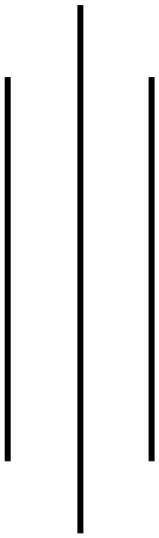
Tribhuwan University
Institute of Engineering
Purbanchal Campus, Dharan

C Lab Report



Submitted By:

Name: Tilak Thapa

Roll No: PUR079BCT094

Submitted To:

Department of Electronic and Computer Engineering

Lab Date:

Submission Date:

Signature:

Table of Contents

Lab Sheet 1

1. WAP to display hello world.

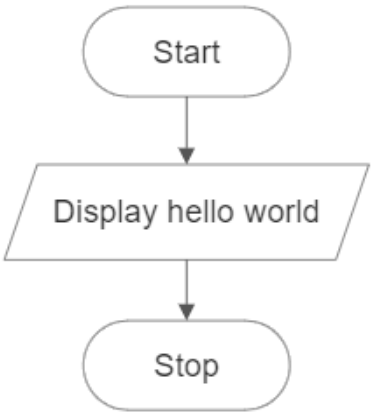
➢ Objective

The objective of this program is to write a code that displays the message "Hello, World!" on the console.

➢ Algorithm

i. Start
ii. Display "Hello, World!"
iii. Stop

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main( ) {
   printf("Hello, World!\n");
   return 0;
}
```

➢ Output

```
Hello, World!
```

➢ Discussion and Conclusion

This program displays the message "Hello, World!" on the console. The printf function is used to print the message. The \n is used to add a new line after the message. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file.

2. WAP to display your name, roll number and address

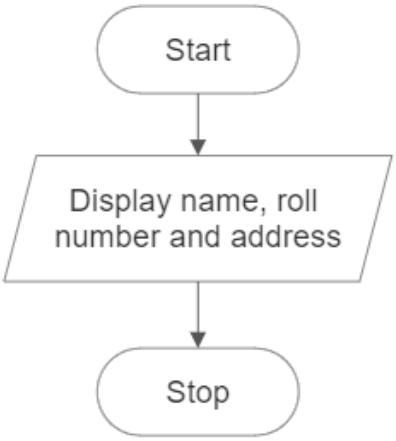
➢ Objective

The objective of this program is to write a code that displays your name, roll number, and address on the console.

➢ Algorithm

i. Start
ii. Display name, roll number and address
iii. Stop

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main( )
{
   printf("Name: Tilak Thapa\n");
   printf("Roll Number: PUR079BCT094\n");
   printf("Address: Tulsipur - 4, Dang\n");
   return 0;
}
```

➢ Output

```
Name: Tilak Thapa
Roll Number: PUR079BCT094
Address: Tulsipur - 4, Dang
```

➢ Discussion and Conclusion

This program displays my name, roll number, and address on the console. The printf function is used to print each piece of information. The newline character \n is used to add a new line after each line of output. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file.

3. WAP to add two integer variables and print sum

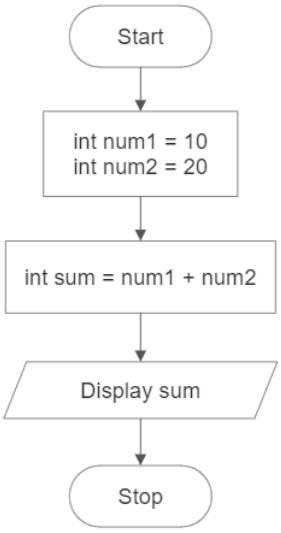
➢ Objective

The objective of this program is to write a code that adds two predefined integer variables and prints their sum.

➢ Algorithm

i. Start.
ii. Declare two integer variables, num1 and num2, and initialize them with predefined values.
iii. Calculate the sum of num1 and num2 and store it in a variable called sum.
iv. Print the value of sum.
v. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main( )
{
   int num1 = 10;
   int num2 = 20;
   int sum = num1 + num2;
   printf("Sum: %d\n", sum);
   return 0;
}
```

➢ Output

```
Sum: 30
```

➢ Discussion and Conclusion

This program adds two predefined integer variables, num1 and num2, and prints their sum. The values of num1 and num2 are initialized with the numbers 10 and 20, respectively. The sum of num1 and num2 is calculated and stored in the sum variable using the addition operator (+). The printf function is used to display the value of sum. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file. The objective of the program was achieved, and the code executed successfully.

4. WAP to multiply two integer variables and print product

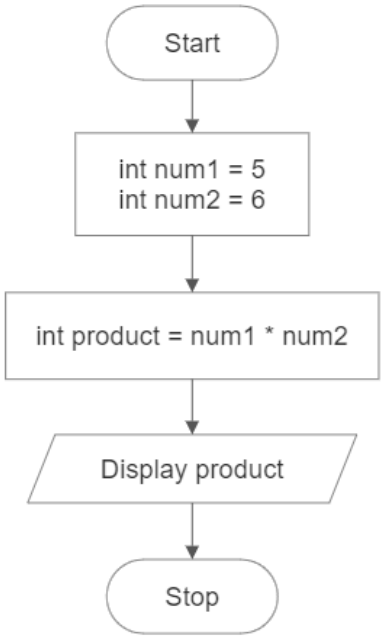
➢ Objective

The objective of this program is to write a code that multiplies two integer variables and prints their product.

➢ Algorithm

i. Start.
ii. Declare two integer variables, num1 and num2, and initialize them with predefined values.
iii. Calculate the product of num1 and num2 and store it in a variable called product.
iv. Print the value of product.
v. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main()
{
   int num1 = 5;
   int num2 = 6;
   int product = num1 * num2;
   printf("Product: %d\n", product);
   return 0;
}
```

➢ Output

```
Product: 30
```

➢ Discussion and Conclusion

This program multiplies two integer variables, num1 and num2, and prints their product. The values of num1 and num2 are assigned as 5 and 6, respectively. The product of num1 and num2 is calculated and stored in the product variable using the multiplication operator (*). The printf function is used to display the value of product. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file. The objective of the program was achieved, and the code executed successfully.

