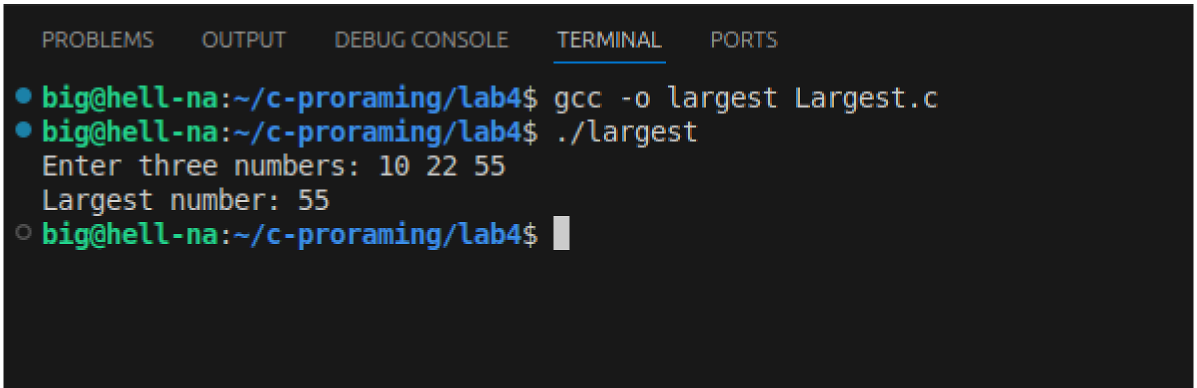
Lab – 4 (Control Statements)

1. Write a program to find the **largest among three numbers** using nested **if-else**.

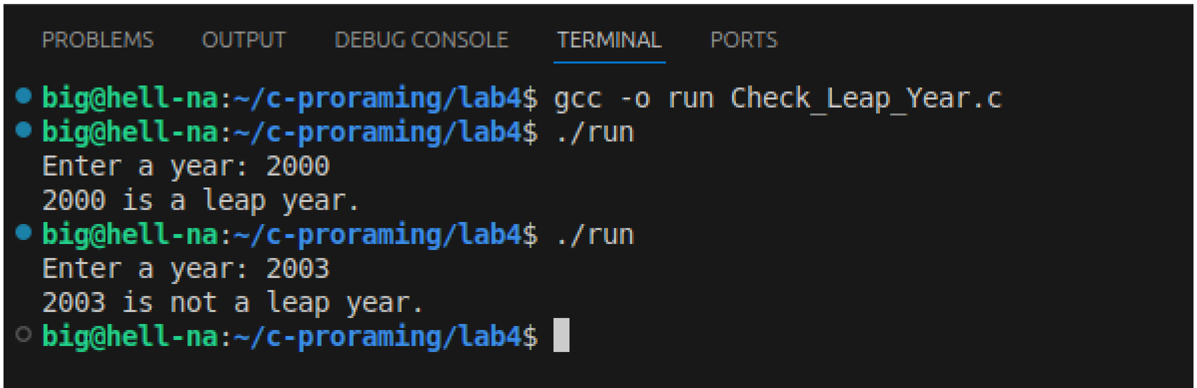
Output :



```
● big@hell-na:~/c-proraming/lab4$ gcc -o largest Largest.c
● big@hell-na:~/c-proraming/lab4$ ./largest
  Enter three numbers: 10 22 55
  Largest number: 55
○ big@hell-na:~/c-proraming/lab4$ 
```

