ASSIGNMENT

By

Aman Puri

2023A6R030

Semester-II

CSE(AI/ML)

Model Institute of Engineering & Technology (Autonomous)

(Permanently Affiliated to the University of Jammu, Accredited by NAAC with “A” Grade)

Jammu, India

2024

ASSIGNMENT

Subject Code: COM-201

Subject Name: Data Structures using C

Due Date: 15 April 2024

Question Number	Course Outcomes	Blooms' Level	Maximum Marks	Marks Obtain
Q1	CO 2	3-6	10	
Q2	CO 3	3-6	10	
Total Marks			20	

Faculty Signature

Email:annusonania.cse@mietjammu.in

Assignment Objectives:

Assignment 1: Objective - Understand and implement algorithms for minimizing operations to distribute fruits among buckets.
Learning Outcome - Proficiency in solving optimization problems using incremental and doubling operations efficiently.

Assignment 2: Objective - Design and implement a console-based contact list application with CRUD functionalities. (Create, Read, Update, and Delete
Learning Outcome - Mastery in managing contact information, including adding, editing, searching, and deleting contacts within a console interface

Assignment Instructions:

1. *Group Size: Assignments will be completed in groups of 4-6 students.*
2. *Assessment Rubrics*
3. *Submission Method: Students should submit their completed assignments on CAMU in course COM-201 under "Assignment" chapter*
4. *Guidelines for Each Question:*

Assignment 1

1. Analyze the problem thoroughly, considering constraints and possible strategies for minimizing operations.
2. Implement and evaluate different algorithms, focusing on efficiency and effectiveness in solving the fruit distribution puzzle.

Assignment 2

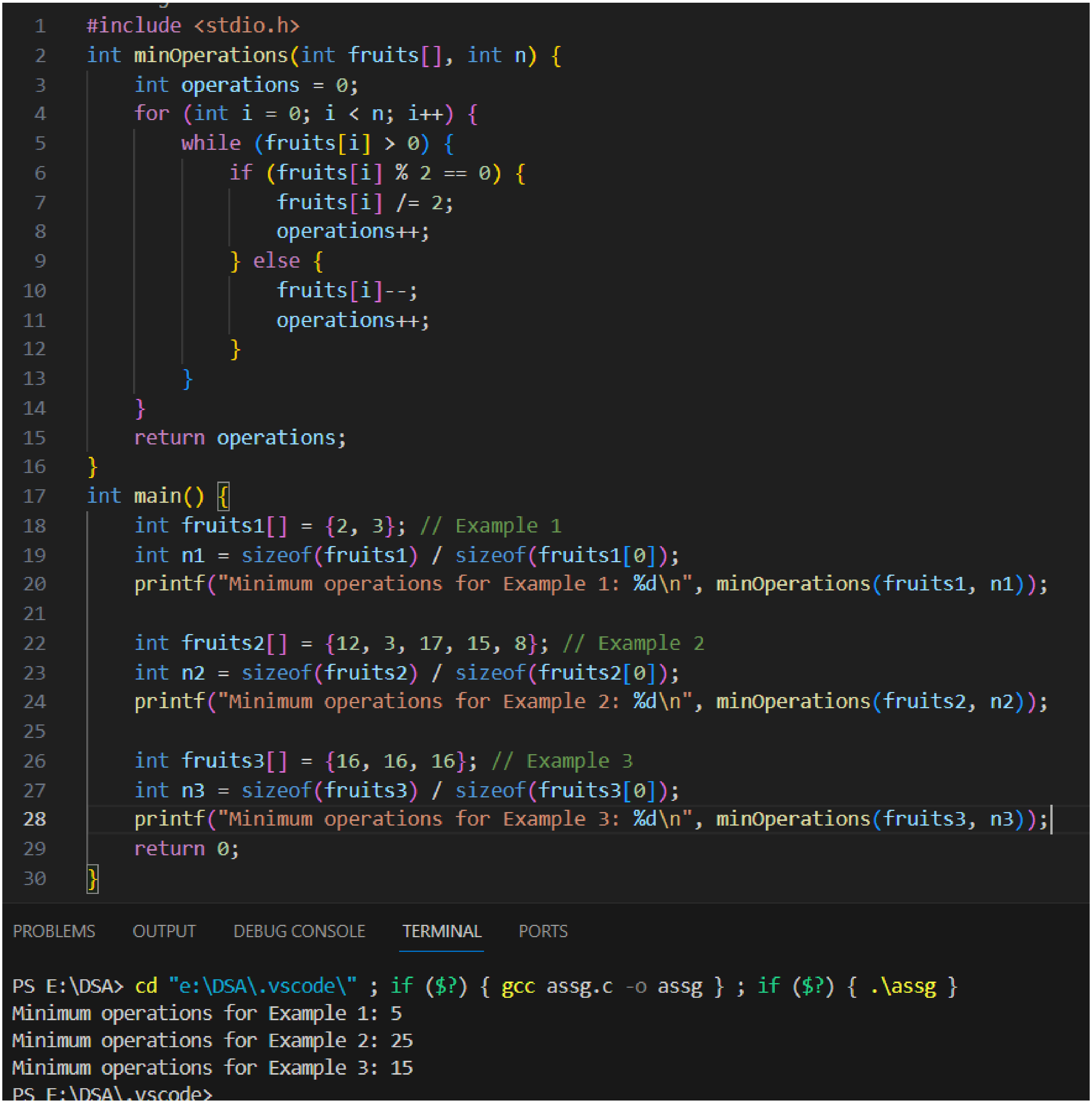
1. Design a console-based contact list application with clear functionalities for adding, editing, searching, and deleting contacts.
2. Ensure user-friendly interface and efficient data management, emphasizing usability and practicality in handling contact information.