5. WAP to calculate and display the simple interest.

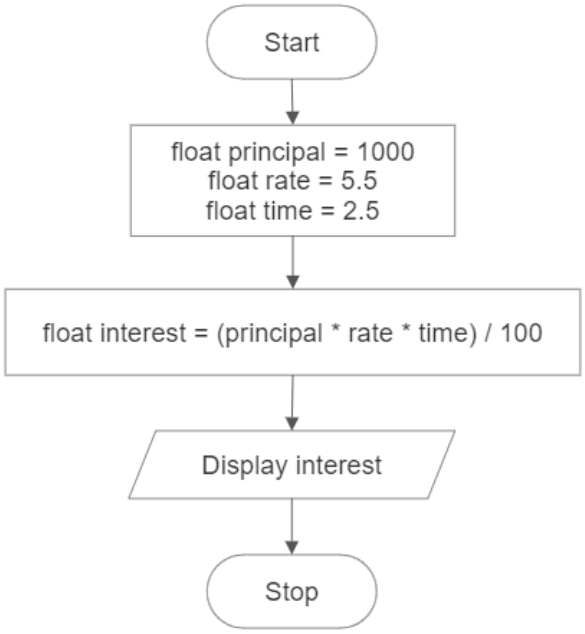
➢ Objective

The objective of this program is to write a code that calculates and displays the simple interest based on predefined values for principal amount, rate, and time.

➢ Algorithm

i. Start.
ii. Declare and initialize three variables: principal, rate, and time with predefined values.
iii. Calculate the simple interest using the formula: interest = (principal * rate * time) / 100 and assign the value to variable called interest.
iv. Print the value of the interest.
v. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main()
{
   float principal = 1000;
   float rate = 5.5;
   float time = 2.5;
   float interest = (principal * rate * time) / 100;
   printf("Simple Interest: Rs %f\n", interest);
   return 0;
}
```

➢ Output

```
Interest: Rs 137.500000
```

➢ Discussion and Conclusion

This program calculates and displays the simple interest based on predefined values for the principal amount, rate of interest, and time period. The values of principal, rate, and time are initialized as 1000, 5.5, and 2.5, respectively. The simple interest is calculated using the formula: interest = (principal * rate * time) / 100. The calculated interest value is then printed using the printf function. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file.

6. WAP to calculate the area of the circle

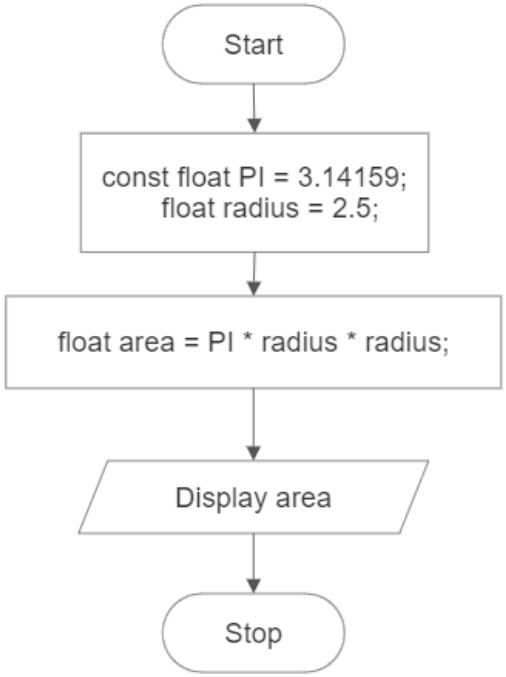
➢ Objective

The objective of this program is to write a code that calculates the area of a circle based on a predefined radius.

➢ Algorithm

i. Start.
ii. Declare a constant variable for PI and a variable for the radius and assign their values.
iii. Calculate the area of the circle using the formula: area = pi * radius * radius and assign the value to a variable called area.
iv. Print the value of the area.
v. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main( )
{
   const float PI = 3.14159;
   float radius = 2.5;
   float area = PI * radius * radius;
   printf("Area of the circle: %.2f sq unit.\n", area);
   return 0;
}
```

➢ Output

```
Area of the circle: 19.63 sq unit.
```

➢ Discussion and Conclusion:

This program calculates the area of a circle based on a predefined radius. The value of radius is assigned as 2.5. The area of the circle is calculated using the formula: area = PI * radius * radius, where PI is a constant value representing the mathematical constant pi (approximately 3.14159). The calculated area value is then printed using the printf function. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file

Lab Sheet 2

1. WAP to declare integer, float and character variable. Initialize them with certain value and print those values. Also display the size of variables.

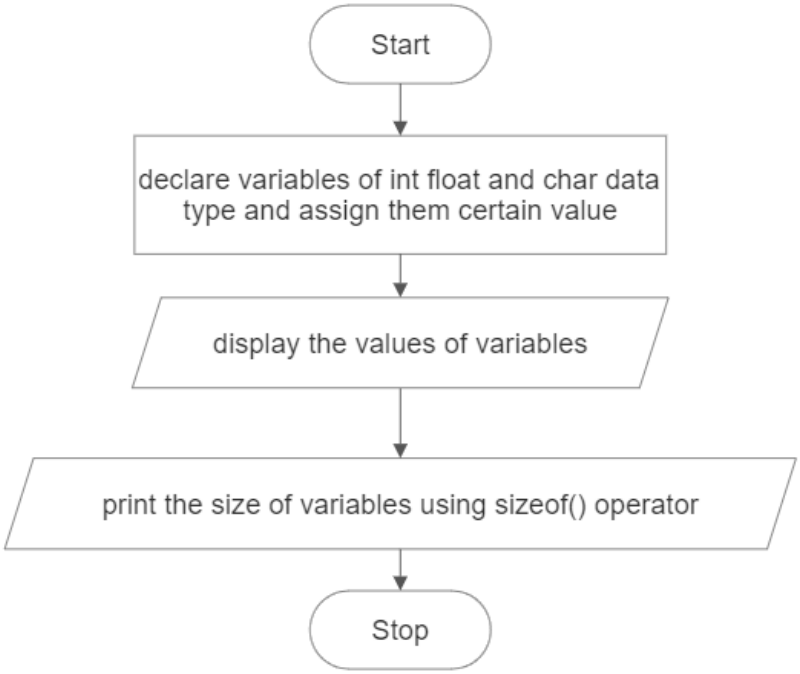
➢ Objective

The objective of this program is to write a code that declares integer, float, and character variables, initializes them with certain values, and prints the values. Additionally, the program will display the size of each variable.

