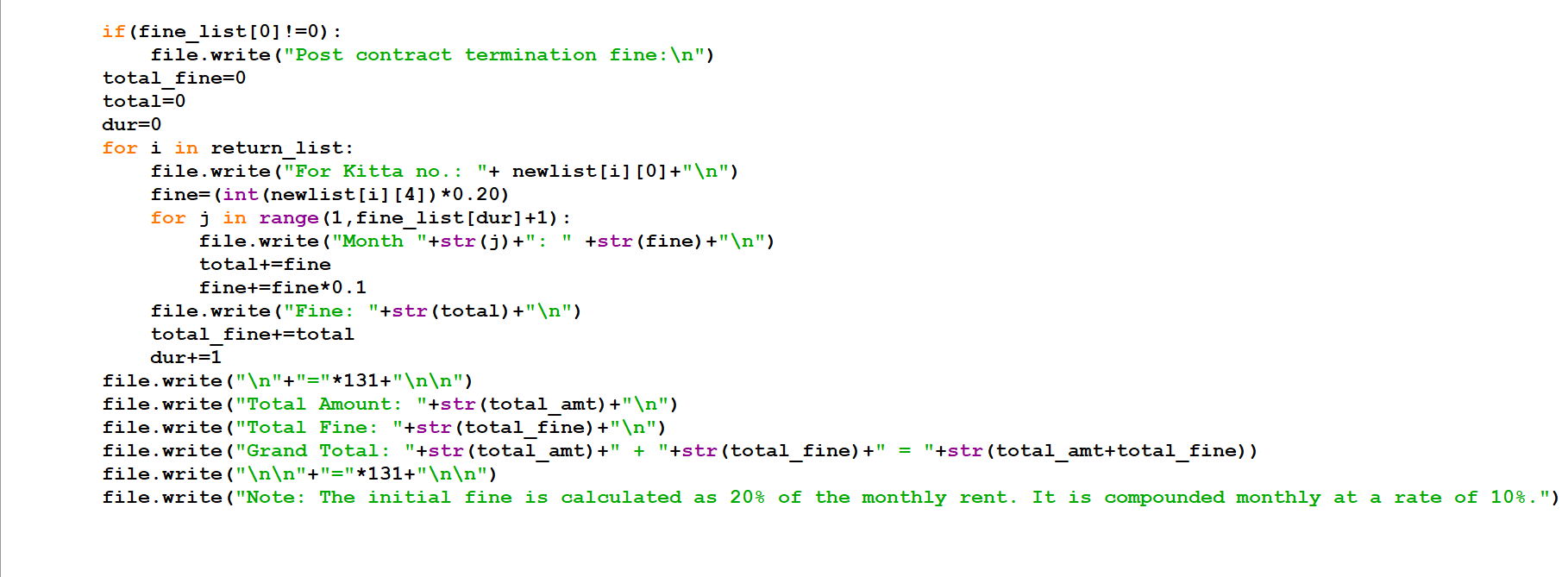


Figure 67: Screenshot of code that generate .txt file after returning multiple land

## Test 5: Update of Availability of Lands

### 4.5.1. Test 5.1: After renting land

|  |  |
| --- | --- |
| **Test no:** | 5.1 |
| **Objective:** | To test the update in availability of land. |
| **Action:** | * View the status before renting. * Complete all fields and rent a land. * Check the status of same land. |
| **Expected Result:** | The status will be changed to not available. |
| **Actual Result:** | The status was changed to not available. |
| **Conclusion:** | The test was successful. |

