Green University of Bangladesh
Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: Fall, Year: 2023, B.Sc. in CSE (Day)

LAB REPORT NO: 07

Course Title: Structured Programming Lab

Course Code: CSE 104 Section: 231

Lab Experiment Name: Structure in C

<u>Student Details</u>

Name	ID
Promod Chandra Das	231002005

Lab Date : _

Submission Date : _

Course Teacher's Name : **Humayan Kabir Rana**

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: ………………………………… **Signature:.....................**

Comments:... **Date:..............................**

❖ TITLE OF THE LAB EXPERIMENT: - Structure in C

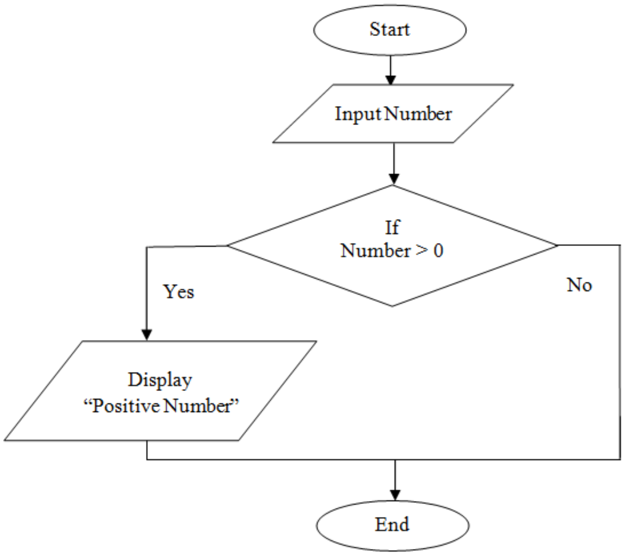
❖ OBJECTIVES/AIM :

Structure is a user-defined datatype in C language which allows us to combine data of different types together.
Structure helps to construct a complex data type which is more meaningful. It is somewhat similar to an Array,
but an array holds data of similar type only. But structure on the other hand, can store data of any type, which
is practical more useful.

- **Exercise :**

Write a C program to check a number is positive or negative:-

1. **Flow chart:**



2. Algorithm:-

1. Accept the input number from the user.
2. Check whether the number is greater than 0.
    - If true, print "The number is positive."
3. If the number is not greater than 0, check whether the number is less than 0.
    - If true, print "The number is negative."
4. If both conditions are false, print "The number is zero."

3. Pseudocode:

```
function checkNumber():
    input number
    if number > 0:
        print "The number is positive."
    else if number < 0:
        print "The number is negative."
    else:
        print "The number is zero."

Call checkNumber()
```

4. C program:-

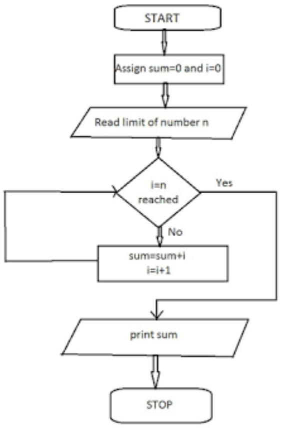
```c
#include <stdio.h>

int main() {
    // Variable to store the input number
    int number;

    // Input
    printf("Enter a number: ");
    scanf("%d", &number);

    // Check whether the number is positive or negative
    if (number > 0) {
        printf("The number is positive.\n");
    } else if (number < 0) {
        printf("The number is negative.\n");
    } else {
        printf("The number is zero.\n");
    }

    return 0;
}
```

5. **RESULT / OUTPUT**

```
Enter a number: 42
The number is positive.
```

- Write a C program for input N numbers in array and print the summation of all value in array.

1.**Flow chart:**



2. Algorithm:-

1.Accept the number of elements (N) in the array from the user.

2.Check if the number of elements is within the array size limit.