Q. No.	Question	BL	CO	Marks	Total Marks

1	Jack and Jill went through a jungle to the city. They encountered a monster who told them they will only be allowed to escape when they solved a puzzle for him. They didn't have a choice, so they agreed. He states the problem like: I have **N** buckets having **0** fruits in each bucket initially. I will give you **N** numbers denoting fruits required at nth position. But you need to keep 2 simple rules: 1. Either you can increment fruit count by 1 in each bucket i.e. Incremental Operation 2. Or you can double the fruits in each bucket i.e. Doubling operation. Example 1: Suppose you have 2 buckets, and you need to put 2 fruits in the 1st bucket and 3 fruits in the 2nd bucket. Then the minimum operation required to do this task is 4. Example 2: Suppose you have 5 buckets, and you need to put 12 fruits in 1st, 3 fruits in 2nd, 17 fruit in 3rd, 15 fruits in 4th and 8 fruits in 5th bucket. Then the minimum operation required to do this task is 15. Example 3: Suppose you have 3 buckets, and you need to put 16 fruits in 1st, 16 fruits in 2nd and 16 fruits in 5th bucket. Then minimum operation required to do this task is 7.	3,4	CO 2	10	10

2	Design a contact list program. It is a simple console base application with no visuals. It's like a contact application you see on your mobile phones. Following are the essential elements of a contact List Application: 1. Add new contacts, including their name, phone number, company, and email address. 2. List all contacts displays a list of all the contacts, along with their contact information. 3. Search Contacts: Contacts may be found by searching by name and phone number. 4. Edit contacts: Make changes to the information provided when adding contacts, such as name, phone number, address, and email address. 5. Delete contacts removes contacts from the list.	5,6	CO 3	10	10

Q1.