Table 6: Update status after renting land test

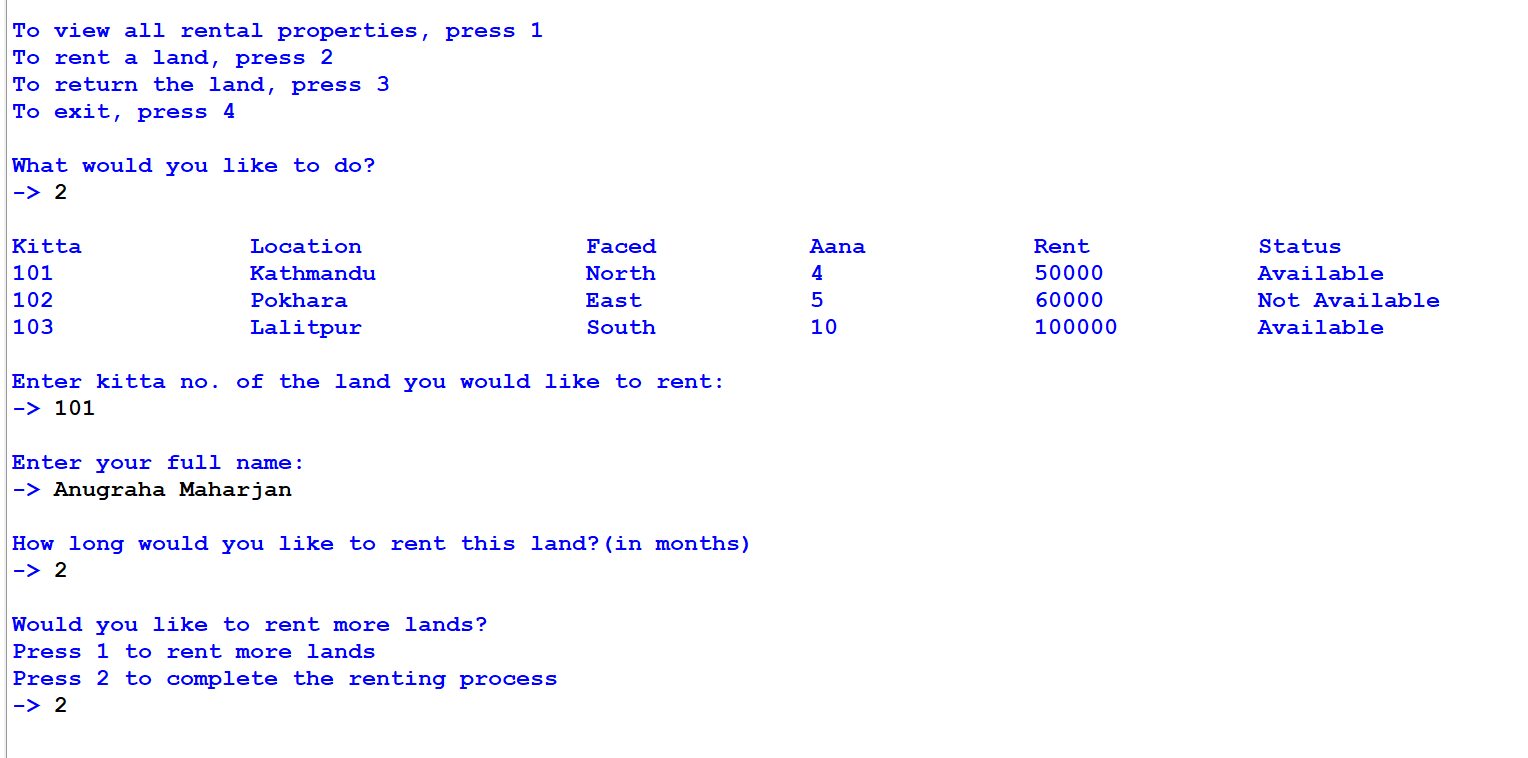


Figure 68: Screenshot of renting a land

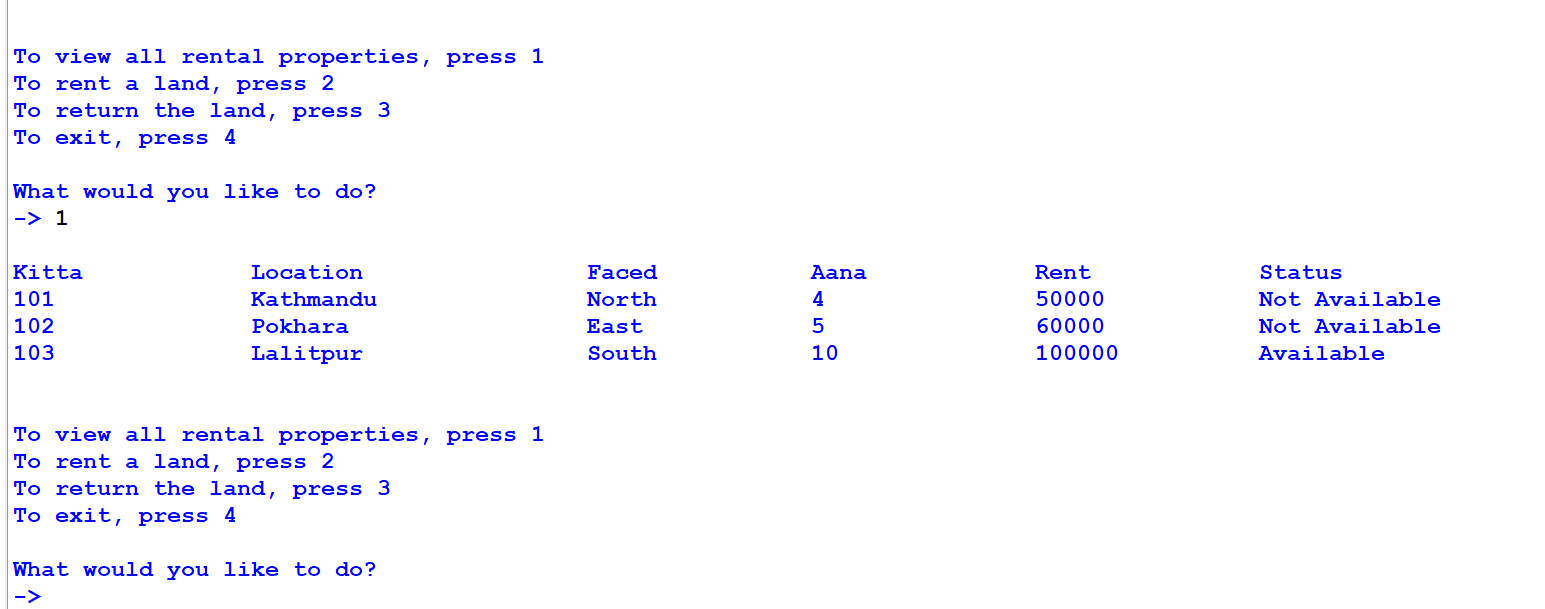


Figure 69: Screenshot of status changed after renting a land

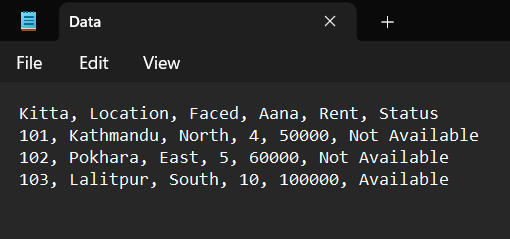


Figure 70: Screenshot of status changed after renting a land in .txt file

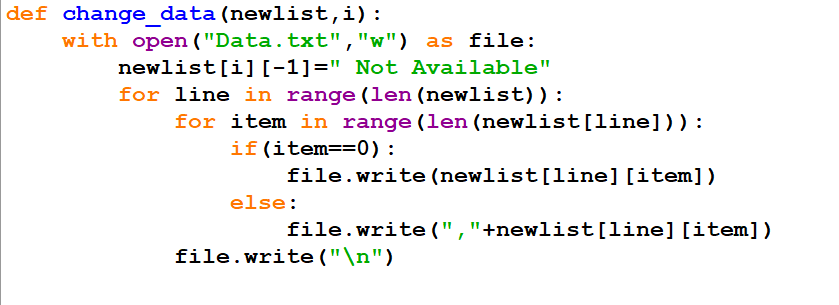


Figure 71: Screenshot of code that change status after renting a land

### 4.5.2. Test 5.2: After returning land

|  |  |
| --- | --- |
| **Test no:** | 5.2 |
| **Objective:** | To test the update in availability of land. |
| **Action:** | * View the status before returning. * Complete all fields and return a land. * Check the status of same land. |
| **Expected Result:** | The status will be changed to available. |
| **Actual Result:** | The status was changed to available. |
| **Conclusion:** | The test was successful. |