3. If valid, input N numbers into the array.
4. Calculate the summation of values in the array.
5. Print the summation.

3.Pseudocode:

```
function calculateSum():
    input n
    if n is within array size limit:
        input numbers into the array
        sum = 0
        for each number in the array:
            sum += number
        print "Summation of values in the array:", sum
    else:
        print "Invalid number of elements. Please enter a value between 1 and MAX_SIZE."

Call calculateSum()
```

4.C program:-

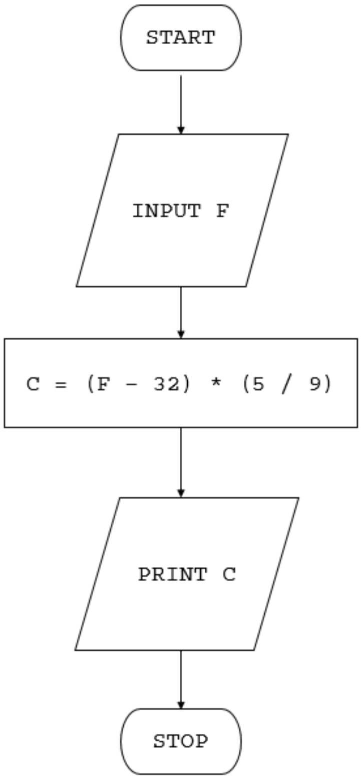
```
#include <stdio.h>

int main() {
    // Maximum size of the array
    #define MAX_SIZE 100

    // Variable to store the number of elements in the array
    int n;
    // Array to store the input numbers
    int numbers[MAX_SIZE];

    // Variable to store the summation of values in the array
    int sum = 0;

    // Input
    printf("Enter the number of elements (N) in the array: ");
    scanf("%d", &n);

    // Check if the number of elements is within the array size limit
    if (n > 0 && n <= MAX_SIZE) {
        printf("Enter %d numbers:\n", n);

        // Input numbers into the array
        for (int i = 0; i < n; i++) {
            printf("Enter number %d: ", i + 1);
            scanf("%d", &numbers[i]);
        }

        // Calculate the summation of values in the array
        for (int i = 0; i < n; i++) {
            sum += numbers[i];
        }

        // Output
        printf("Summation of values in the array: %d\n", sum);
    } else {
        printf("Invalid number of elements. Please enter a value between 1 and %d.\n", MAX_SIZE);
    }

    return 0;
}
```

RESULT / OUTPUT

```
Enter the number of elements (N) in the array: 5
Enter 5 numbers:
Enter number 1: 10
Enter number 2: 20
Enter number 3: 30
Enter number 4: 40
Enter number 5: 50
Summation of values in the array: 150
```

- **Write a C program to calculate Celsius to Fahrenheit**

1. **Flow chart:**



2. Algorithm:-

1. Accept the temperature in Celsius from the user.
2. Use the formula Fahrenheit=95×Celsius+32Fahrenheit=59×Celsius+32 to convert Celsius to Fahrenheit.
3. Print the temperature in Fahrenheit.

3.Pseudocode:

```
function celsiusToFahrenheit():
    input celsius
    fahrenheit = (celsius * 9 / 5) + 32
    print "Temperature in Fahrenheit:", fahrenheit

Call celsiusToFahrenheit()
```

4.C program:-

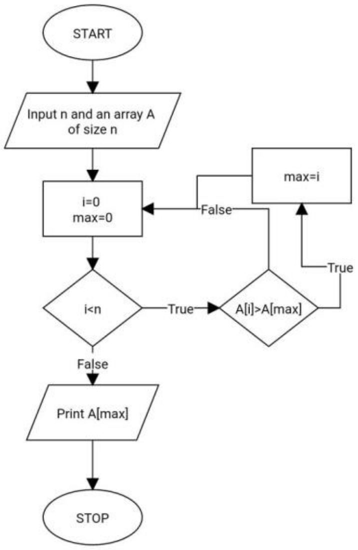
```c
#include <stdio.h>

int main() {
    // Variable to store temperature in Celsius
    float celsius;

    // Variable to store temperature in Fahrenheit
    float fahrenheit;

    // Input
    printf("Enter temperature in Celsius: ");
    scanf("%f", &celsius);

    // Conversion from Celsius to Fahrenheit
    fahrenheit = (celsius * 9 / 5) + 32;

    // Output
    printf("Temperature in Fahrenheit: %.2f\n", fahrenheit);
    return 0;
}
```

5.RESULT / OUTPUT

```
Enter temperature in Celsius: 25
Temperature in Fahrenheit: 77.00
```

- **Write a C Program to take input N size array and print reverse array.**

1**. Flow chart:**



2. Algorithm:-

1. Accept the number of elements (N) in the array from the user.
2. Check if the number of elements is within the array size limit.
3. If valid, input N numbers into the array.
4. Output the original array.
5. Output the reverse array.