2. Write a program to determine whether a **year is a leap year** using an **if-else** statement.

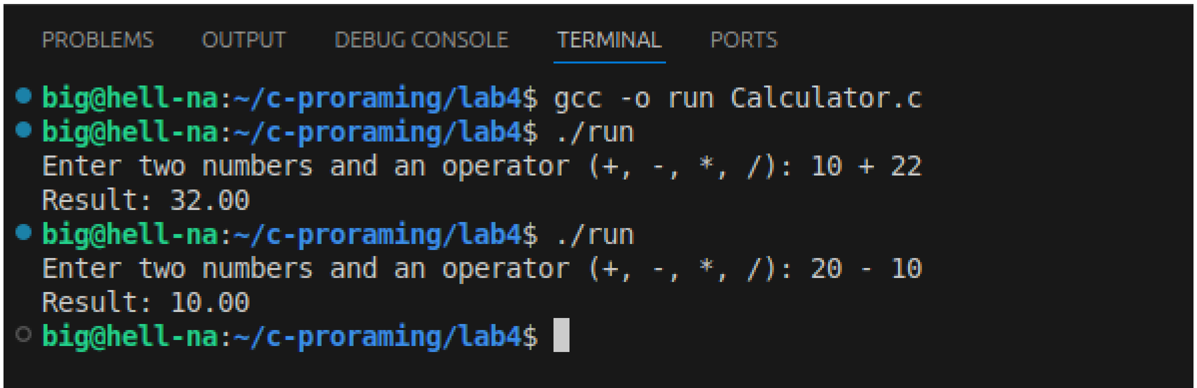
Output :



```
● big@hell-na:~/c-proraming/lab4$ gcc -o run Check_Leap_Year.c
● big@hell-na:~/c-proraming/lab4$ ./run
  Enter a year: 2000
  2000 is a leap year.
● big@hell-na:~/c-proraming/lab4$ ./run
  Enter a year: 2003
  2003 is not a leap year.
○ big@hell-na:~/c-proraming/lab4$ 
```

3. Create a simple **calculator** using the **switch-case** statement that takes two numbers and an operator (+, -, *, /) as input.

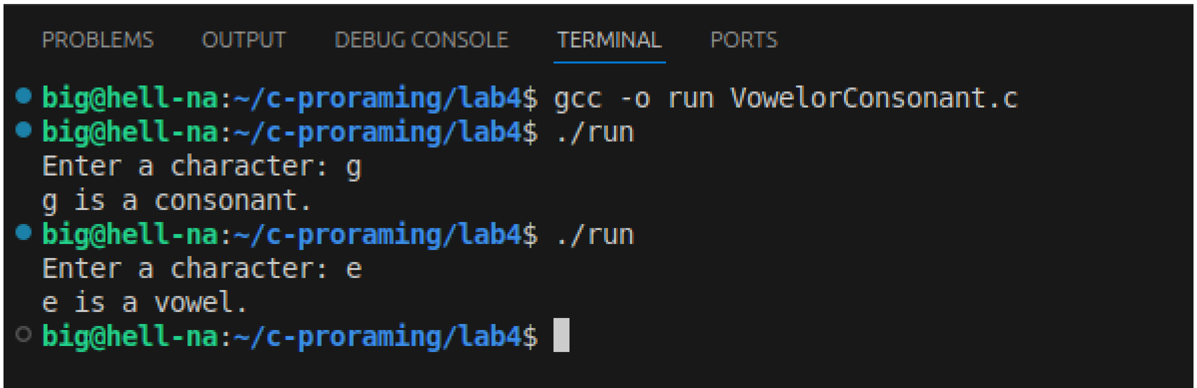
Output :



```
● big@hell-na:~/c-proraming/lab4$ gcc -o run Calculator.c
● big@hell-na:~/c-proraming/lab4$ ./run
  Enter two numbers and an operator (+, -, *, /): 10 + 22
  Result: 32.00
● big@hell-na:~/c-proraming/lab4$ ./run
  Enter two numbers and an operator (+, -, *, /): 20 - 10
  Result: 10.00
○ big@hell-na:~/c-proraming/lab4$ 
```

4. Write a program to determine whether a given **character is a vowel or consonant** using **switch-case**.

Output :

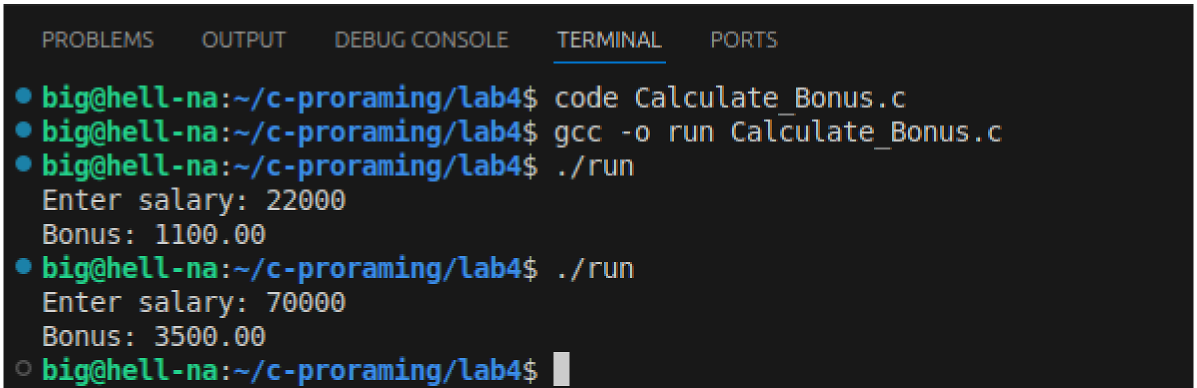


```
● big@hell-na:~/c-proraming/lab4$ gcc -o run VowelorConsonant.c
● big@hell-na:~/c-proraming/lab4$ ./run
  Enter a character: g
  g is a consonant.
● big@hell-na:~/c-proraming/lab4$ ./run
  Enter a character: e
  e is a vowel.
○ big@hell-na:~/c-proraming/lab4$ 
```