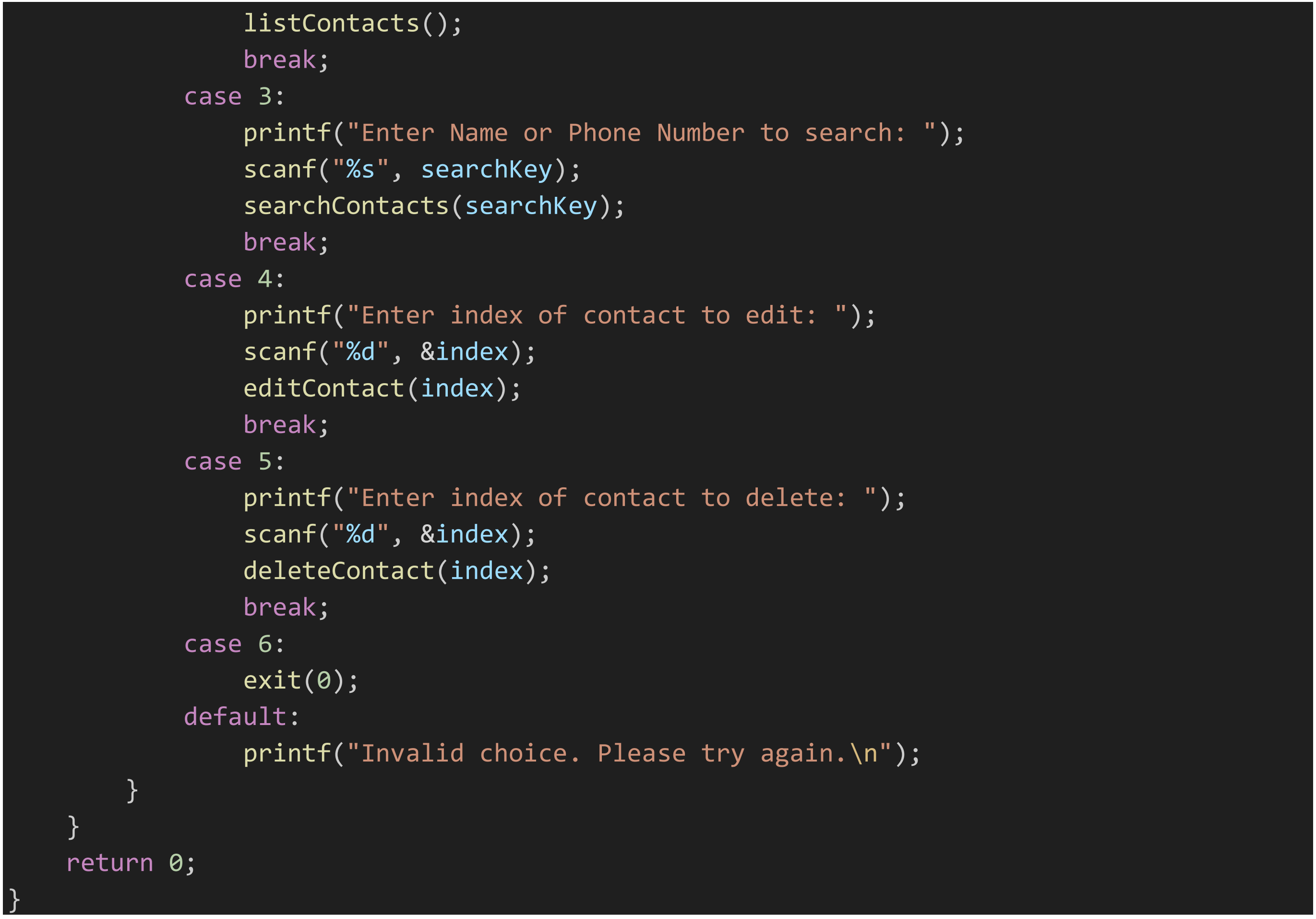


```c
#include <stdio.h>
int minOperations(int fruits[], int n) {
    int operations = 0;
    for (int i = 0; i < n; i++) {
        while (fruits[i] > 0) {
            if (fruits[i] % 2 == 0) {
                fruits[i] /= 2;
                operations++;
            } else {
                fruits[i]--;
                operations++;
            }
        }
    }
    return operations;
}
int main() {
    int fruits1[] = {2, 3}; // Example 1
    int n1 = sizeof(fruits1) / sizeof(fruits1[0]);
    printf("Minimum operations for Example 1: %d\n", minOperations(fruits1, n1));

    int fruits2[] = {12, 3, 17, 15, 8}; // Example 2
    int n2 = sizeof(fruits2) / sizeof(fruits2[0]);
    printf("Minimum operations for Example 2: %d\n", minOperations(fruits2, n2));

    int fruits3[] = {16, 16, 16}; // Example 3
    int n3 = sizeof(fruits3) / sizeof(fruits3[0]);
    printf("Minimum operations for Example 3: %d\n", minOperations(fruits3, n3));
    return 0;
}
```