Table 7: Update status after returning land test

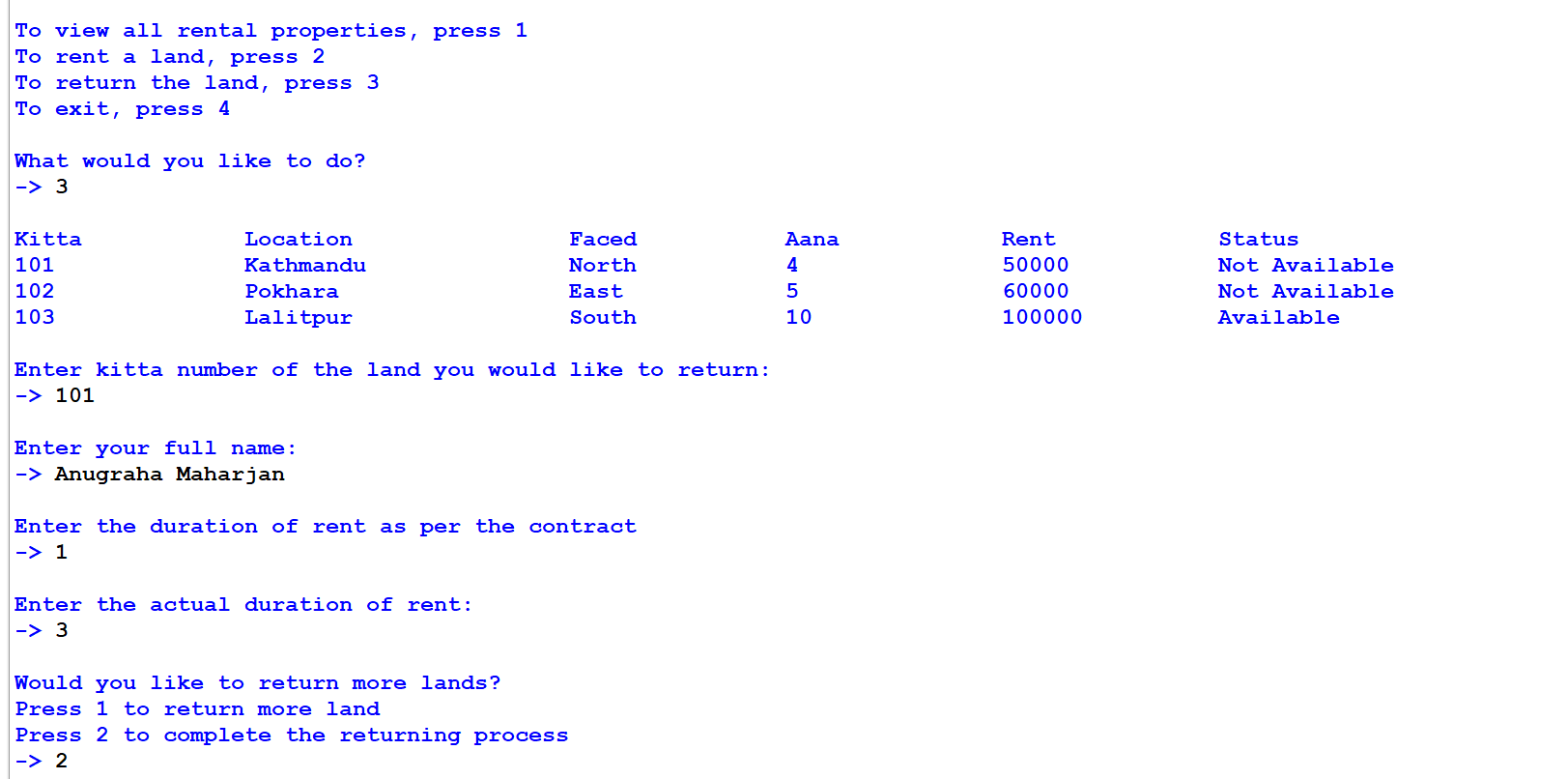


Figure 72: Screenshot of returning a land

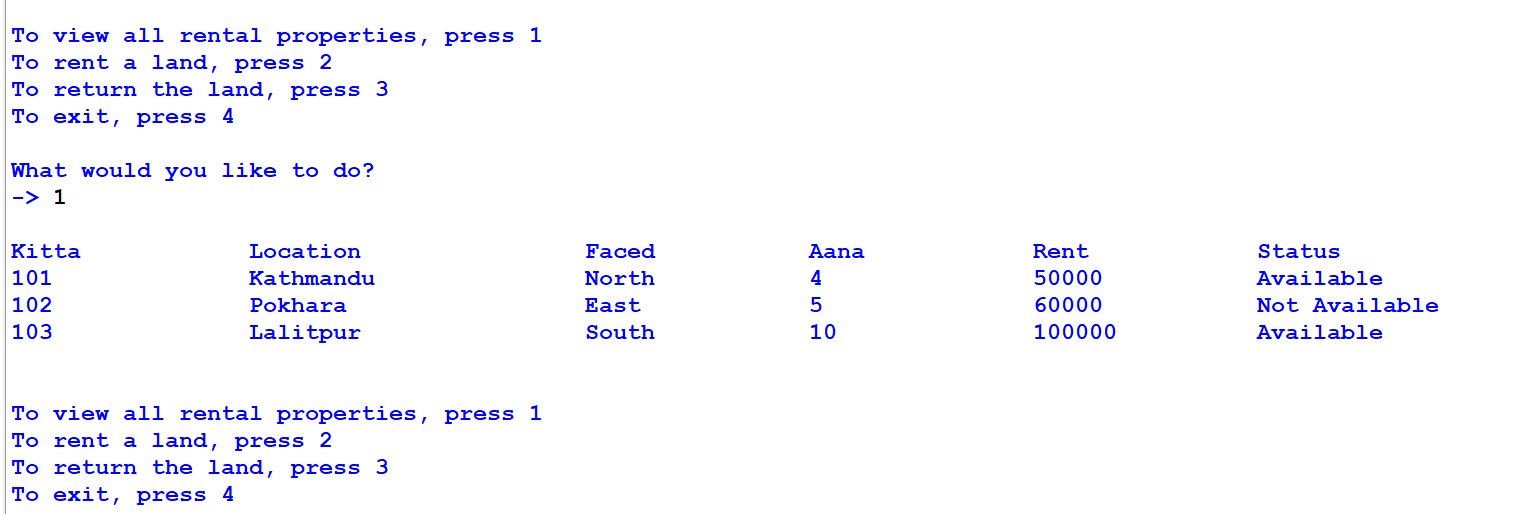


Figure 73: Screenshot of status changed after returning a land

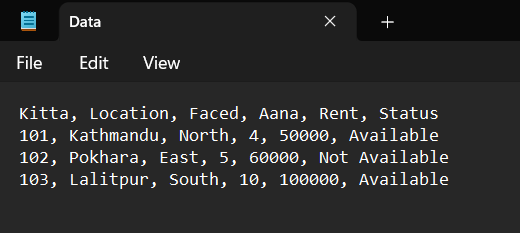


Figure 74: Screenshot of status changed after returning land in .txt file

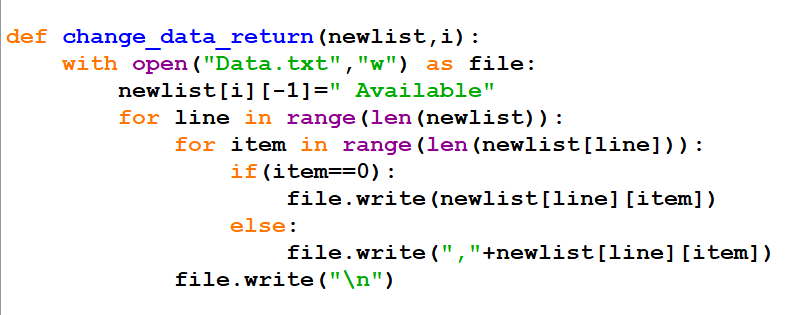


Figure 75: Screenshot of code to change status after returning land

# Conclusion

The development of Land Rental System is a step towards using technology to better our human life. The traditional method which was full of uncertainties will be replaced by more reliable system that can be used by anyone with a computer device. The system is also new user friendly, even a new user will start to have idea about the renting system unlike traditional method where one can get easily scammed.