5. A company gives a bonus based on the salary:
   a. If salary < 10,000 → 10% bonus
   b. If salary >= 10,000 and < 20,000 → 8% bonus
   c. If salary >= 20,000 → 5% bonus

   Write a program to calculate the bonus based on user input.

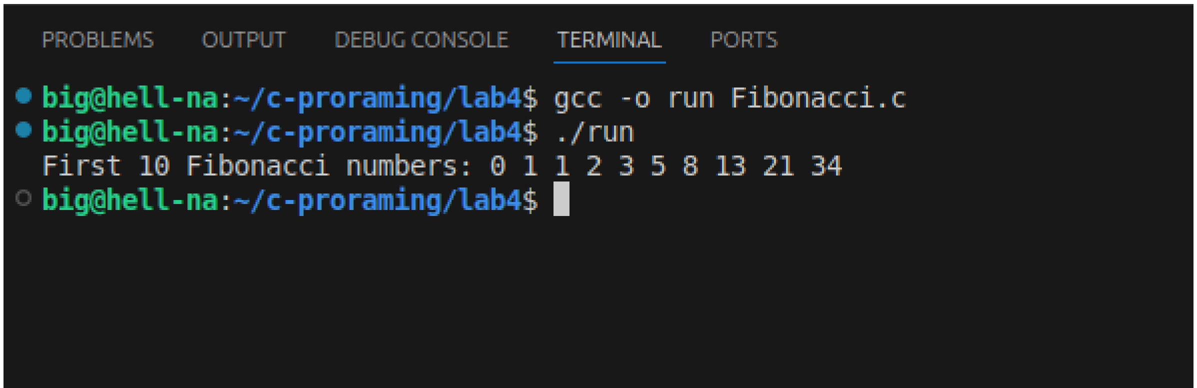
Output :



```
big@hell-na:~/c-proraming/lab4$ code Calculate_Bonus.c
big@hell-na:~/c-proraming/lab4$ gcc -o run Calculate_Bonus.c
big@hell-na:~/c-proraming/lab4$ ./run
Enter salary: 22000
Bonus: 1100.00
big@hell-na:~/c-proraming/lab4$ ./run
Enter salary: 70000
Bonus: 3500.00
big@hell-na:~/c-proraming/lab4$
```

6. Write a program to **print the first 10 Fibonacci numbers** using a **for loop**.

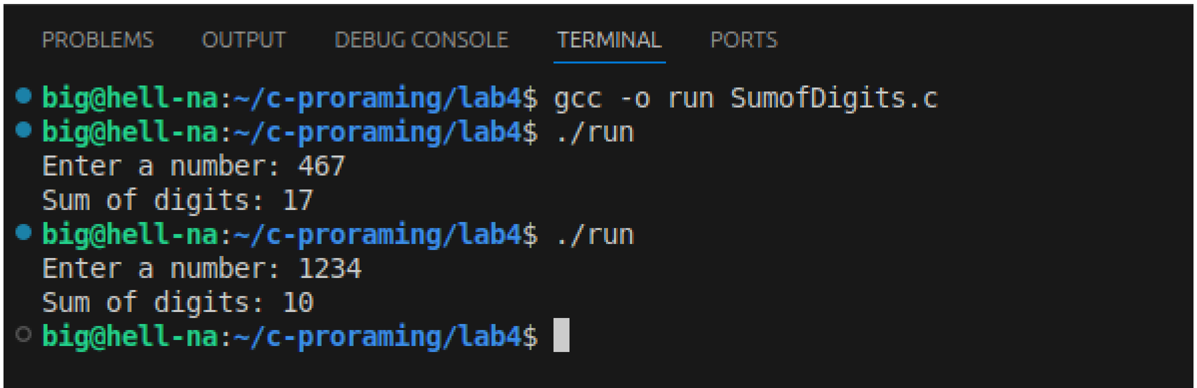
Output :



```
● big@hell-na:~/c-proraming/lab4$ gcc -o run Fibonacci.c
● big@hell-na:~/c-proraming/lab4$ ./run
  First 10 Fibonacci numbers: 0 1 1 2 3 5 8 13 21 34
○ big@hell-na:~/c-proraming/lab4$ 
```