3. Pseudocode:

```
function reverseArray():
    input n
    if n is within array size limit:
        input numbers into the array
        print "Original array:"
        for each number in the array:
            print number
        print "Reverse array:"
        for each number in the array in reverse order:
            print number
    else:
        print "Invalid number of elements. Please enter a value between 1 and MAX_SIZE."

Call reverseArray()
```

4. **C program:-**

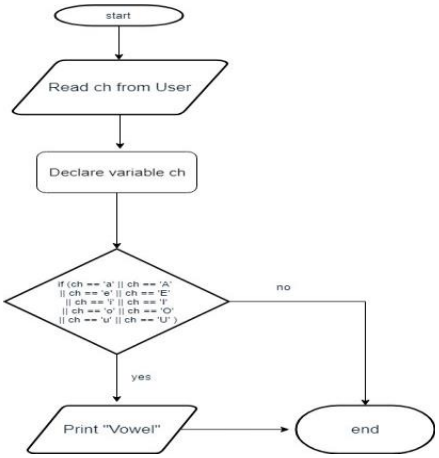
```
#include <stdio.h>

int main() {
    // Maximum size of the array
    #define MAX_SIZE 100

    // Variable to store the number of elements in the array
    int n;

    // Array to store the input numbers
    int numbers[MAX_SIZE];

    // Input
    printf("Enter the number of elements (N) in the array: ");
    scanf("%d", &n);

    // Check if the number of elements is within the array size limit
    if (n > 0 && n <= MAX_SIZE) {
        printf("Enter %d numbers:\n", n);

        // Input numbers into the array
        for (int i = 0; i < n; i++) {
            printf("Enter number %d: ", i + 1);
            scanf("%d", &numbers[i]);
        }
        // Output the original array
        printf("Original array: ");
        for (int i = 0; i < n; i++) {
            printf("%d ", numbers[i]);
        }
        printf("\n");

        // Output the reverse array
        printf("Reverse array: ");
        for (int i = n - 1; i >= 0; i--) {
            printf("%d ", numbers[i]);
        }
        printf("\n");
    } else {
        printf("Invalid number of elements. Please enter a value between 1 and %d.\n",
MAX_SIZE);
    }

    return 0;
}
```

5. RESULT / OUTPUT

```
Enter the number of elements (N) in the array: 4
Enter 4 numbers:
Enter number 1: 10
Enter number 2: 20
Enter number 3: 30
Enter number 4: 40
Original array: 10 20 30 40
Reverse array: 40 30 20 10
```

- **Write a C program to check a character is a vowel or not.**

1. Flow chart:


start
Read ch from User
Declare variable ch
if (ch == 'a' || ch == 'A'
|| ch == 'e' || ch == 'E'
|| ch == 'i' || ch == 'I'
|| ch == 'o' || ch == 'O'
|| ch == 'u' || ch == 'U')
no
yes
Print "Vowel"
end

2. Algorithm:-

1. Accept a character from the user.
2. Check if the character is 'a', 'e', 'i', 'o', or 'u' (both uppercase and lowercase).
3. If true, print that the character is a vowel.
4. If false, print that the character is not a vowel.

3. Pseudocode:

```
function checkVowel():
    input character
    if character is 'a' or 'A' or 'e' or 'E' or 'i' or 'I' or 'o' or 'O' or 'u' or 'U':
        print "Character is a vowel."
    else:
        print "Character is not a vowel."

Call checkVowel()
```

