**Batch: C3-3 Roll No.: 16010122221**

**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:**

1.By performing this experiments we will be able to use if else operators along with their shortcuts.

2. From this experiments we will be able to use ternary operators such as ? AND : **\_**

**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, 1st 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
2. Using ternary operator

**Algorithm:**

**Implementation details:**

**PROBLEM-1**

#include <stdio.h>

int main(void)

{

int a,b,c;

printf("Enter the numbers ");

scanf("%d %d %d",&a,&b,&c); // Enter numbers

if(a>b)

{

if(a > c )

{ printf(" Highest number is a = %d",a);}

else

{ printf(" Highest number is c = %d",c);}

}

else

{

if(b > c )

{ printf( "highest number is b = %d",b);}

else

{printf(" highest number is c= %d",c);}

}

return 0;

}

**PROBLEM-2**

#include <stdio.h>

int main() {

   float p,q,r;

printf("enter the value of first number");

scanf("%f",&p);

printf("enter the value of second number");

scanf("%f",&q);

printf("enter the value of third number");

scanf("%f",&r);

float greatest = p>q?(p>r ?p:r):(q>r?q:r);

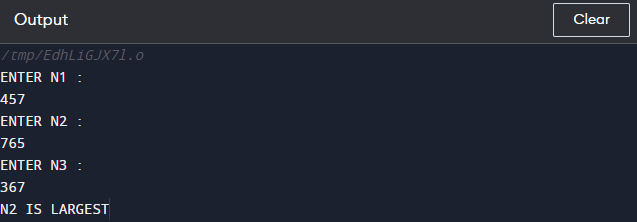
printf("the greatest value is=%f",greatest);

    return 0;

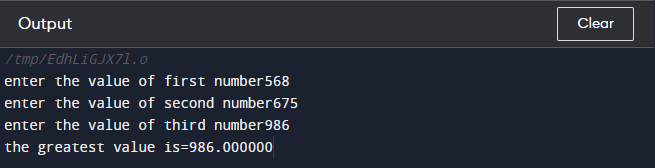
}

**Output(s):**

**PROBLEM-1**



**PROBLEM-2**



**Conclusion:**

GAINED THE KNOWLEDGE AND ABILITY TO USE IF AND IF ELSE

OPERATORS ALONGWITH THEIR SHORTCUTS WHICH ARE TERNARY

OPERATORS SUCH AS ? AND : ALONGWITH BEING ABLE TO WRITE A

PROGRAMME TO DIFFERENTIATE BETWEEN 3 INTEGERS TO FIND OUT

THE ONE WITH THE GREATEST VALUE AMONG THEM.

**Post Lab Descriptive Questions**

1. **Explain bitwise operators with examples**

A bitwise operator is an operator used to perform bitwise operations on bit patterns or binary numerals that involve the manipulation of individual bits.

Some examples are :-

* OR (|)
* XOR (^)
* NOT (~)
* >> (Right-Shift)
* << (Left-Shift)

1. **Write associative rules and precedence table of various operators.**
2. Associativity is only used when there are two or more operators of same precedence.
3. All operators with the same precedence have same associativity.
4. Precedence and associativity of postfix ++ and prefix ++ are different.
5. Comma has the least precedence among all operators and should be used carefully.
6. There is no chaining of comparison operators in C.

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Associativity** |
| ( )  [ ]  .  ->  ++ –  – | 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 |
| ++ –  –  + –  ! ~  (*type*)  \*  &  size of | Prefix increment/decrement  Unary plus/minus  Logical negation/bitwise complement  Cast (convert value to temporary value of *type*)  Dereference  Address (of operand)  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: \_\_\_\_\_\_\_\_\_\_\_\_\_ Signature of faculty in-charge**