Tribhuwan University
Institute of Engineering
Purbanchal Campus, Dharan

C Lab Report

<u>Submitted By:</u>

Name: Anurag Dahal

Roll No: PUR079BCT010

<u>Submitted To:</u>

Department of Electronic and Computer Engineering

Lab Date:

Submission Date:

<u>Signature:</u>

Table of Contents

Lab Sheet 1

1. WAP to display hello world.

- **Code**

```
#include <stdio.h>

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

- **Output**

```
Hello, World!
```

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

- **Code**

```
#include <stdio.h>

int main( )
{
   printf("Name: Anurag Dahal\n");
   printf("Roll Number: PUR079BCT010\n");
   printf("Address: Birtamod\n");
   return 0;
}
```

- **Output**

```
Name: Anurag Dahal
Roll Number: PUR079BCT010
Address: Birtamod
```

3. WAP to add two integer variables and print sum

➢ 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
```

4. WAP to multiply two integer variables and print product

- **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
```

5. WAP to calculate and display the simple interest.

➢ 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
```

6. WAP to calculate the area of the circle

➢ 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.
```

➢ 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

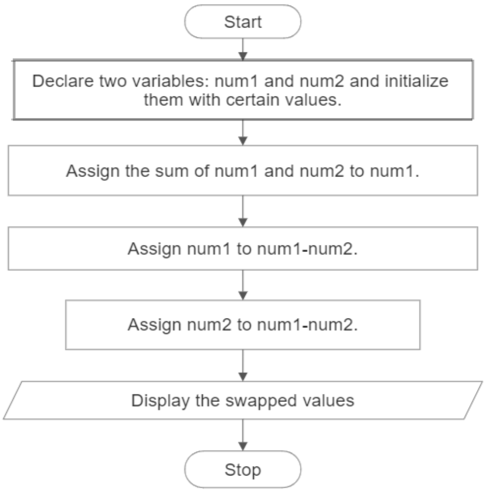
```
Before swapping:
num1 = 10
num2 = 20
After swapping (using third variable):
num1 = 20
num2 = 10
```

I. 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


Start
Declare two variables: num1 and num2 and initialize them with certain values.
Assign the sum of num1 and num2 to num1.
Assign num1 to num1-num2.
Assign num2 to num1-num2.
Display the swapped values
Stop

➢ 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.

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

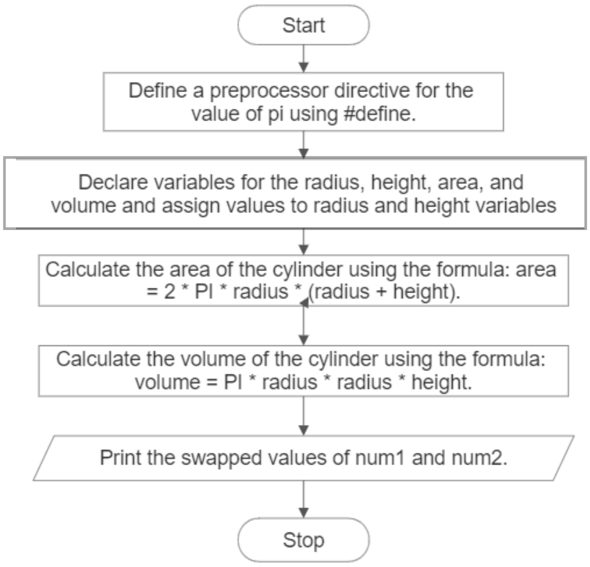
➢ Objective

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


Start
Define a preprocessor directive for the value of pi using #define.
Declare variables for the radius, height, area, and volume and assign values to radius and height variables
Calculate the area of the cylinder using the formula: area = 2 * PI * radius * (radius + height).
Calculate the volume of the cylinder using the formula: volume = PI * radius * radius * height.
Print the swapped values of num1 and num2.
Stop

➢ Code

```c
#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.

