Group 18 Documentation

Phonebook application

Section A: Algorithms representation of the different modules/functions(pseudocode/flowcharts)

	Name	Surname	Student nr	Class Group
1	Justin Jerome	Campbell	224017435	Group 16
2	Anel	De Waal	224076221	Group 5
3	Kristine .C.N	Makili	224039350	Group 1
4	Isaiah	Mjumira	224099078	Group 12
5	Sheldon Tsysmurwa	Bhunu Shava	224094491	Group 4
6	Lenchiwasijey	Mudzengerere	224091034	Group 2

Section A .1 Design the algorithm

Application Modules

A module in our context refers to a self-contained component of a phonebook application that performs a specific function related to contact management .Each module handles a particular task , allowing for organized and efficient management of contacts.Moreover each modules contributes to the overall functionality of the phonebook, enabling users to manage their contacts efficiently.

In our phonebook modules are:

1. Contact Module : Insert, Update, Delete, and Search Contacts.
2. Display Module : Display all contacts.
3. Sorting Module : For faster searches.
4. Efficiency Analysis Module: To analyze the performance of the search algorithm.
5. Saving contacts : Save contacts to file
6. Loading contacts : Loading contacts in a fie

1*. LOGIC REPRESENTATION OF CONTACT MODULE*

1.1 INSERT CONTACTS -Adds a new contact to the phonebook, ensuring that no duplicate entries are created and that the phonebook has space for new contacts.

1.2 SEARCH CONTACTS -Allows users to find a contact by their name and retrieve their phone number.

1.3 DELETE CONTACTS -Removes a contact from the phonebook based on their name.

1.5 UPDATE CONTACTS -Modifies the phone number of an existing contact.

2. LOGIC REPRESENTATION OF DISPLAY MODULE

2.1 DISPLAY ALL CONTACTS -Shows a list of all contacts currently in the phonebook.

3. LOGIC REPRESENTATION OF SORTING MODULE

3.1 SORT CONTACTS -Arranges the contacts in alphabetical order by name to make searching faster

4. LOGIC REPRESENTATION OF ANALYSING MODULE

4.1 ANALYZE EFFICIENCY -Measures how long it takes to search for a contact, helping to understand the performance of the search operation.

5. *LOGIC REPRESENTATION OF SAVING CONTACTS*

5.1 Save Contacts to File - Saves all contacts to a file for persistence.

6. *LOGIC REPRESENTATION OF LOADING CONTACTS*

6.2 Load Contacts from File - Loads contacts from a file into the phonebook.

System Components:

1.1 INSERT CONTACTS

INPUT	PROCESS	OUTPUT
• phonebook (name and phone_number) • new_contact (name and phone_number) • size (maximum number of contacts the phonebook can hold)	Check if the Phonebook is Full Check for Duplicate Contacts Find an Empty Slot and Insert the New Contact	"Contact added successfully" or "Phonebook is full" or "Contact already exists").

1.2 SEARCH CONTACT

INPUT	PROCESS	OUTPUT
• phonebook (Array of Contact objects, where each Contact has name and phone_number) • search_name (STRING)	• Loop through each contact in the phonebook. • If a contact with the name matching search_name is found, return the phone number. • If the loop finishes without finding a match, return "Contact not found."	The phone number of the contact if found, or a "Contact not found" message.

1.3 DELETE CONTACTS

INPUT	PROCESS	OUTPUT
• phonebook (Array of Contact objects) • contact_name (String)	Check if the phonebook is empty. • If it is not empty, loop through each contact and print their details. • If it is empty, print "No contacts available.	. • A message indicating whether the contact was successfully deleted or not found.

1.4 UPDATE CONTACTS

INPUT	PROCESS	OUTPUT
• phonebook (Array of Contact objects) • contact_name (String) • new_phone_number (String	) • Loop through each contact to find a match by contact_name. • If a match is found, update their phone number and indicate success. • If the loop finishes without finding the contact, return "Contact not found	." A message indicating whether the contact was successfully updated or not found.

2.1 DISPLAY CONTACT

INPUT	PROCESS	OUTPUT
phonebook (Array of Contact objects	) 1. Check if the phonebook is empty. 2. If it is not empty, loop through each contact and print their details. 3. If it is empty, print "No contacts available	." A list of contact details or a "No contacts available" message..

3.1 SORT CONTACTS

INPUT	PROCESS	OUTPUT
phonebook (Array of Contact objects)	• Use a simple sorting algorithm (e.g., Bubble Sort) to sort the contacts by name. • Print "Contacts sorted successfully."	The sorted p V honebook.

4.1 ANALYSE CONTACTS

INPUT	PROCESS	OUTPUT
• phonebook (Array of Contact objects) • search_name (String)	)• Start a timer before performing the search. • Call the SearchContact function. • Stop the timer and calculate the elapsed time. • Print the time taken for the search.	The time taken to perform the search