K. J. Somaiya College of Engineering, Mumbai-77

(A Constituent College of Somaiya Vidyavihar University)

Batch: C2-1	**Roll No.: 16010122104**
Experiment / assignment / tutorial No. 2	
Grade: AA / AB / BB / BC / CC / CD /DD	
Signature of the Staff In-charge with date	

TITLE: Write a program to accept 3 numbers from the user and find the largest of the 3 numbers using
If - else if-else
Ternary operator

AIM: Write a program to accept 3 numbers from the user and find the largest of the 3 numbers using
If - else if-else
Ternary operator

Expected OUTCOME of Experiment:

a. To run a program successfully and find the greatest of the given three numbers using if-else conditions
b. To run a program successfully and find the greatest of the given three numbers using ternary operator.

Books/ Journals/ Websites referred:

1. Programming in ANSI C, E. Balagurusamy, 7 th Edition, 2016, McGraw-Hill Education, India.
2. Structured Programming Approach, Pradeep Dey and Manas Ghosh, 1 st Edition, 2016, Oxford University Press, India.
3. Let Us C, Yashwant Kanetkar, 15th Edition, 2016, BPB Publications, India.

Problem Definition:

Ask user to input three numbers. Compare three numbers to find the largest of them using

1. Nested if else statement

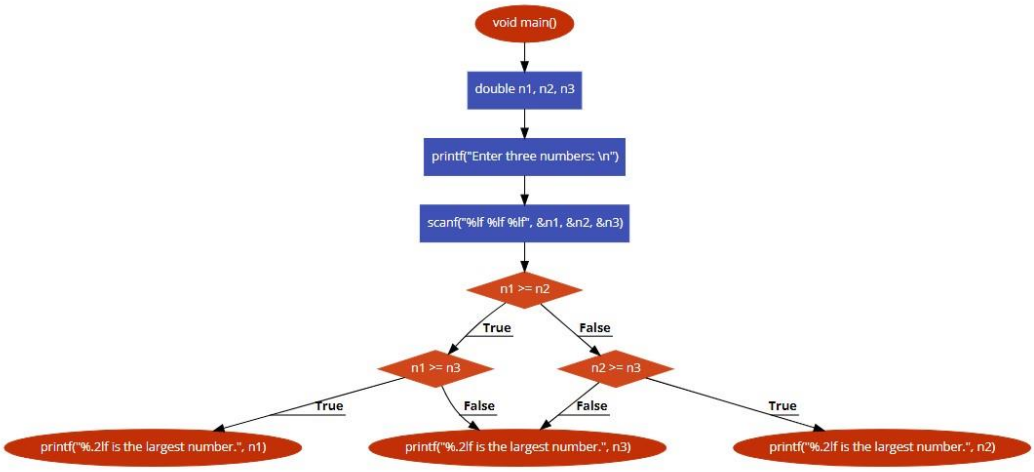
```
#include <stdio.h>

void main()
{

  double n1, n2, n3;

  printf("Enter three numbers: ");
  scanf("%lf %lf %lf", &n1, &n2, &n3);

  if (n1 >= n2)
    {

    if (n1 >= n3)
      printf("%.2lf is the largest number.", n1);
    else
      printf("%.2lf is the largest number.", n3);
    }

  else
    {

    if (n2 >= n3)
      printf("%.2lf is the largest number.", n2);
    else
      printf("%.2lf is the largest number.", n3);
    }
}
```