➢ Algorithm

i. Start.
ii. Declare an integer variable and assign certain value.
iii. Declare a float variable and assign certain value.
iv. Declare a character variable and assign certain value.
v. Print the values of the variables using the printf function.
vi. Use the sizeof() operator to determine the size of each variable and print the sizes of the variables.
vii. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main( )
{
   int integerVariable = 10;
   float floatVariable = 3.14;
   char charVariable = 'A';

   printf("Integer Variable: %d\n", integerVariable);
   printf("Float Variable: %f\n", floatVariable);
   printf("Character Variable: %c\n\n", charVariable);

   printf("Size of Integer Variable: %d bytes\n", sizeof(integerVariable));
   printf("Size of Float Variable: %d bytes\n", sizeof(floatVariable));
   printf("Size of Character Variable: %d bytes\n", sizeof(charVariable));

   return 0;
}
```

➢ Output

```
Integer Variable: 10
Float Variable: 3.140000
Character Variable: A

Size of Integer Variable: 4 bytes
Size of Float Variable: 4 bytes
Size of Character Variable: 1 bytes
```

➢ Discussion and Conclusion

This program declares an integer variable, a float variable, and a character variable. The variables are initialized with certain values. The values of the variables are printed using the printf function. The sizeof operator is used to determine the size of each variable, and the sizes are printed. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file.

2. WAP to swap the values of the variable with and without using third variable.

➢ **Objective**

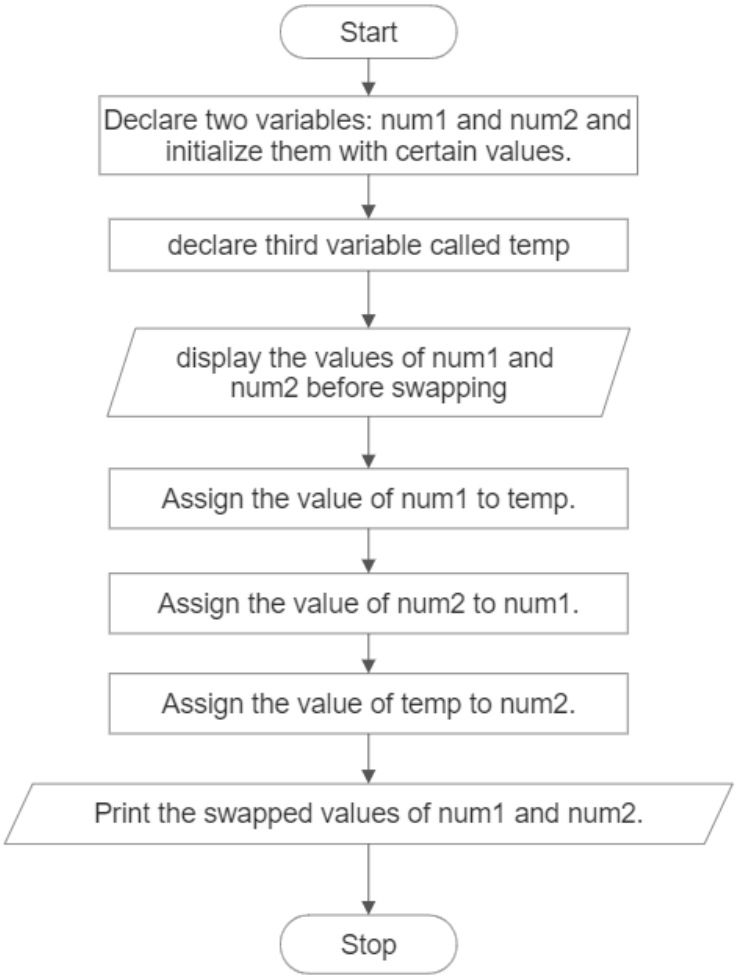
The objective of this program is to write a code that swaps the values of two variables, both with and without using a third variable.

I. Approach 1 (Using third variable)

➢ **Algorithm**

i. Start.
ii. Declare two variables: num1 and num2 and initialize them with certain values.
iii. Declare a third variable, temp.
iv. Display the value of num1 and num2 before swapping.
v. Assign the value of num1 to temp.
vi. Assign the value of num2 to num1.
vii. Assign the value of temp to num2.
viii. Print the swapped values of num1 and num2.
ix. Stop.

➢ **Flowchart**



➢ Code

```
#include <stdio.h>

int main( )
{
   int num1 = 10;
   int num2 = 20;
   int temp;

   printf("Before swapping:\n");
   printf("num1 = %d\n", num1);
   printf("num2 = %d\n", num2);

   temp = num1;
   num1 = num2;
   num2 = temp;

   printf("After swapping (using third variable):\n");
   printf("num1 = %d\n", num1);
   printf("num2 = %d\n", num2);

   return 0;
}
```

➢ Output

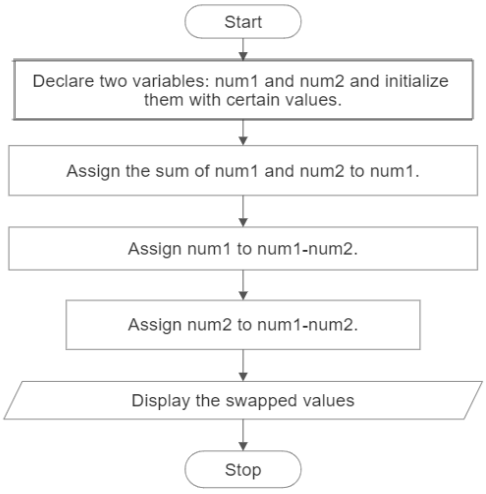
```
Before swapping:
num1 = 10
num2 = 20
After swapping (using third variable):
num1 = 20
num2 = 10
```

II. Approach 2 (Without Using Third Variable)

➢ Algorithm

i. Start.
ii. Declare two variables: num1 and num2 and initialize them with certain values.
iii. Display the value of num1 and num2 before swapping.
iv. Assign the sum of num1 and num2 to num1.
v. Assign num1 to num1-num2.
vi. Again, assign num2 to num1-num2.
vii. Print the swapped values of num1 and num2.
viii. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main()
{
   int num1 = 10;
   int num2 = 20;

   printf("Before swapping:\n");
   printf("num1 = %d\n", num1);
   printf("num2 = %d\n", num2);

   num1 = num1 + num2;
   num2 = num1 - num2;
   num1 = num1 - num2;

   printf("After swapping (without using third variable):\n");
   printf("num1 = %d\n", num1);
   printf("num2 = %d\n", num2);

   return 0;
}
```

➢ Output

```
Before swapping:
num1 = 10
num2 = 20
After swapping (without using third variable):
num1 = 20
num2 = 10
```

➢ Discussion and Conclusion

In the first part of the program, the values of num1 and num2 are swapped using a third variable. The values are stored in a temporary variable, temp, before swapping. Then, the values are exchanged by assigning num2 to num1 and temp to num2.

In the second part of the program, the values of num1 and num2 are swapped without using a third variable. This is achieved using the simple addition and subtraction operation. By performing that operations on the two variables, the original values are swapped without the need for an additional variable.