4.C program:-

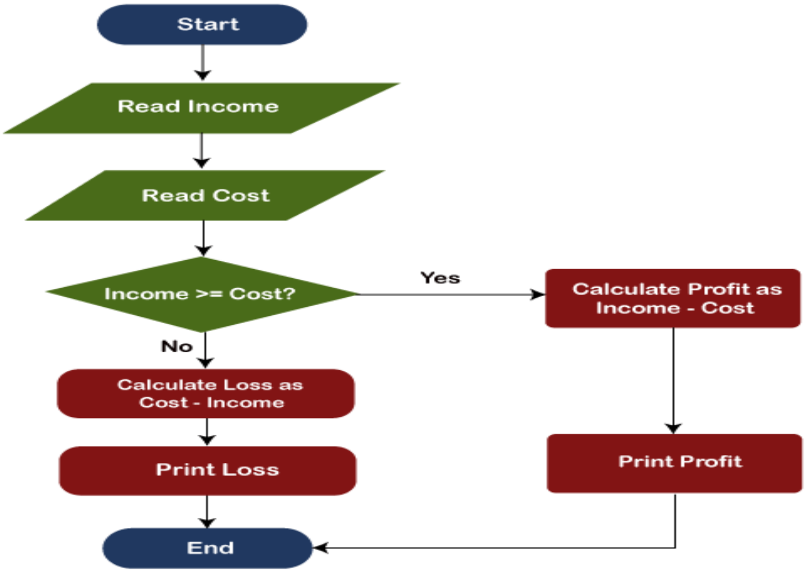
```
#include <stdio.h>

int main() {
    // Variable to store the input character
    char ch;

    // Input
    printf("Enter a character: ");
    scanf(" %c", &ch);

    // Check if the character is a vowel
    if ((ch == 'a' || ch == 'A') || (ch == 'e' || ch == 'E') ||
        (ch == 'i' || ch == 'I') || (ch == 'o' || ch == 'O') ||
        (ch == 'u' || ch == 'U')) {
        printf("%c is a vowel.\n", ch);
    } else {
        printf("%c is not a vowel.\n", ch);
    }

    return 0;
}
```

5. RESULT / OUTPUT

```
Enter a character: A
A is a vowel.
```

- **Write a C program using nested structure to print your name and address with location.**

1. Flow chart:



2. Algorithm:-

1. Create structures for Location, Address, and Person.
2. Accept input for the person's first name, last name, street, zip code, city, state, and country.
3. Print the person's name and address information.

4. Pseudocode:

```
structure Location:
    city
    state
    country

structure Address:
    street
    zipCode
    location (nested structure)

structure Person:
    firstName
    lastName
    address (nested structure)

function main():
    create a person structure

    input first name
    input last name
    input street
    input zip code
    input city
    input state
    input country

    print "Personal Information:"
    print "Name: firstName lastName"
    print "Address: street, zipCode, city, state, country"
```

```
Call main()
```

5. C program:-

```
#include <stdio.h>

// Define a structure for location
struct Location {
    char city[50];
    char state[50];
    char country[50];
};

// Define a structure for address
struct Address {
    char street[100];
    int zipCode;
    struct Location location; // Nested structure
};

// Define a structure for personal information
struct Person {
    char firstName[50];
    char lastName[50];
    struct Address address; // Nested structure
};257   8

int main() {
    // Create a person structure
    struct Person person;

    // Input
    printf("Enter first name: ");
    scanf("%s", person.firstName);

    printf("Enter last name: ");
    scanf("%s", person.lastName);

    printf("Enter street: ");
    scanf("%s", person.address.street);

    printf("Enter zip code: ");
    scanf("%d", &person.address.zipCode);

    printf("Enter city: ");
    scanf("%s", person.address.location.city);

    printf("Enter state: ");
    scanf("%s", person.address.location.state);

    printf("Enter country: ");
    scanf("%s", person.address.location.country);

    // Output
    printf("\nPersonal Information:\n");
    printf("Name: %s %s\n", person.firstName, person.lastName);
    printf("Address: %s, %d, %s, %s, %s\n", person.address.street, person.address.zipCode,
           person.address.location.city,                    person.address.location.state,
person.address.location.country);

    return 0;
}
```

5. RESULT / OUTPUT

```
Enter first name: John
Enter last name: Doe
Enter street: 123 Main Street
Enter zip code: 12345
Enter city: Anytown
Enter state: CA
Enter country: USA

Personal Information:
Name: John Doe
Address: 123 Main Street, 12345, Anytown, CA, USA
```

❖ DISCUSSION:-

In C programming, a structure is a user-defined data type that allows you to group different variables under a single name. Each variable within a structure is called a member, and these members can different data types. Structures are useful for organizing and representing related pieces of information in a more cohesive manner.