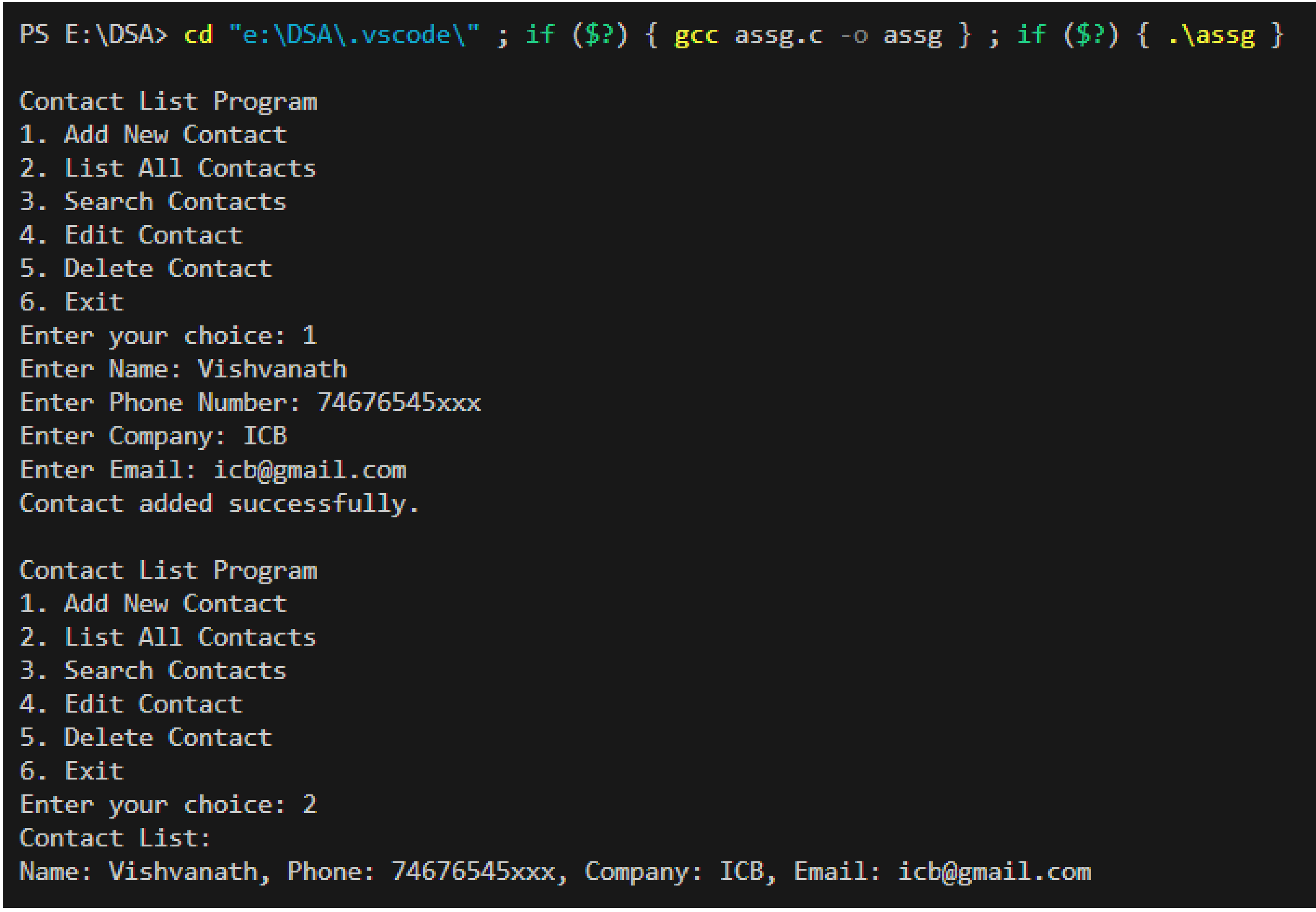
```
PS E:\DSA> cd "e:\DSA\.vscode\" ; if ($?) { gcc assg.c -o assg } ; if ($?) { .\assg }
Minimum operations for Example 1: 5
Minimum operations for Example 2: 25
Minimum operations for Example 3: 15
PS E:\DSA\.vscode>
```

