Contact Algorithm

1 .Define the class `Contact:`

a. Define the `__init__` method that takes `name`, `phone`, and `email` as input parameters:

- Set the instance variables `name`, `phone`, and `email` to the corresponding input values.

b. Define the `__str__` method:

- Return a string representation of the contact details in the format: "Name: {name}\\nPhone: {phone}\\nEmail: {email}."

2 .Define the class `ContactManager:`

a. Define the `__init__` method:

- Initialize an empty list `contacts` to store the contacts.

b. Define the `add_contact` method that takes a `contact` object as an input parameter:

- Append the `contact` object to the `contacts` list.
- Print "Contact added successfully".

c. Define the `remove_contact` method that takes `name` as an input parameter:

- Iterate over each `contact` in the `contacts` list.
- If the `name` of the `contact` matches the input `name`, remove the `contact` from the `contacts` list.
- Print "Contact removed successfully".
- If no matching contact is found, print "Contact not found".

d. Define the `search_contact` method that takes `name` as an input parameter:

- Iterate over each `contact` in the `contacts` list.

- If the `name` of the `contact` matches the input `name`, return the `contact` object.

- If no matching contact is found, return `None.`

e. Define the `display_contacts` method:

- Check if the `contacts` list is empty.

- If it is empty, print "No contacts found".

- If there are contacts in the list, iterate over each `contact` and print its details.

3 .Create an instance of the `ContactManager` class called `contact_manager.`

4 .Start an infinite loop:

a. Print the menu options for the contact manager.

b. Read the user's choice from the input.

c. Based on the user's choice, perform the corresponding action:

- If the choice is "1", prompt the user to enter the contact details, create a `Contact` object, and call the `add_contact` method of `contact_manager.`

- If the choice is "2", prompt the user to enter the name of the contact to remove and call the `remove_contact` method of `contact_manager.`

- If the choice is "3", prompt the user to enter the name of the contact to search, call the `search_contact` method of `contact_manager`, and print the contact details if found.

- If the choice is "4", call the `display_contacts` method of `contact_manager` to print all the contacts.

- If the choice is "5", print "Exit Program" and break out of the loop.

- If the choice is not valid, print an error message.