TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

LAB 4

By:

Sinjal Dahal (081/BEL/080)

DEPARTMENT OF COMPUTER ENGINEERING

LALITPUR, NEPAL

Project 1

Create a CLI (Command Line Interface) contact book that allows users to:

- *Add a new contact (append to file)*
- *View all contacts (read from file)*
- *Search for a contact (read and filter)*
- *Handle file-related exceptions (e.g., file not found)*

File Used:

contacts.txt (stores contact info: Name, Phone)

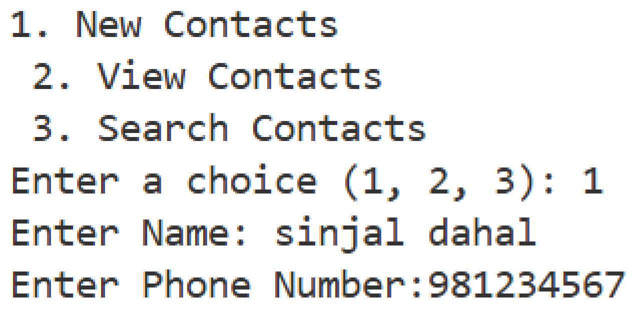
Code:

```
import os
directory = r"C:\Users\daSynZyoll SIR\Desktop\sinjaldahal_BEL-main\sinjaldahal_BEL-main\LabWork\lab_4\project_1"


filename = "contacts.txt"


file_path = os.path.join(directory, filename)


contacts = []
print("1. New Contacts \n 2. View Contacts \n 3. Search Contacts")
choice = input("Enter a choice (1, 2, 3): ")
```

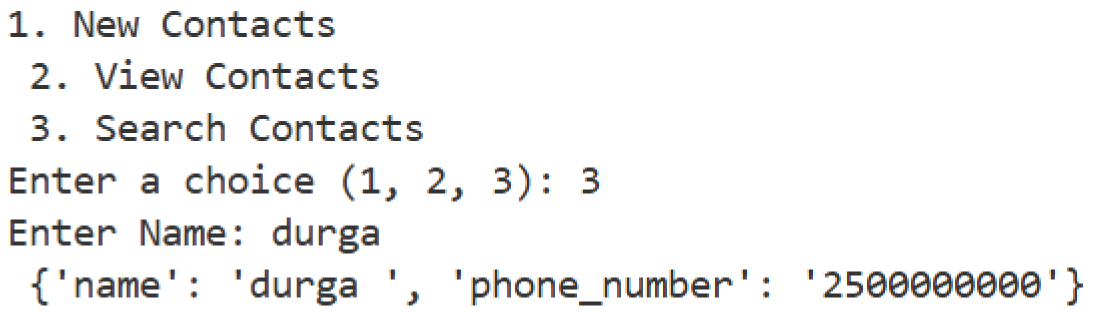
```
def new_contacts(contacts,name,number):
    name = input("Enter Name: ")
    number = input("Enter Phone Number:")
    contacts_add = {"name" : name , "phone_number" : number }

    with open(file_path,"a") as file:
        file.write(str(contacts_add) + "\n ")

    file.close()

def view_contacts():
    with open(file_path, "r") as file:
        for line in file:
            print(line.strip())
    file.close()

def search_contacts():
    name_input = input("Enter Name: ")
    with open(file_path,"r") as file:
        for line in file:
            if(name_input in line):
                print(line, end="")
                found = True
        if not found:
            print("Name not found !!!")
```

```
match choice:
    case "1":
        new_contacts(contacts,name="person",number="0000000000")
    case "2":
        view_contacts()
    case "3":
        search_contacts()
    case _:
        print("Invalid choice.")
```

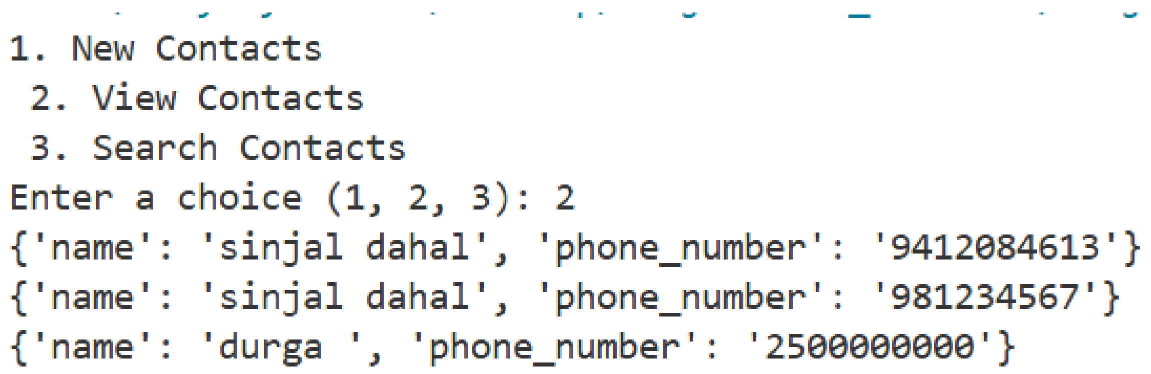
Output:



```
1. New Contacts
 2. View Contacts
 3. Search Contacts
Enter a choice (1, 2, 3): 1
Enter Name: sinjal dahal
Enter Phone Number:981234567
```



```
1. New Contacts
 2. View Contacts
 3. Search Contacts
Enter a choice (1, 2, 3): 3
Enter Name: durga
 {'name': 'durga ', 'phone_number': '2500000000'}
```



```
1. New Contacts
 2. View Contacts
 3. Search Contacts
Enter a choice (1, 2, 3): 2
{'name': 'sinjal dahal', 'phone_number': '9412084613'}
{'name': 'sinjal dahal', 'phone_number': '981234567'}
{'name': 'durga ', 'phone_number': '2500000000'}
```