Both cases print the values of num1 and num2 before and after swapping to demonstrate the results. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file.

3. WAP to calculate the area and volume of a cylinder using pre-processor directive for value of PI.

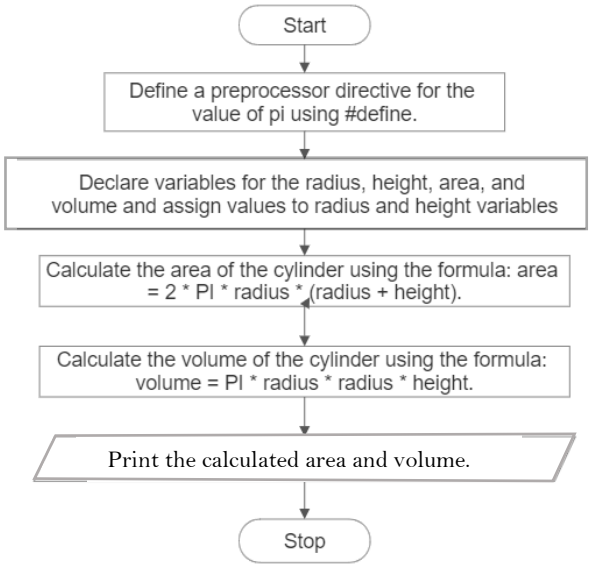
➢ Objective

The objective of this program is to write a code that calculates the area and volume of a cylinder using a preprocessor directive for the value of pi.

➢ Algorithm

i. Start.
ii. Define a preprocessor directive for the value of pi using #define.
iii. Declare variables for the radius, height, area, and volume.
iv. Assign predetermined values to the radius and height variables.
v. Calculate the area of the cylinder using the formula: area = 2 * PI * radius * (radius + height).
vi. Calculate the volume of the cylinder using the formula: volume = PI * radius * radius * height.
vii. Print the calculated area and volume.
viii. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

#define PI 3.14159

int main()
{
   float radius = 2.5;
   float height = 5.0;
   float area, volume;

   area = 2 * PI * radius * (radius + height);
   volume = PI * radius * radius * height;

   printf("Area of the cylinder: %.2f\n", area);
   printf("Volume of the cylinder: %.2f\n", volume);

   return 0;
}
```

➢ Output

```
Area of the cylinder: 117.81
Volume of the cylinder: 98.17
```

➢ Discussion and Conclusion

This program calculates the area and volume of a cylinder using a preprocessor directive for the value of pi. The values of the radius and height are predetermined and assigned to the respective variables. The area of the cylinder is calculated using the formula: area = 2 * PI * radius * (radius + height), and the volume is calculated using the formula: volume = PI * radius * radius * height. The calculated values are then printed using the printf function. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file.

4. WAP to input two numbers from user and display the minimum using conditional operator.

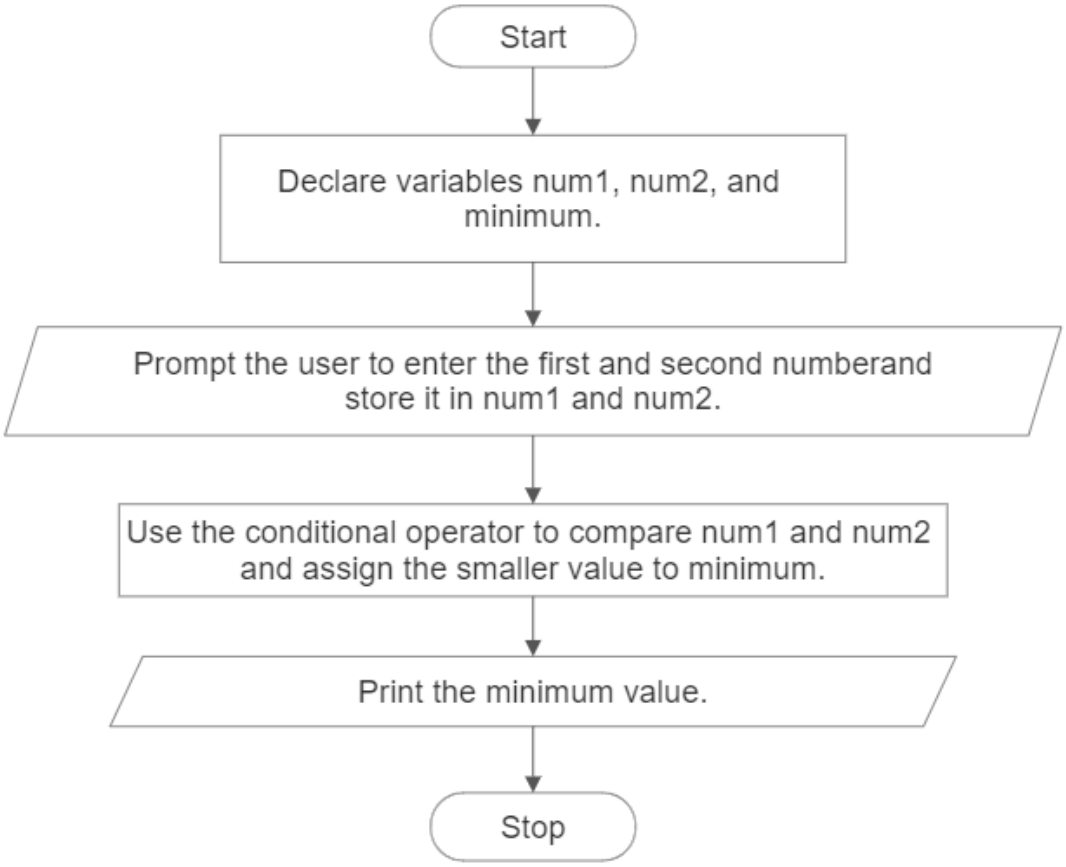
➢ **Objective**

The objective of this program is to write a code that takes two numbers as input from the user and displays the minimum of the two numbers using the conditional operator.

➢ **Algorithm**

i. Start.
ii. Declare variables num1, num2, and minimum.
iii. Prompt the user to enter the first number and store it in num1.
iv. Prompt the user to enter the second number and store it in num2.
v. Use the conditional operator to compare num1 and num2 and assign the smaller value to minimum.
vi. Print the value of minimum.
vii. Stop.

➢ **Flowchart**



➢ Code