This system contains four main functionalities, to view all rental properties, renting a property, returning a property and exiting from the system. Anyone capable of reading English and using computer will be able to easily able to use this system with very little knowledge about lands. Each functionality works so that both landlord and tenants have easier time working with each other. With each transaction being stored safely, in case of problem in truth is available.

During the development of this system, I have encountered some problems that I was able to solve with help of internet research and our module leaders. One of the main problems I faced was to separate bill creating function. Since modules cannot be imported in loop, I could not call write function directly from operations module. For this I have taken the name and duration input in main module and called the write function from main module where write module is imported. Due to the need of exception handling for these types of inputs, the main module has been a little longer than expected.

Problems like these have caused me to research more on these topics and have enabled me to learn more about python in depth. Creation of algorithm and flowchart has helped me understand the logic designing process of a system and the coding process helped me deepen my understanding of python. Thus, this project has helped me understand important parts of coding such as exception handling, functions, file handling, etc.

# Bibliography

Coursera. (2024, April 4). *What is Python used for*. Retrieved from Coursera: https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python

Educative. (2024, May 2). *Definition IDLE*. Retrieved from Educative: https://www.educative.io/answers/definition-idle

Geeks for Geeks. (2024, May 2). *What is Python*. Retrieved from Geeks for Geeks: https://www.geeksforgeeks.org/what-is-python/

Gupta, T. (2024, March 31). *Difference between primitive and non primitive data structure*. Retrieved from Scaler: https://www.scaler.com/topics/difference-between-primitive-and-non-primitive-data-structures/

Higssoftware. (2024, May 2). *What is Python- IDLE*. Retrieved from Higssoftware: http://higssoftware.com/what-is-python-idle.php

Mitchell, B. (2022, May 30). *Python data structure explained*. Retrieved from Codingdojo: https://www.codingdojo.com/blog/top-python-data-structures#:~:text=Types%20of%20Data%20Structures%20in%20Python&text=The%20four%20primitive%20data%20structures,of%20values%2C%20in%20varying%20formats.

Prepbytes. (2023, May 5). *Primitive data structure*. Retrieved from Prepbyres: https://www.prepbytes.com/blog/data-structure/primitive-data-structure/#:~:text=The%20int%20data%20type%20is,of%204%20bytes%20in%20memory.

Simplilearn. (2024, May 1). *What is Dictionary in python*. Retrieved from Simplilearn: https://www.simplilearn.com/dictionary-in-python-article#:~:text=In%20Python%2C%20dictionaries%20are%20mutable,update()%2C%20dict.

W3schools. (2024, 6 5). *Introduction to data structure*. Retrieved from W3schools: https://www.w3schools.com/dsa/dsa\_intro.php#:~:text=Common%20data%20structures%20include%20arrays%2C%20linked%20lists%2C%20and%20binary%20trees.&text=A%20measure%20of%20the%20amount,the%20algorithm%20is%20working%20on.&text=A%20measure%20of%20the%20amount

w3schools. (2024, May 2). *Python Introduction*. Retrieved from w3schools: https://www.w3schools.com/python/python\_intro.asp

# Appendix

## 7.1. main.py

from read import \*

from write import \*

from operations import \*

import datetime;

print("\n\n\n\t\t\t\t\t\t Techno Property Nepal")

print("\t\t\t\t\t\tKamalpokhari, Kathmandu")

print("\t\t\t\t\t\t"+"-"\*23)

unique\_bill=str(datetime.datetime.now().year)+str(datetime.datetime.now().month)+str(datetime.datetime.now().day)+str(datetime.datetime.now().hour)+str(datetime.datetime.now().minute)+str(datetime.datetime.now().second)

date\_bill=str(datetime.datetime.now().year)+"."+str(datetime.datetime.now().month)+"."+str(datetime.datetime.now().day)

loop=True

while(loop):

print("\n\nTo view all rental properties, press 1")

print("To rent a land, press 2")

print("To return the land, press 3")

print("To exit, press 4")

while(True):

try:

reply=int(input("\nWhat would you like to do?\n-> "))

break

except:

print("Invalid Format, Please provide proper values")

newlist=read()

if (reply==1):

viewAllLands(newlist)

elif (reply==2):

rent\_count=0

rented\_list=[]

rent\_more\_outer=0

duration\_list=[]