Project 2

Create a simple banking system that:

- *Stores customer info in a file*
- *Allows deposits and withdrawals using functions*
- *Updates customer balance*
- *Logs all transactions in a separate file*
- *Handles exceptions gracefully*

Files Used:

customers.txt — stores customer records in the format:

Name, Account Number, Balance

transactions.txt — appends every deposit or withdrawal record with timestamp

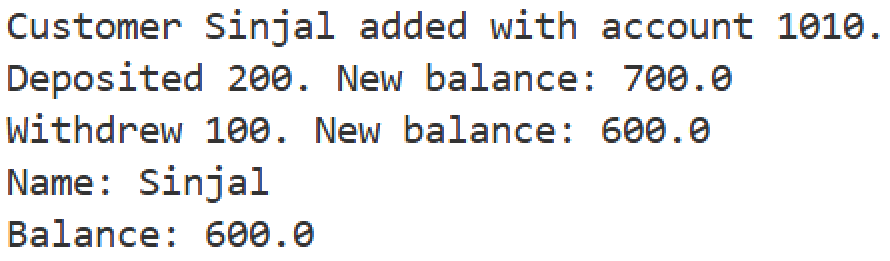
Code:

```
import os

from datetime import datetime

folder = r"C:\Users\daSynZyoll SIR\Desktop\sinjaldahal_BEL-main\sinjaldahal_BEL-
main\LabWork\lab_4\project_2"

os.makedirs(folder, exist_ok=True)


customer_file = os.path.join(folder, "customer_info.txt")

transaction_file = os.path.join(folder, "transactions.txt")


def load_customers():

    customers = {}

    if not os.path.exists(customer_file):

        return customers

    with open(customer_file, "r") as f:

        for line in f:

            line = line.strip()

            if line:

                parts = line.split(",")

                if len(parts) == 3:

                    name, acc_num, balance = parts

                    customers[acc_num] = {"name": name, "balance": float(balance)}

    return customers


def save_customers(customers):

    with open(customer_file, "w") as f:

        for acc_num, info in customers.items():
            f.write(f"{info['name']},{acc_num},{info['balance']}\n")

def log_transaction(acc_num, type_, amount):
    time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    with open(transaction_file, "a") as f:
        f.write(f"{time},{acc_num},{type_},{amount}\n")

def add_customer(customers, name, acc_num, balance=0):
    if acc_num in customers:
        print("This account number already exists.")
        return
    customers[acc_num] = {"name": name, "balance": float(balance)}
    save_customers(customers)
    print(f"Customer {name} added with account {acc_num}.")

def deposit(customers, acc_num, amount):
    if amount <= 0:
        print("Enter a positive amount.")
        return
    if acc_num not in customers:
        print("Account not found.")
        return
    customers[acc_num]["balance"] += amount
    save_customers(customers)
    log_transaction(acc_num, "DEPOSIT", amount)
    print(f"Deposited {amount}. New balance: {customers[acc_num]['balance']}")
```

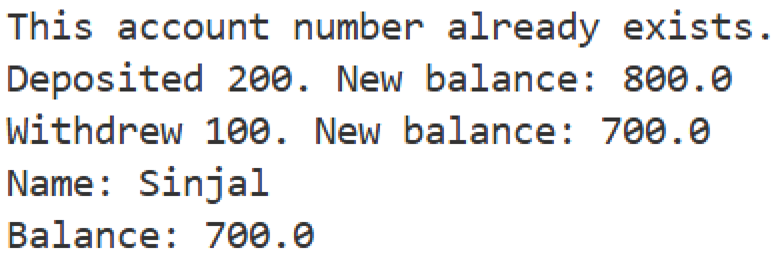
```
def withdraw(customers, acc_num, amount):
    if amount <= 0:
        print("Enter a positive amount.")
        return
    if acc_num not in customers:
        print("Account not found.")
        return
    if customers[acc_num]["balance"] < amount:
        print("Not enough balance.")
        return
    customers[acc_num]["balance"] -= amount
    save_customers(customers)
    log_transaction(acc_num, "WITHDRAW", amount)
    print(f"Withdrew {amount}. New balance: {customers[acc_num]['balance']}")


def show_balance(customers, acc_num):
    if acc_num in customers:
        print(f"Name: {customers[acc_num]['name']}")
        print(f"Balance: {customers[acc_num]['balance']}")
    else:
        print("Account not found.")


customers = load_customers()
add_customer(customers, "Sinjal", "1010", 500)
deposit(customers, "1010", 200)
```

```
withdraw(customers, "1010", 100)

show_balance(customers, "1010")
```

Output:



```
Customer Sinjal added with account 1010.
Deposited 200. New balance: 700.0
Withdrew 100. New balance: 600.0
Name: Sinjal
Balance: 600.0
```



```
This account number already exists.
Deposited 200. New balance: 800.0
Withdrew 100. New balance: 700.0
Name: Sinjal
Balance: 700.0
```