```
#include <stdio.h>

int main( )
{
   int num1, num2, minimum;

   printf("Enter the first number: ");
   scanf("%d", &num1);

   printf("Enter the second number: ");
   scanf("%d", &num2);

   minimum = (num1 < num2) ? num1 : num2;

   printf("The minimum number is: %d\n", minimum);

   return 0;
}
```

➢ Output

```
Enter the first number: 4
Enter the second number: 5
The minimum number is: 4
```

➢ Discussion and Conclusion

This program takes two numbers as input from the user and determines the minimum of the two numbers using the conditional operator. The user is prompted to enter the first number and the second number, which are stored in num1 and num2 variables, respectively. The conditional operator (num1 < num2) ? num1 : num2 compares the two numbers and assigns the smaller value to the minimum variable. Finally, the minimum value is printed using the printf function. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file.

5. WAP to display whether a number is even or odd using conditional operator

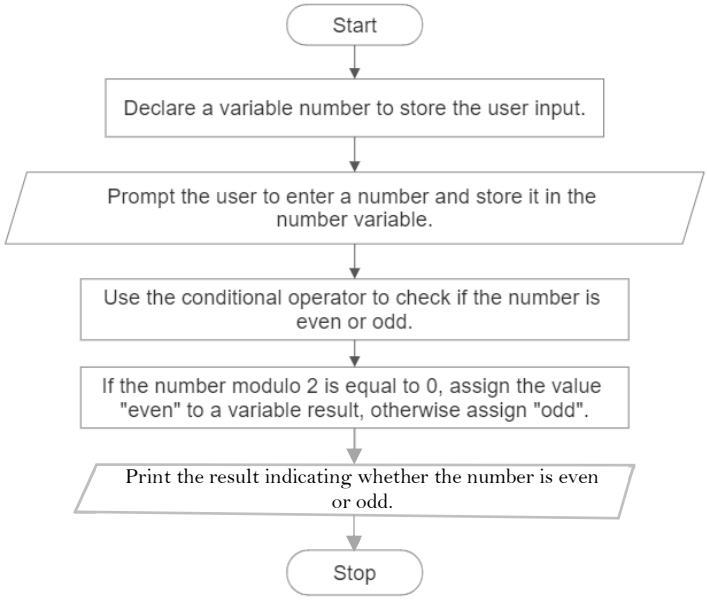
➢ Objective

The objective of this program is to write a code that takes a number as input from the user and displays whether the number is even or odd using the conditional operator.

➢ Algorithm

i. Start.
ii. Declare a variable number to store the user input.
iii. Prompt the user to enter a number and store it in the number variable.
iv. Use the conditional operator to check if the number is even or odd.
v. If the number % 2 is equal to 0, assign the value "even" to a variable result, otherwise assign "odd".
vi. Print the result indicating whether the number is even or odd.
vii. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main( )
{
   int number;
   char *result;
   printf("Enter a number: ");
   scanf("%d", &number);
   result = (number % 2 == 0) ? "even" : "odd";
   printf("The number is %s.\n", result);
   return 0;
}
```

➢ Output

```
Enter a number: 6
The number is even.
```

➢ Discussion and Conclusion

This program takes a number as input from the user and determines whether the number is even or odd using the conditional operator. The user is prompted to enter a number, which is stored in the number variable. The conditional operator (number % 2 == 0) ? "even" : "odd" checks if the number modulo 2 is equal to 0. If it is, the value "even" is assigned to the result variable; otherwise, the value "odd" is assigned. Finally, the program prints the result indicating whether the number is even or odd. The program was implemented using the VS Code IDE and compiled using gcc to generate an executable file.

6. What are the output of the following programs:

➢ Objective

The objective is to find the input of given code.

➢ Code

```
#include <stdio.h>
int main()
{
   int a = 5, b = 9;
   printf("a = %d, b = %d\n", a, b);
   printf("a&b = %d\n", a & b);
   printf("a|b = %d\n", a | b);
   printf("a^b = %d\n", a ^ b);
   printf("~a = %d\n", ~a);
   printf("(b<<2)+(a<<1) = %d\n", (b << 2) + (a << 1));
   printf("(b>>1)+(a>>1) = %d\n", (b >> 1) + (a >> 1));
   return 0;
}
```

➢ Output

```
a = 5, b = 9
a&b = 1
a|b = 13
a^b = 12
~a = -6
(b<<2)+(a<<1) = 46
(b>>1)+(a>>1) = 6
```

➢ Discussion and Conclusion

The program displays the output showing the effects of the bitwise operations on the given variables.

- a & b performs a bitwise AND operation between a and b. In binary, 5 is 0101 and 9 is 1001. The result of a & b is 0001, which is equal to 1 in decimal.
- a | b performs a bitwise OR operation between a and b. In binary, the result is 1101, which is equal to 13 in decimal.
- a ^ b performs a bitwise XOR operation between a and b. In binary, the result is 1100, which is equal to 12 in decimal.
- ~a performs a bitwise complement operation on a. In binary, the result is 11111111111111111111111111111010, which is equal to -6 in decimal due to two's complement representation.
- (b<<2)+(a<<1) performs a left shift by 2 bits on b and a left shift by 1 bit on a, then adds the results. The value of b after the left shift is 36, and the value of a after the left shift is 10. The final result is 46.
- (b>>1)+(a>>1) performs a right shift by 1 bit on both b and a, then adds the results. The value of b after the right shift is 4, and the value of a after the right shift is 2. The final result is 7.