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

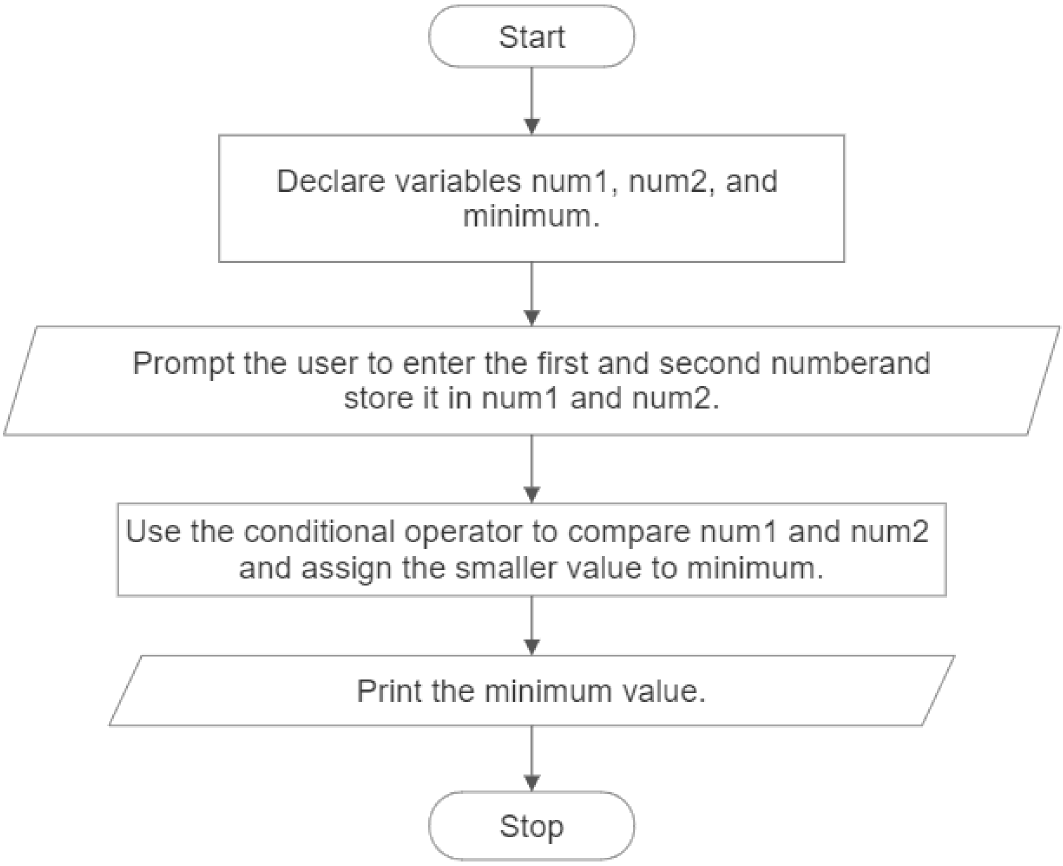
➢ Objective

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


Start
Declare variables num1, num2, and minimum.
Prompt the user to enter the first and second numberand store it in num1 and num2.
Use the conditional operator to compare num1 and num2 and assign the smaller value to minimum.
Print the minimum value.
Stop

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.

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

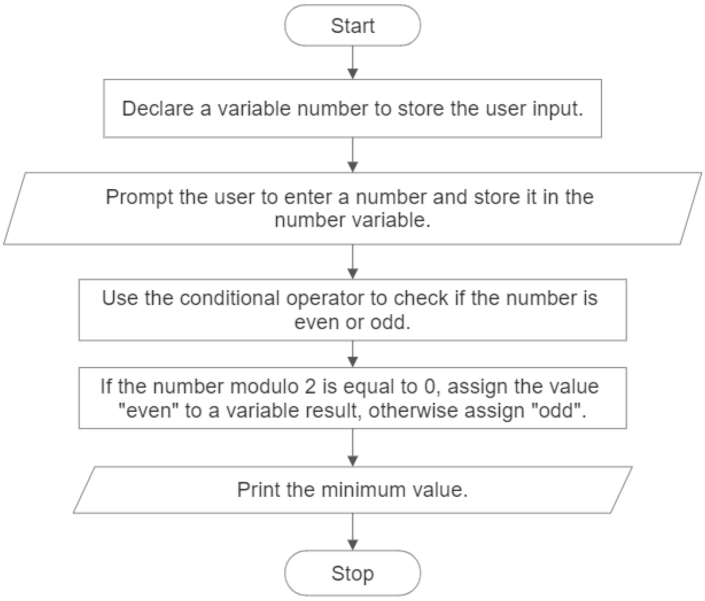
➢ Objective

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


Start
Declare a variable number to store the user input.
Prompt the user to enter a number and store it in the number variable.
Use the conditional operator to check if the number is even or odd.
If the number modulo 2 is equal to 0, assign the value "even" to a variable result, otherwise assign "odd".
Print the minimum value.
Stop

➢ 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.

4. 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.