Q2.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_CONTACTS 100
#define MAX_NAME 50
#define MAX_PHONE 15
#define MAX_COMPANY 50
#define MAX_EMAIL 50

char contacts[MAX_CONTACTS][MAX_NAME];
char phoneNumbers[MAX_CONTACTS][MAX_PHONE];
char companies[MAX_CONTACTS][MAX_COMPANY];
char emails[MAX_CONTACTS][MAX_EMAIL];
int top = -1;

void push() {
    if (top == MAX_CONTACTS - 1) {
        printf("Contact list is full.\n");
    } else {
        top++;
        printf("Enter Name: ");
        scanf("%s", contacts[top]);
        printf("Enter Phone Number: ");
        scanf("%s", phoneNumbers[top]);
        printf("Enter Company: ");
        scanf("%s", companies[top]);
        printf("Enter Email: ");
        scanf("%s", emails[top]);
        printf("Contact added successfully.\n");
    }
}
void listContacts() {
    if (top == -1) {
        printf("Contact list is empty.\n");
    } else {
        printf("Contact List:\n");
        for (int i = 0; i <= top; i++) {
            printf("Name: %s, Phone: %s, Company: %s, Email: %s\n", contacts[i],
phoneNumbers[i], companies[i], emails[i]);
        }
    }
}
void searchContacts(char searchKey[]) {
    int found = 0;
    for (int i = 0; i <= top; i++) {
        if (strcmp(contacts[i], searchKey) == 0 || strcmp(phoneNumbers[i], searchKey) ==
0) {
            printf("Contact found:\nName: %s, Phone: %s, Company: %s, Email: %s\n",
contacts[i], phoneNumbers[i], companies[i], emails[i]);
            found = 1;
        }
    }
    if (!found) {
        printf("Contact not found.\n");
    }
}
void editContact(int index) {
    printf("Enter new Name: ");
    scanf("%s", contacts[index]);
    printf("Enter new Phone Number: ");
    scanf("%s", phoneNumbers[index]);
    printf("Enter new Company: ");
    scanf("%s", companies[index]);
    printf("Enter new Email: ");
    scanf("%s", emails[index]);
    printf("Contact edited successfully.\n");
}
void deleteContact(int index) {
    for (int i = index; i < top; i++) {
        strcpy(contacts[i], contacts[i + 1]);
        strcpy(phoneNumbers[i], phoneNumbers[i + 1]);
        strcpy(companies[i], companies[i + 1]);
        strcpy(emails[i], emails[i + 1]);
    }
    top--;
    printf("Contact deleted successfully.\n");
}
int main() {
    int choice, index;
    char searchKey[MAX_NAME];

    while (1) {
        printf("\nContact List Program\n");
        printf("1. Add New Contact\n2. List All Contacts\n3. Search Contacts\n4. Edit
Contact\n5. Delete Contact\n6. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                push();
                break;
            case 2:
```



```
                listContacts();
                break;
            case 3:
                printf("Enter Name or Phone Number to search: ");
                scanf("%s", searchKey);
                searchContacts(searchKey);
                break;
            case 4:
                printf("Enter index of contact to edit: ");
                scanf("%d", &index);
                editContact(index);
                break;
            case 5:
                printf("Enter index of contact to delete: ");
                scanf("%d", &index);
                deleteContact(index);
                break;
            case 6:
                exit(0);
            default:
                printf("Invalid choice. Please try again.\n");
        }
    }
    return 0;
}
```



```
PS E:\DSA> cd "e:\DSA\.vscode\" ; if ($?) { gcc assg.c -o assg } ; if ($?) { .\assg }

Contact List Program
1. Add New Contact
2. List All Contacts
3. Search Contacts
4. Edit Contact
5. Delete Contact
6. Exit
Enter your choice: 1
Enter Name: Vishvanath
Enter Phone Number: 74676545xxx
Enter Company: ICB
Enter Email: icb@gmail.com
Contact added successfully.

Contact List Program
1. Add New Contact
2. List All Contacts
3. Search Contacts
4. Edit Contact
5. Delete Contact
6. Exit
Enter your choice: 2
Contact List:
Name: Vishvanath, Phone: 74676545xxx, Company: ICB, Email: icb@gmail.com
```