1. Write a C program to check whether a number is negative, positive, or zero.

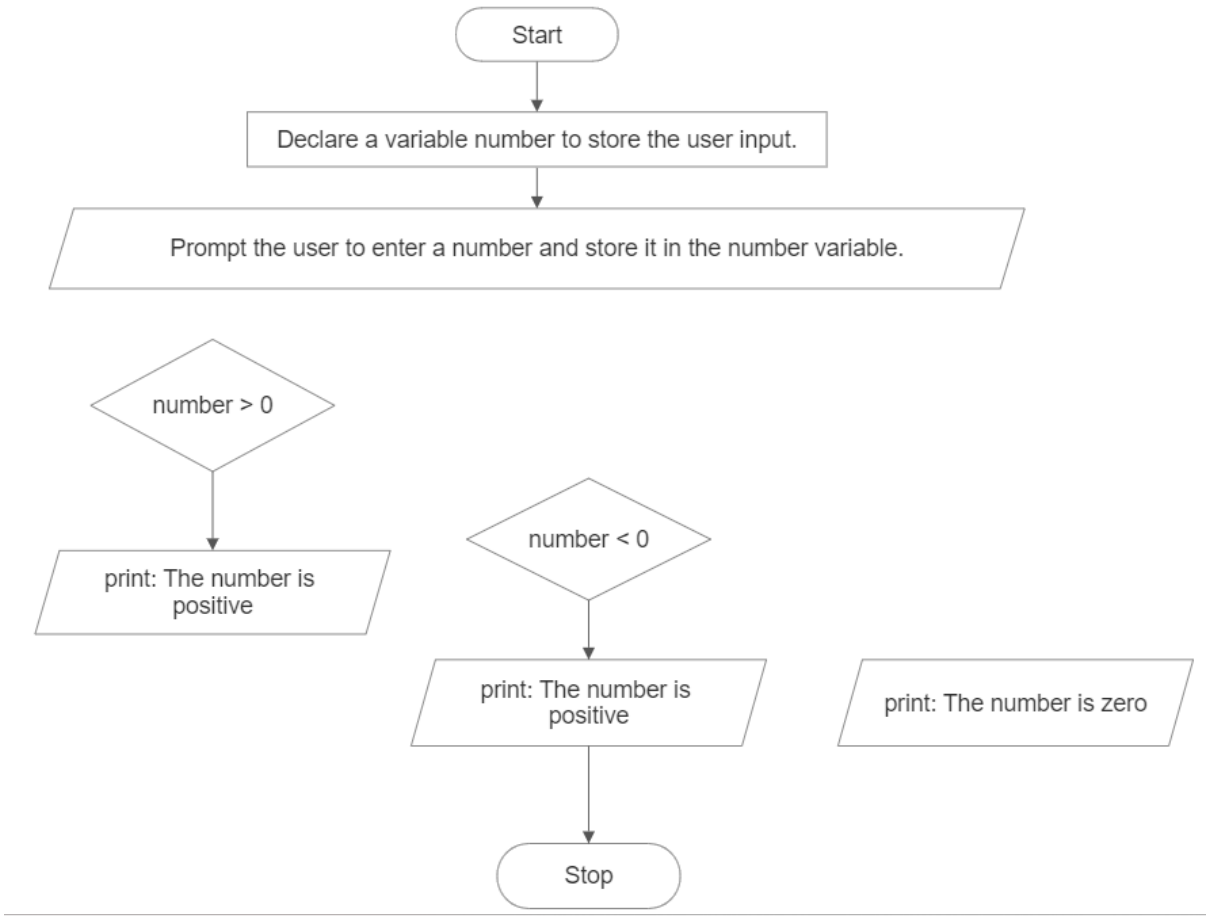
➢ **Objective**

The objective of this program is to write a code that takes a number as input from the user and determines whether the number is negative, positive, or zero. The program will use conditional statements to check the value of the number and display the appropriate message indicating its classification.

➢ **Algorithm**

i. Start.
ii. Declare a variable number to store the user input.
iii. Prompt the user to enter a number and store it in the number variable.
iv. Use conditional statements to check the value of the number:
v. If number is greater than 0, print "The number is positive."
vi. If number is less than 0, print "The number is negative."
vii. If number is equal to 0, print "The number is zero."
viii. Stop.

➢ **Flowchart**



➢ Code

```
#include <stdio.h>

int main( )
{
   int number;

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

   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;
}
```

➢ Output

```
Enter a number: 5
The number is positive.
```

➢ Discussion and Conclusion

This program takes a number as input from the user and determines whether the number is negative, positive, or zero using conditional statements. The user is prompted to enter a number, which is stored in the number variable. The program checks the value of the number using conditional statements (if, else if, and else) and prints the appropriate message indicating whether the number is positive, negative, or zero. The program was implemented using the VS Code IDE and compiled using GCC to generate an executable file. By using conditional statements, the program accurately classifies the input number based on its value.

2. WAP to find maximum between three numbers entered by the user.

➢ Objective

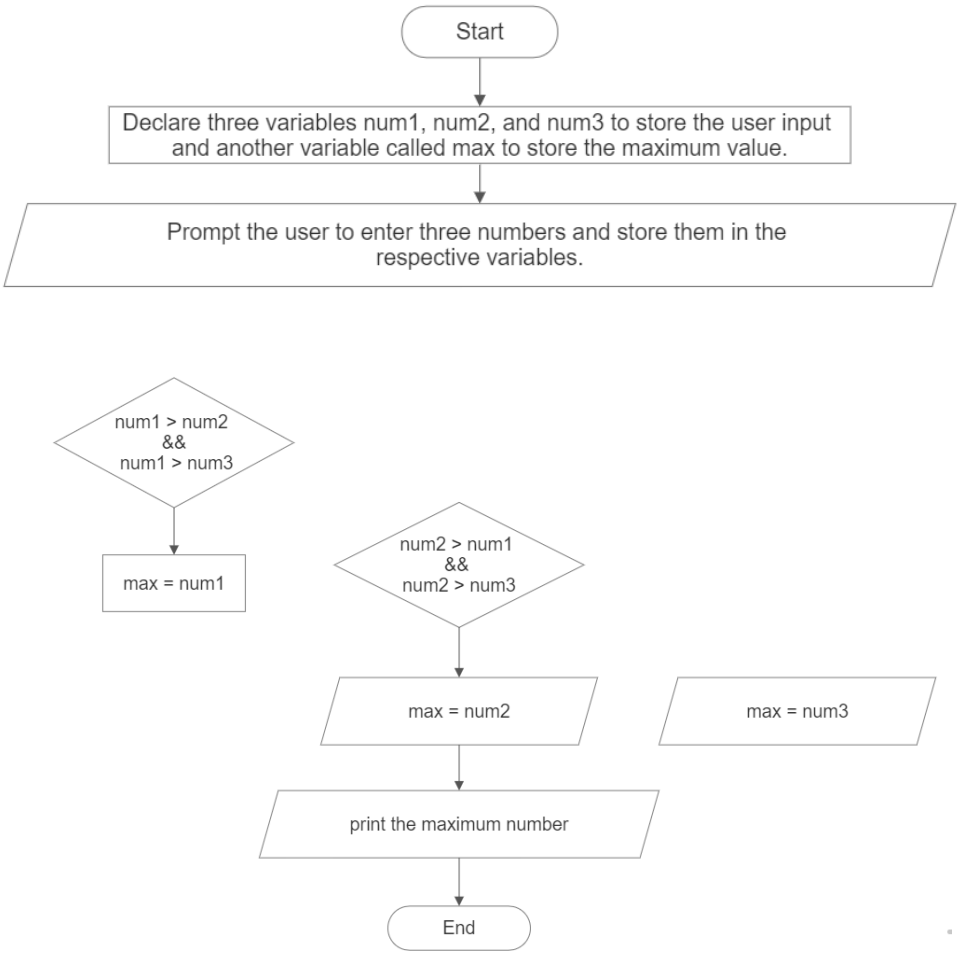
The objective of this program is to write a code that takes three numbers as input from the user and determines the maximum among them. The program will use conditional statements to compare the values of the three numbers and display the maximum value.