while(rent\_more\_outer==0):

viewAllLands(newlist)

valid\_kitta=True

while valid\_kitta:

try:

kitta=int(input("\nEnter kitta no. of the land you would like to rent:\n-> "))

valid\_kitta=False

except:

print("Invalid Format, Please provide proper values")

in\_loop=0 #to loop again from check\_kitta

while (in\_loop==0):

i=check\_kitta(newlist,kitta)

if(i==-1):

rent\_more\_outer=1

break

available=check\_availability(newlist,i)

if(available==-1):

rent\_more\_outer=1

break

elif(available==0):

if(rent\_count==0):

name=input("\nEnter your full name:\n-> ")

valid\_duration=True

while True:

while valid\_duration:

try:

duration=int(input("\nHow long would you like to rent this land?(in

months)\n-> "))

break

except:

print("Invalid Format, Please provide proper values")

if(duration<1):

print("Please enter valid duration for renting")

else:

duration\_list.append(duration)

break

rent\_more=0

while(rent\_more==0):

while True:

try:

rent\_more\_reply=int(input("\nWould you like to rent more

lands?\nPress 1 to rent more lands\nPress 2 to complete the renting

process\n-> "))

break

except:

print("Invalid Format, Please provide proper values")

if(rent\_more\_reply==1):

in\_loop=1

rented\_list.append(i)

bill(i,newlist,name,duration,rent\_count,unique\_bill,date\_bill)

rent\_count+=1

change\_data(newlist,i)

break

elif(rent\_more\_reply==2):

rented\_list.append(i)

bill(i,newlist,name,duration,rent\_count,unique\_bill,date\_bill)

change\_data(newlist,i)

bill\_total(newlist,name,rented\_list,unique\_bill,duration\_list)

print\_bill(newlist,name,duration\_list,rented\_list,unique\_bill,date\_bill)

print("\n\n\n\t\t\t\t\t\tWelcome to the Techno Property Nepal family.\n")

in\_loop=1

rent\_more\_outer=1

break

else:

print("Invalid input detected, Please provide valid number")

else:

kitta=available

elif(reply==3):

return\_list=[]

duration\_ini\_list=[]

duration\_fin\_list=[]

fine\_list=[]

return\_loop=True

return\_count=0

while(return\_loop):

viewAllLands(newlist)

while True:

try:

return\_land\_no=int(input("\nEnter kitta number of the land you would like to

return:\n-> "))

break

except:

print("Invalid format, Please provide proper values")

not\_available=0

while(not\_available==0):

i=check\_kitta(newlist,return\_land\_no)

if(i==-1):

return\_loop=False

break

unavailable=check\_unavailability(newlist,i)

if(unavailable==-1):

return\_loop=False

break

elif(unavailable==0):

if(return\_count==0):

name=input("\nEnter your full name:\n-> ")

while True:

while(True):

try:

duration\_ini=int(input("\nEnter the duration of rent as per the

contract\n-> "))

break

except:

print("Invalid Format, Please provide proper values")

if(duration\_ini<1):

print("Please enter valid duration for renting")

else:

break

while True:

while(True):

try:

duration\_final=int(input("\nEnter the actual duration of rent:\n-> "))

break

except:

print("Invalid Format, Please provide proper values")

if(duration\_final<1):

print("Please enter valid duration for renting")

else:

break

fine=check\_fine(duration\_ini,duration\_final)

fine\_list.append(fine)

while(True):

while(True):

try:

return\_more=int(input("\nWould you like to return more

lands?\nPress 1 to return more land\nPress 2 to complete the

returning process\n-> "))

break

except:

print("Invalid Format, Please provide proper values")

if(return\_more==1):

change\_data\_return(newlist,i)

return\_list.append(i)

duration\_ini\_list.append(duration\_ini)

duration\_fin\_list.append(duration\_final)

return\_count+=1

not\_available=1

break

elif(return\_more==2):

return\_list.append(i)

duration\_ini\_list.append(duration\_ini)

duration\_fin\_list.append(duration\_final)

return\_note(newlist, return\_list, name, duration\_ini\_list,

duration\_fin\_list, unique\_bill,date\_bill,fine\_list)

change\_data\_return(newlist,i)

print\_ret\_bill(newlist,return\_list,name,duration\_ini\_list,

duration\_fin\_list,unique\_bill,date\_bill,fine\_list)

print("\n\n\n\t\t\t\t\tLand returned sucessfully")

return\_loop=False

not\_available=1

break

else:

print("Invalid number, Please provide valid number")

else:

return\_land\_no=unavailable

elif(reply==4):

print("\n\n\n\t\t\t\t\t\tThank you for choosing us\n\n")

loop=False

else:

print("Invalid number, Please provide valid value.")