7. Write a program to **find the sum of digits** of a given number using a **while loop**.

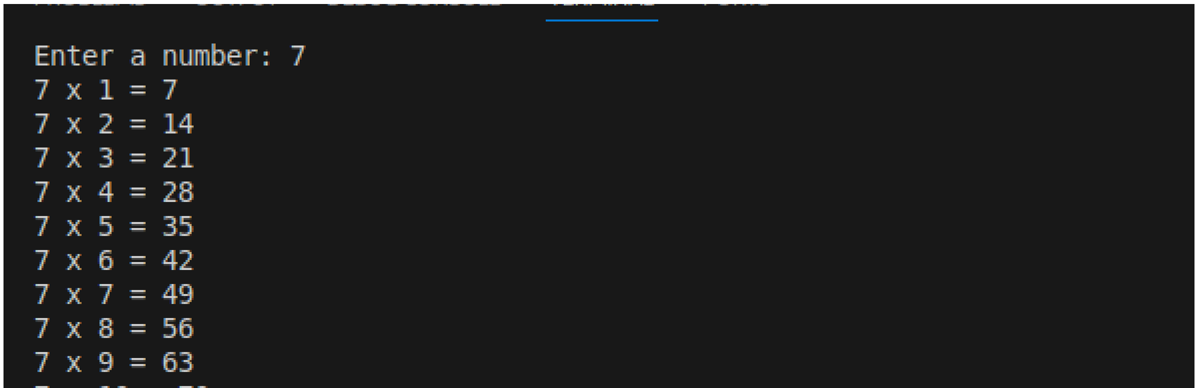
Output :



```
● big@hell-na:~/c-proraming/lab4$ gcc -o run SumofDigits.c
● big@hell-na:~/c-proraming/lab4$ ./run
  Enter a number: 467
  Sum of digits: 17
● big@hell-na:~/c-proraming/lab4$ ./run
  Enter a number: 1234
  Sum of digits: 10
○ big@hell-na:~/c-proraming/lab4$ 
```

8. Write a C program to **print the multiplication table of a given number** using a **do-while loop**.

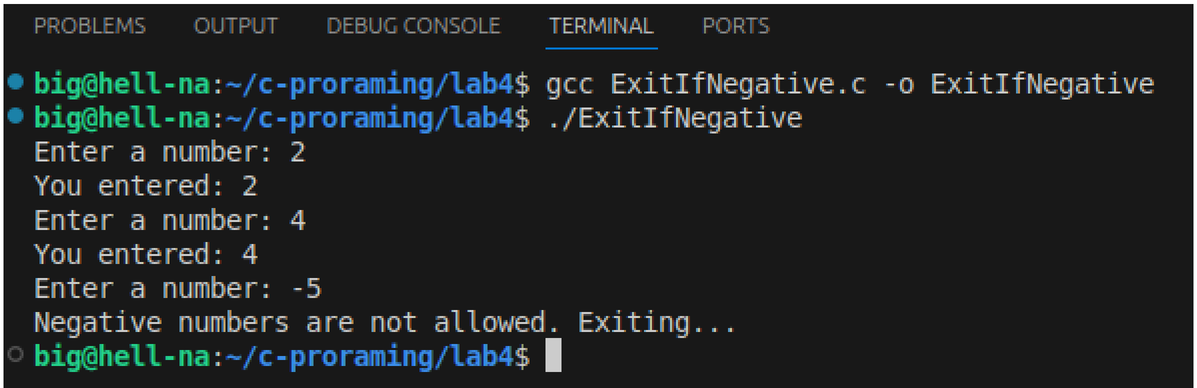
Output :



```
Enter a number: 7
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
```

9. Write a program to accept a number from the user. If the number is **negative**, display a message `"Negative numbers are not allowed"` and **exit** the program using the `exit()` function.