➢ Algorithm

i. Start.
ii. Declare three variables num1, num2, and num3 to store the user input and another variable called max to store maximum number.
iii. Prompt the user to enter three numbers and store them in the respective variables.
iv. Use conditional statements to compare the values of the three numbers:
- If num1 is greater than both num2 and num3, it is the maximum.
- If num2 is greater than both num1 and num3, it is the maximum.
- If num3 is greater than both num1 and num2, it is the maximum.

v. Print the maximum value.
vi. End the program.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main()
{
   int num1, num2, num3, max;

   printf("Enter three numbers: ");
   scanf("%d %d %d", &num1, &num2, &num3);

   if (num1 > num2 && num1 > num3)
      max = num1;
   else if (num2 > num1 && num2 > num3)
      max = num2;
   else
      max = num3;

   printf("The maximum number is: %d\n", max);

   return 0;
}
```

➢ Output

```
Enter three numbers: 1 2 2
The maximum number is: 2
```

➢ Discussion and Conclusion

This program takes three numbers as input from the user and determines the maximum among them using conditional statements. The user is prompted to enter three numbers, which are stored in the variables num1, num2, and num3. The program compares the values of these numbers using conditional statements (if and else if) and assigns the maximum value to the variable max. Finally, the program prints the maximum number. The program was implemented using the VS Code IDE and compiled using GCC to generate an executable file. By comparing the values of the three numbers, the program accurately determines the maximum value.

3. WAP to input a character from the user and check whether the character is vowel or consonant.

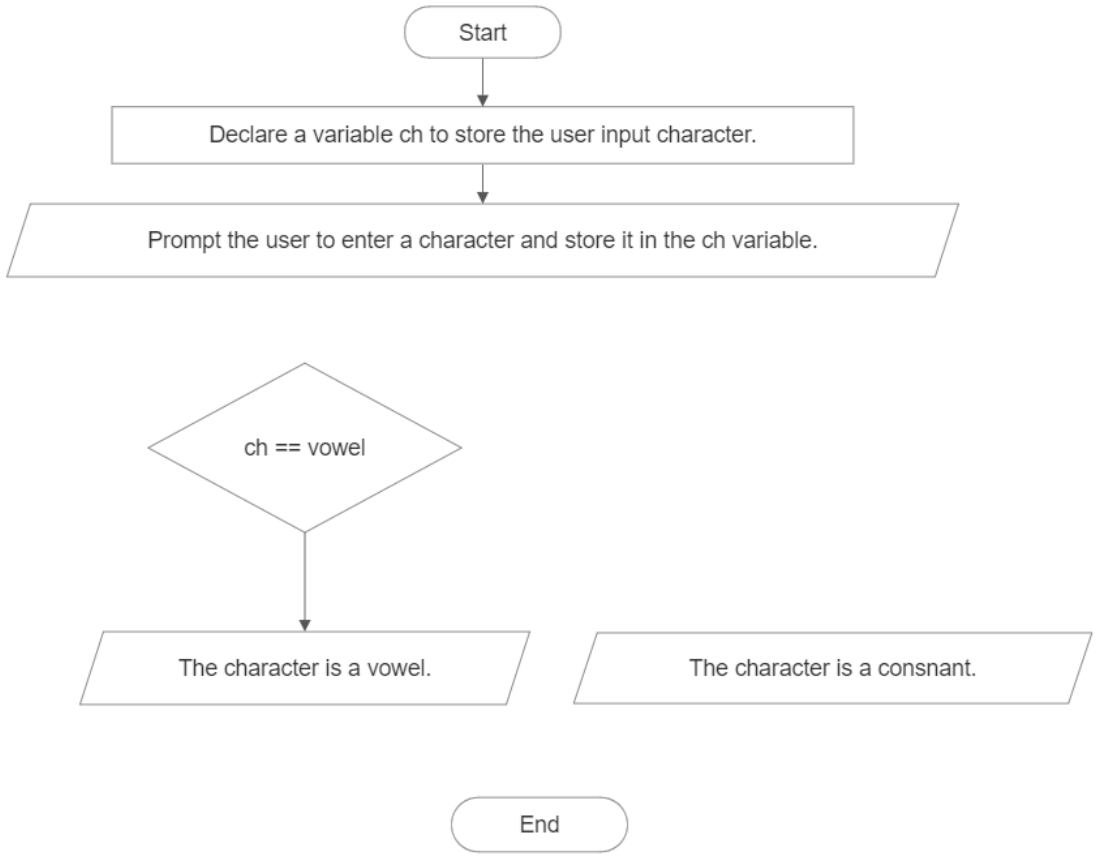
➢ Objective

The objective of this program is to write a code that takes a character as input from the user and determines whether the character is a vowel or a consonant.

➢ Algorithm

i. Start.
ii. Declare a variable ch to store the user input character.
iii. Prompt the user to enter a character and store it in the ch variable.
iv. Use conditional statements to check if the character is a vowel or a consonant:
   - If ch is equal to 'a', 'e', 'i', 'o', or 'u', it is a vowel.
v. If the character is a vowel, print a message indicating that it is a vowel.
vi. If the character is not a vowel, print a message indicating that it is a consonant.
vii. Stop.

➢ Flowchart



➢ Code

```
#include <stdio.h>

int main( )
{
   char ch;

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

   if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||
      ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U')
   {
      printf("The character is a vowel.\n");
   }
   else
   {
      printf("The character is a consonant.\n");
   }