## 7.2. read.py

def read():

newlist=[]

with open("Data.txt","r") as file:

for line in file:

line=line.replace("\n","")

newlist.append(line.split(','))

return newlist

## 7.3. operations.py

from read import \*

def viewAllLands(newlist):

print()

for line in newlist:

for item in line:

print(item,end="\t\t")

print()

def check\_kitta(newlist,kitta):

for i in range(len(newlist)):

if(newlist[i][0]==str(kitta)):

return i

reply=input("\nPlease enter valid kitta no. or Type exit to return to main menu:\n-> ")

if (reply.lower()=="exit"):

return -1

else:

return check\_kitta(newlist,reply)

def check\_availability(newlist,i):

check\_avai\_list=newlist

check\_avai\_list[i][-1]=check\_avai\_list[i][-1].replace(" ","")

if(check\_avai\_list[i][-1].lower()=="available"):

return 0

else:

print("\nThe selected land is not available to rent.")

reply=input("Please select another kitta no. OR Type exit to return to main menu:\n-> ")

if (reply.lower()=="exit"):

return -1

else:

return reply

def print\_bill(newlist,name,duration,rented,unique,date):

dur=0

print("\n\n\n\t\t\t\t\t\t Techno Property Nepal")

print("\t\t\t\t\t\tKamalpokhari, Kathmandu")

print("\t\t\t\t\t\t"+"-"\*23+"\n")

print("Date: "+date)

print("Bill no.: "+unique)

print("\n"+"="\*131+"\n")

print("Name: "+name.upper())

for i in rented:

print("\nKitta Number: "+newlist[i][0])

print("City/District: "+newlist[i][1])

print("Direction of Land: "+newlist[i][2])

print("Area of Land (Anna): "+newlist[i][3])

print("\nDuration of rent: ",duration[dur])

print("\nAmount: ",int(newlist[i][4])\*duration[dur])

print("\n"+"="\*131)

dur+=1

total=0

print("\nTotal Amount:",end=" ")

count=0

dur=0

for i in rented:

if(count==0):

print(int(newlist[i][4])\*duration[dur],end=" ")

else:

print("+",int(newlist[i][4])\*duration[dur],end=" ")

count+=1

dur+=1

total+=int(newlist[i][4])

if(count>1):

print("=",total)

def check\_unavailability(newlist,i):

check\_unavai\_list=newlist

check\_unavai\_list[i][-1]=check\_unavai\_list[i][-1].replace(" ","")

if(check\_unavai\_list[i][-1].lower()=="notavailable"):

return 0

else:

print("\nThe selected land has not been rented")

reply=input("Please select another kitta no. OR Type exit to return to main menu:\n-> ")

if (reply.lower()=="exit"):

return -1

else:

return reply

def print\_ret\_bill(newlist,return\_list,name,ini,fin,unique,date,fine\_list):

dur=0

total\_amt=0

print("\n\n\n\t\t\t\t\t\t Techno Property Nepal")

print("\t\t\t\t\t\tKamalpokhari, Kathmandu")

print("\t\t\t\t\t\t"+"-"\*23+"\n")

print("Date: "+date)

print("Bill no.: "+unique+"\n")

print("="\*131+"\n")

print("Name: "+name.upper()+"\n")

for i in return\_list:

print("Kitta Numer: "+newlist[i][0])

print("City/District: "+newlist[i][1])

print("Direction of Land: "+newlist[i][2])

print("Area of Land (Anna): "+newlist[i][3]+"\n")

print("Duration of rent according to contract: "+str(ini[dur]))

print("Actual duration of rent : "+str(fin[dur])+"\n")

if(fine\_list[dur]!=0):

amount=int(newlist[i][4])\*fin[dur]

print("Amount: "+str(newlist[i][4])+" x "+str(fin[dur])+" = "+str(amount))

else:

amount=int(newlist[i][4])\*ini[dur]

print("Amount: "+str(newlist[i][4])+" x "+str(ini[dur])+" = "+str(amount))

total\_amt+=amount

print()

print("="\*131+"\n")

dur+=1

if(fine\_list[0]!=0):

print("Post contract termination fine:")

total\_fine=0

total=0

dur=0

for i in return\_list:

print("\nFor Kitta no.: "+ newlist[i][0])

fine=(int(newlist[i][4])\*0.20)

for j in range(1,fine\_list[dur]+1):

print("Month "+str(j)+": " +str(fine))

total+=fine

fine+=fine\*0.1

print("Fine: "+str(total))

total\_fine+=total

dur+=1

print("\n"+"="\*131+"\n")

print("Total Amount: "+str(total\_amt))

print("Total Fine: "+str(total\_fine))

print("Grand Total: "+str(total\_amt)+" + "+str(total\_fine)+" = "+str(total\_amt+total\_fine))

print("\n"+"="\*131+"\n")

print("Note: The initial fine is calculated as 20% of the monthly rent. It is compounded monthly at a rate of 10%.")