2. Using ternary operator

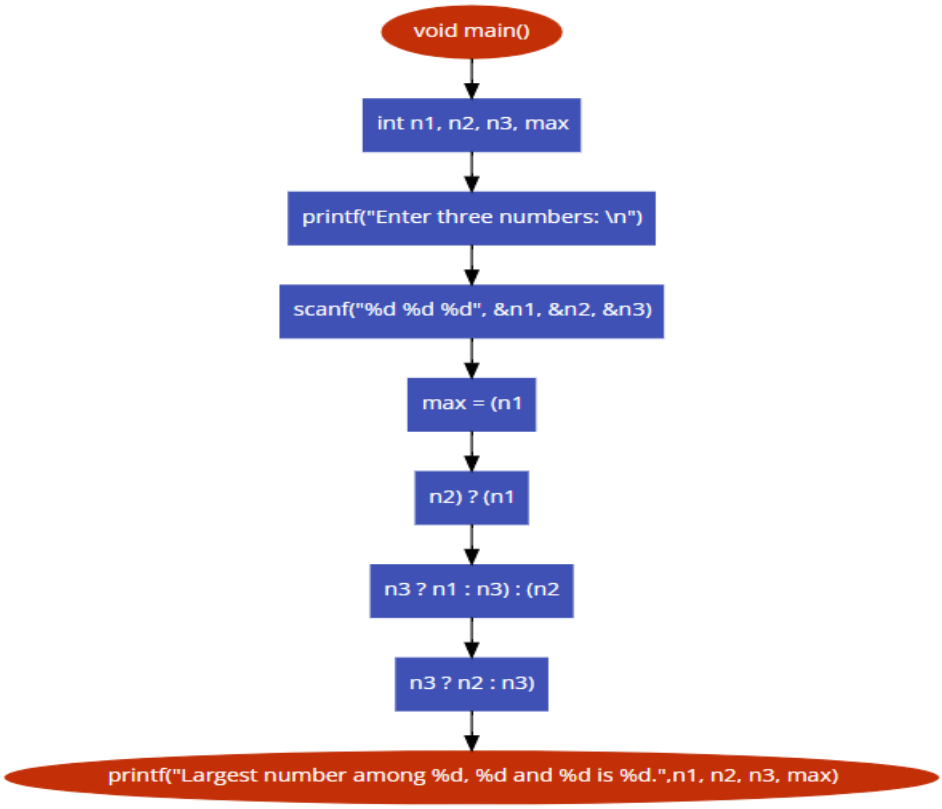
```
#include <stdio.h>

void main()
{
    int n1, n2, n3, max;

    printf("Enter three numbers: \n");
    scanf("%d %d %d", &n1, &n2, &n3);

    max = (n1 > n2) ? (n1 > n3 ? n1 : n3) : (n2 > n3 ? n2 : n3);

    printf("Largest number among %d, %d and %d is %d.",n1, n2, n3, max);
}
```

Flowchart:

a.



b.



Implementation details:

a.

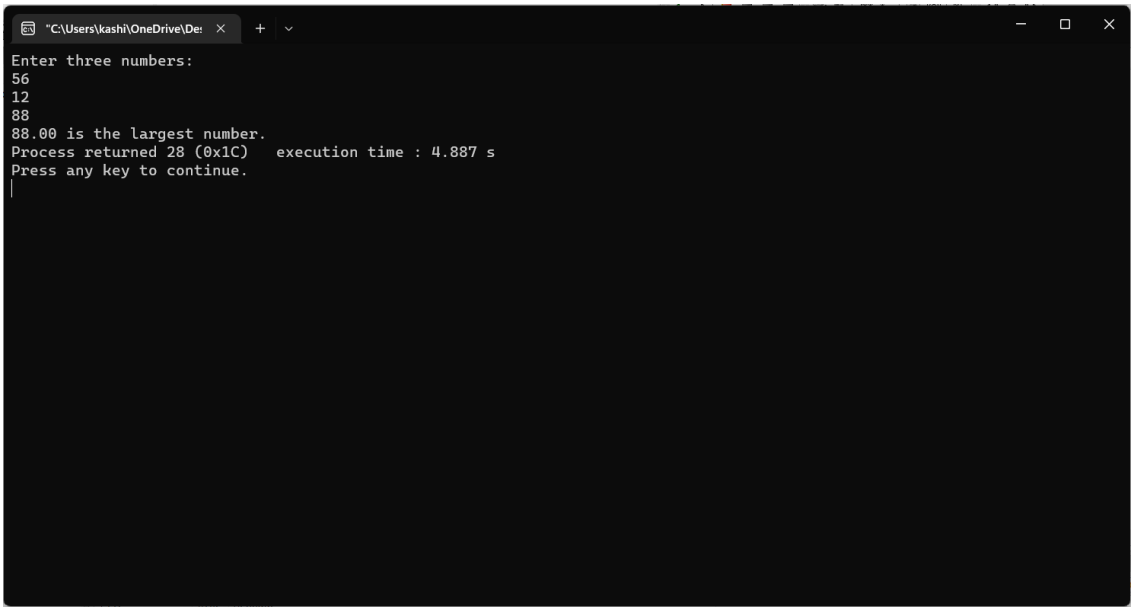
1) Start
2) Declare n1, n2, n3 as three numbers
3) Accept n1, n2, n3 from user
4) Check if n1>n2
5) Check if n1>n3
6) Check if n2>n3
7) Display greatest number
8) Stop

b.