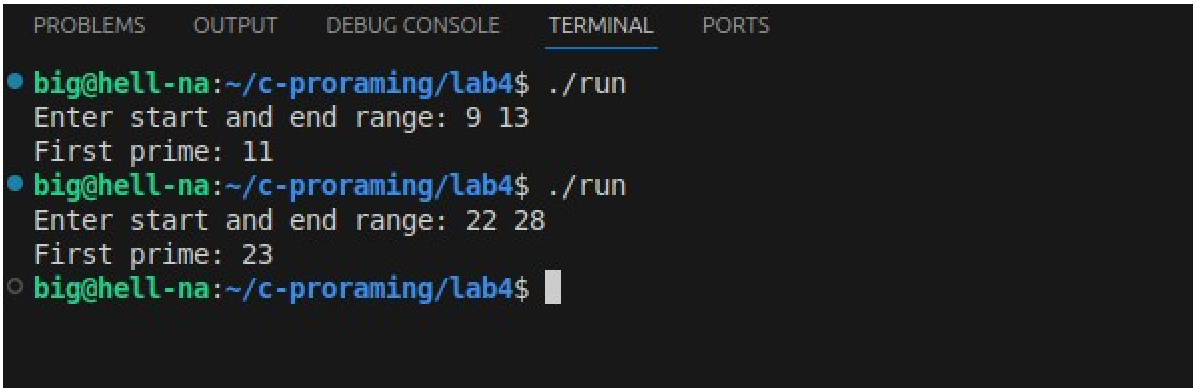
Output :



```
big@hell-na:~/c-proraming/lab4$ gcc ExitIfNegative.c -o ExitIfNegative
big@hell-na:~/c-proraming/lab4$ ./ExitIfNegative
Enter a number: 2
You entered: 2
Enter a number: 4
You entered: 4
Enter a number: -5
Negative numbers are not allowed. Exiting...
big@hell-na:~/c-proraming/lab4$ 
```

10. Write a C program to **find the first prime number** between a given ranges of numbers. If a prime number is found, stop searching using the **break** statement.

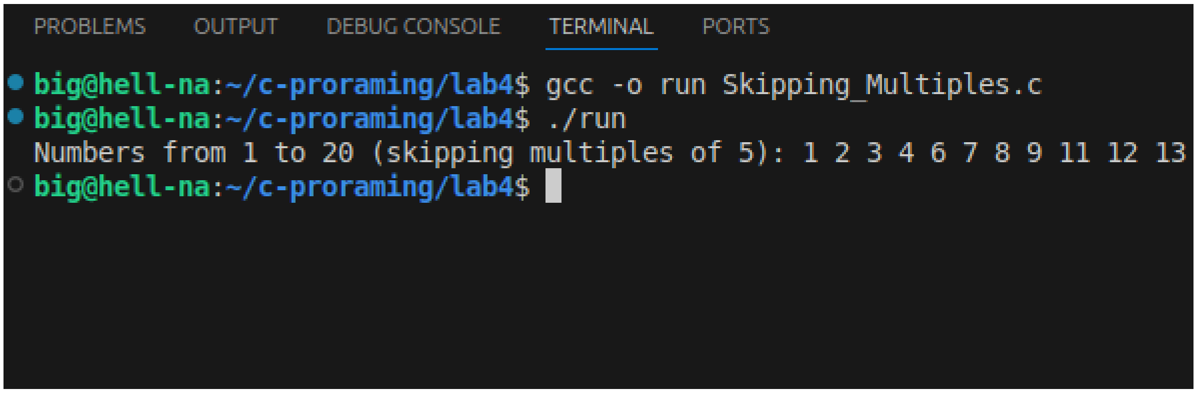
Output :



```
● big@hell-na:~/c-proraming/lab4$ ./run
  Enter start and end range: 9 13
  First prime: 11
● big@hell-na:~/c-proraming/lab4$ ./run
  Enter start and end range: 22 28
  First prime: 23
○ big@hell-na:~/c-proraming/lab4$
```

11.Write a C program to print numbers from **1 to 20**, but **skip numbers that are divisible by 5**, using the **continue** statement.

Output :



```
big@hell-na:~/c-proraming/lab4$ gcc -o run Skipping_Multiples.c
big@hell-na:~/c-proraming/lab4$ ./run
Numbers from 1 to 20 (skipping multiples of 5): 1 2 3 4 6 7 8 9 11 12 13
big@hell-na:~/c-proraming/lab4$
```