def check\_fine(ini,fin):

if(ini<fin):

return fin-ini

else:

return 0

## 7.4. write.py

from operations import \*;

def change\_data(newlist,i):

with open("Data.txt","w") as file:

newlist[i][-1]=" Not Available"

for line in range(len(newlist)):

for item in range(len(newlist[line])):

if(item==0):

file.write(newlist[line][item])

else:

file.write(","+newlist[line][item])

file.write("\n")

def bill(i,newlist,name,duration,count,unique,date):

if(count==0):

with open("Rent\_"+name+"\_"+unique+".txt","w") as file:

file.write("\t\t\t\t\t\t\t Techno Property Nepal\n")

file.write("\t\t\t\t\t\t\tKamalpokhari, Kathmandu\n")

file.write("\t\t\t\t\t\t\t"+"-"\*23+"\n\n")

file.write("Date: "+date+"\n")

file.write("Bill no.: "+unique+"\n\n")

file.write("="\*131+"\n\n")

file.write("Name: "+name.upper()+"\n\n")

file.write("Kitta Number: "+newlist[i][0]+"\n")

file.write("City/District: "+newlist[i][1]+"\n")

file.write("Direction of Land: "+newlist[i][2]+"\n")

file.write("Area of Land (Anna): "+newlist[i][3]+"\n\n")

file.write("Duration of rent: "+str(duration)+"\n\n")

file.write("Amount: "+str(int(newlist[i][4])\*duration)+"\n")

else:

with open("Rent\_"+name+"\_"+unique+".txt","a") as file:

file.write("\n"+"="\*131+"\n\n")

file.write("Kitta Number: "+newlist[i][0]+"\n")

file.write("City/District: "+newlist[i][1]+"\n")

file.write("Direction of Land: "+newlist[i][2]+"\n")

file.write("Area of Land (Anna): "+newlist[i][3]+"\n\n")

file.write("Duration of rent: "+str(duration)+"\n\n")

file.write("Amount: "+str(int(newlist[i][4])\*duration)+"\n")

def bill\_total(newlist,name,rented,unique,duration):

with open("Rent\_"+name+"\_"+unique+".txt","a") as file:

total=0

file.write("\n"+"="\*131+"\n\n")

file.write("Total Amount:"+" ")

count=0

for i in rented:

if(count==0):

file.write(str(int(newlist[i][4])\*duration[count])+" ")

else:

file.write("+"+str(int(newlist[i][4])\*duration[count])+" ")

count+=1

total+=int(newlist[i][4])

if(count>1):

file.write("= "+str(total))

def change\_data\_return(newlist,i):

with open("Data.txt","w") as file:

newlist[i][-1]=" Available"

for line in range(len(newlist)):

for item in range(len(newlist[line])):

if(item==0):

file.write(newlist[line][item])

else:

file.write(","+newlist[line][item])

file.write("\n")

def return\_note(newlist,return\_list,name,ini,fin,unique,date,fine\_list):

with open("Return\_"+name+"\_"+unique+".txt","w") as file:

dur=0

total\_amt=0

file.write("\t\t\t\t\t\t Techno Property Nepal\n")

file.write("\t\t\t\t\t\tKamalpokhari, Kathmandu\n")

file.write("\t\t\t\t\t\t"+"-"\*23+"\n\n")

file.write("Date: "+date)

file.write("\nBill no.: "+unique+"\n\n")

file.write("="\*131+"\n\n")

file.write("Name: "+name.upper()+"\n")

for i in return\_list:

file.write("Kitta Number: "+newlist[i][0]+"\n")

file.write("City/District: "+newlist[i][1]+"\n")

file.write("Direction of Land: "+newlist[i][2]+"\n")

file.write("Area of Land (Anna): "+newlist[i][3]+"\n\n")

file.write("Duration of rent according to contract: "+str(ini[dur])+"\n")

file.write("Actual duration of rent : "+str(fin[dur])+"\n\n")

if(fine\_list[dur]!=0):

amount=int(newlist[i][4])\*fin[dur]

file.write("Amount: "+str(newlist[i][4])+" x "+str(fin[dur])+" = "+str(amount))

else:

amount=int(newlist[i][4])\*ini[dur]

file.write("Amount: "+str(newlist[i][4])+" x "+str(ini[dur])+" = "+str(amount))

total\_amt+=amount

file.write("\n\n")

file.write("="\*131+"\n\n")

dur+=1

if(fine\_list[0]!=0):

file.write("Post contract termination fine:\n")

total\_fine=0

total=0

dur=0

for i in return\_list:

file.write("For Kitta no.: "+ newlist[i][0]+"\n")

fine=(int(newlist[i][4])\*0.20)

for j in range(1,fine\_list[dur]+1):

file.write("Month "+str(j)+": " +str(fine)+"\n")

total+=fine

fine+=fine\*0.1

file.write("Fine: "+str(total)+"\n")

total\_fine+=total

dur+=1

file.write("\n"+"="\*131+"\n\n")

file.write("Total Amount: "+str(total\_amt)+"\n")

file.write("Total Fine: "+str(total\_fine)+"\n")

file.write("Grand Total: "+str(total\_amt)+" + "+str(total\_fine)+" = "+str(total\_amt+total\_fine))

file.write("\n\n"+"="\*131+"\n\n")

file.write("Note: The initial fine is calculated as 20% of the monthly rent. It is compounded monthly at a rate of 10%.")