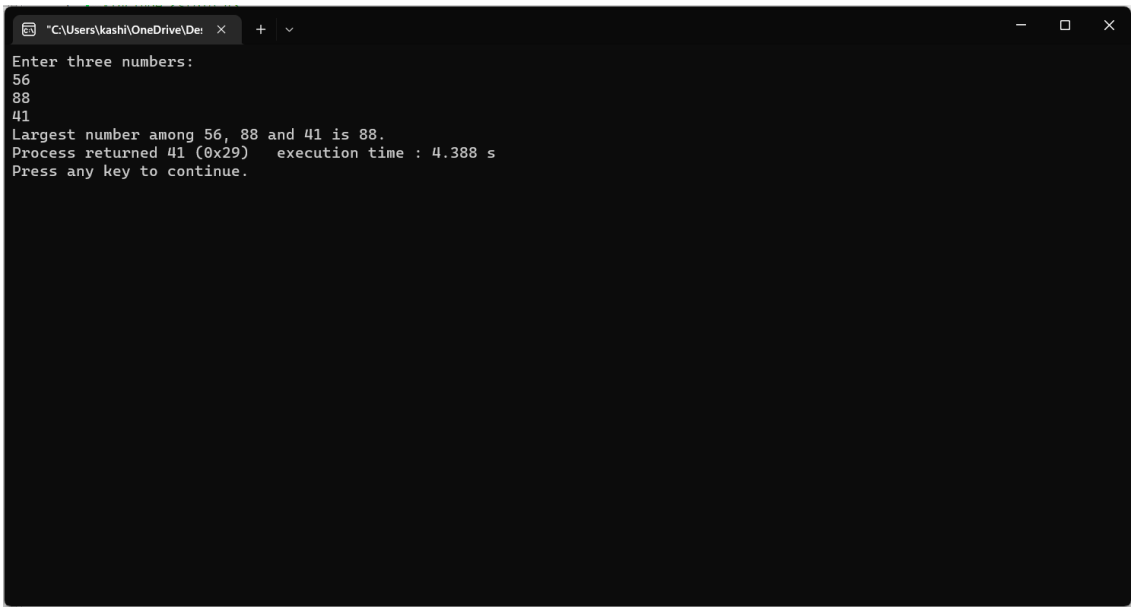
1) Start
2) Declare n1, n2, n3 as three numbers and max as greatest number
3) Accept n1, n2, n3 from user
4) max = (n1 > n2) ? (n1 > n3 ? n1 : n3) : (n2 > n3 ? n2 : n3);
5) display greatest number
6) Stop

Output(s):

a.



b.



Conclusion:

Through these programs, we learnt how to use nested if else statement as well as the usage of the conditional operator – ternary operator. We understood that for small programs like checking the largest among 3 numbers, ternary operator is a better choice as it reduces the amount of code. However, for larger programs, nested if else is better suited as we can include a lot more code in each if and else statements.

Post Lab Descriptive Questions

1. **Explain relational, logical and bitwise operators with examples.**

Ans: The operators which perform operation of relation between two operands are called relational operators. Eg: a<b

The operators which perform combine or negate the expressions that contain relational operators are called logical operators. Eg: &&

Bit manipulation operators manipulate individual bits within a variable. Bitwise operators modify variables considering the bit patterns that represent the values they store. Eg: ~

2. **Write associative rules and precedence table of various operators.**

Ans:

( )	Parentheses (function call) (see Note 1)	
[ ]	Brackets (array subscript)	
.	Member selection via object name	
->	Member selection via pointer	
++ – –	Postfix increment/decrement (see Note 2)	left-to-right
++ – –	Prefix increment/decrement	
+ –	Unary plus/minus	
! ~	Logical negation/bitwise complement	
(*type*)	Cast (convert value to temporary value of *type*)	
*	Dereference	
&	Address (of operand)	
sizeof	Determine size in bytes on this implementation	right-to-left
* / %	Multiplication/division/modulus	left-to-right
+ –	Addition/subtraction	left-to-right
<< >>	Bitwise shift left, Bitwise shift right	left-to-right
< <=	Relational less than/less than or equal to	
> >=	Relational greater than/greater than or equal to	left-to-right
== !=	Relational is equal to/is not equal to	left-to-right
&	Bitwise AND	left-to-right
^	Bitwise exclusive OR	left-to-right
|	Bitwise inclusive OR	left-to-right
&&	Logical AND	left-to-right
| |	Logical OR	left-to-right
? :	Ternary conditional	right-to-left
=	Assignment	
+= -=	Addition/subtraction assignment	
*= /=	Multiplication/division assignment	
%= &=	Modulus/bitwise AND assignment	
^= |=	Bitwise exclusive/inclusive OR assignment	
<<= >>=	Bitwise shift left/right assignment	right-to-left
,	Comma (separate expressions)	left-to-right

Date: 07/01/2023 **Signature of faculty in-charge**