   return 0;
}
```

➢ Output

```
Enter a character: r
The character is a consonant.
```

➢ Discussion

This program takes a character as input from the user and checks whether the character is a vowel or a consonant using conditional statements. The user is prompted to enter a character, which is stored in the variable ch. The program compares the value of ch with the vowels ('a', 'e', 'i', 'o', 'u') in both lowercase and uppercase forms to determine whether it is a vowel. If it is a vowel, the program prints a message indicating that it is a vowel. If it is not a vowel, the program considers it a consonant and prints a corresponding message. The program was implemented using the VS Code IDE and compiled using GCC to generate an executable file. By comparing the value of the character with the predefined vowels, the program accurately determines whether it is a vowel or a consonant.

4. WAP to input a character from the user and check whether the character is Alphabet or not. If the character is alphabet then show whether it is uppercase or lowercase

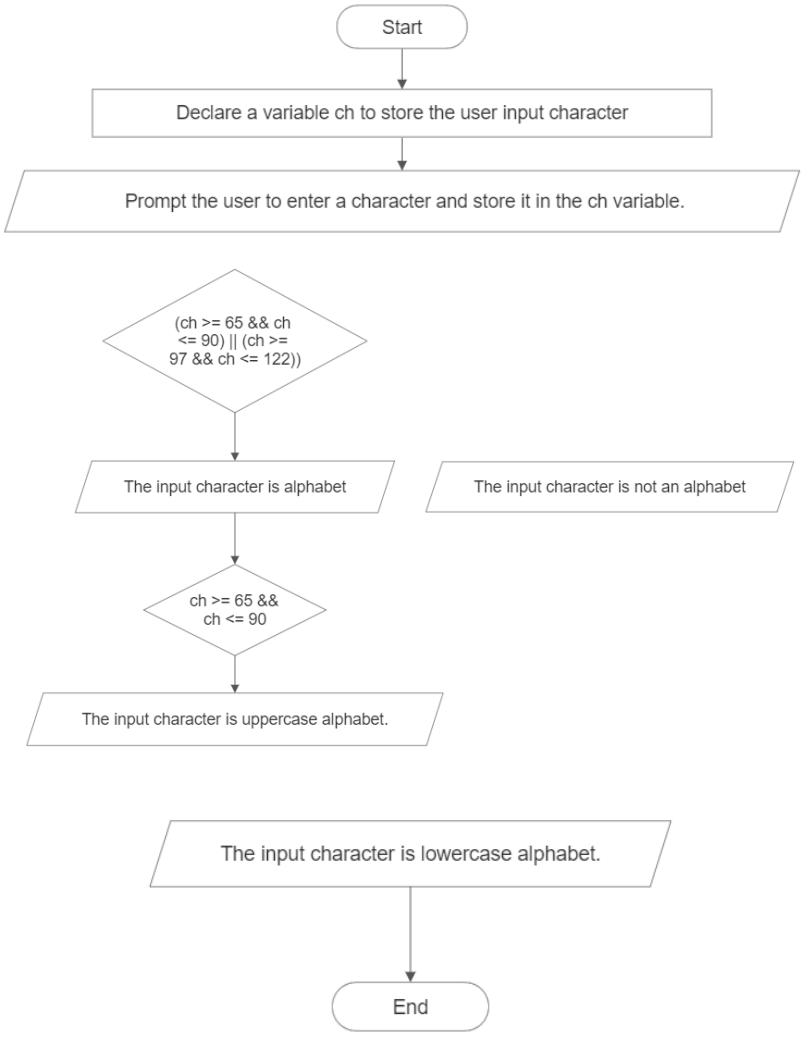
- **Objective**

  The objective of this program is to write a code that takes a character as input from the user and determines whether the character is an alphabet or not. If the character is an alphabet, the program will further identify whether it is in uppercase or lowercase.

- **Algorithm**

  i. . Start.
  ii. Declare a variable ch to store the user input character.
  iii. Prompt the user to enter a character and store it in the ch variable.
  iv. Use conditional statements to check if the character is an alphabet:
  v. If the ASCII value of ch is within the range of uppercase letters (65 to 90) or lowercase letters (97 to 122), it is an alphabet.
  vi. If the character is an alphabet, further check whether it is in uppercase or lowercase:
  vii. If the ASCII value of ch is within the range of uppercase letters (65 to 90), it is an uppercase letter and if the ASCII value of ch is within the range of lowercase letters (97 to 122), it is a lowercase letter.
  viii. Print the appropriate message indicating the result.
  ix. Stop.

- **Flowchart**



➢ Code

```
#include <stdio.h>

int main()
{
   char ch;

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

   if ((ch >= 65 && ch <= 90) || (ch >= 97 && ch <= 122))
   {
      printf("The character is an alphabet.\n");

      if (ch >= 65 && ch <= 90)
         printf("It is in uppercase.\n");
      else
         printf("It is in lowercase.\n");
   }
   else
   {
      printf("The character is not an alphabet.\n");
   }

   return 0;
}
```

➢ Output

```
Enter a character: e
The character is an alphabet.
It is in lowercase.
```

➢ Discussion and Conclusion

This program takes a character as input from the user and checks whether the character is an alphabet or not using conditional statements. The user is prompted to enter a character, which is stored in the variable ch. The program compares the ASCII value of the character to determine if it falls within the range of uppercase or lowercase letters. If it is an alphabet, the program prints the appropriate messages indicating that it is an alphabet and whether it is in uppercase or lowercase. If it is not an alphabet, the program notifies the user accordingly. The program was implemented using the VS Code IDE and compiled using GCC to generate an executable file. By checking the ASCII value of the character, the program accurately determines whether it is an alphabet and its case.

5. WAP to check whether the year entered by the user is leap year or not

➢ Objective

The objective of this program is to write a code that takes a year as input from the user and determines whether the year is a leap year or not.

➢ Algorithm

i. . Start.
ii. Declare a variable year to store the user input year.
iii. Prompt the user to enter a year and store it in the year variable.
iv. Use conditional statements to check if the year is a leap year:
- If the year is evenly divisible by 4, it is a potential leap year.
- If the year is also divisible by 100, it must be divisible by 400 to be considered a leap year.

v. If the year satisfies the leap year conditions, print a message indicating that it is a leap year.
vi. If the year does not satisfy the leap year conditions, print a message indicating that it is not a leap year.
vii. Stop.

➢ Flowchart

➢ Code

```
#include <stdio.h>

int main( ) {
   int year;

   printf("Enter a year: ");
   scanf("%d", &year);

   if (year % 4 == 0) {
      if (year % 100 == 0){
         if (year % 400 == 0){
            printf("%d is a leap year.\n", year);
         } else {
            printf("%d is not a leap year.\n", year);
         }
      } else {
         printf("%d is a leap year.\n", year);
      }
   } else {
      printf("%d is not a leap year.\n", year);
   }

   return 0;
}
```

➢ Output

```
Enter a year: 2022
2022 is not a leap year.
```

➢ Discussion and Conclusion

The program checks whether a year entered by the user is a leap year or not. It uses nested if statements to evaluate the leap year conditions. First, it checks if the year is divisible by 4. If it is, it further checks if the year is divisible by 100. If it is divisible by 100, it checks if it is also divisible by 400. If it satisfies all these conditions, it is considered a leap year. Otherwise, it is not a leap year. The program then displays the result accordingly. The program successfully determines whether a given year is a leap year or not using nested if statements. It prompts the user for a year and applies the leap year conditions to determine the result. The program was implemented using the VS Code IDE and compiled with GCC. By using the nested if statements, the program accurately identifies leap years based on the defined rules.