**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **28-05-2020** | **Name:** | **Karthik J** |
| **Course:** | **Logic Design** | **USN:** | **4AL16EC030** |
| **Topic:** | Boolean algebra  MUX  BCD Counter | **Semester & Section:** | **8TH A** |
| **GitHub Repository:** | Karthik-J |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session**       **Boolean Algebra**  * **Boolean algebra,** symbolic system of [mathematical logic](https://www.britannica.com/topic/formal-logic) that represents relationships between entities—either ideas or objects. * The basic rules of this system were formulated in 1847 by [George Boole](https://www.britannica.com/biography/George-Boole) of England and were subsequently refined by other mathematicians and applied to [set theory](https://www.britannica.com/science/set-theory). * Furthermore, it [constitutes](https://www.merriam-webster.com/dictionary/constitutes) the basis for the design of circuits used in electronic [digital computers](https://www.britannica.com/technology/digital-computer). * The ordinary [algebra](https://www.britannica.com/science/algebra) in which the elements are the real numbers and the commutative binary operations are addition and multiplication does not satisfy all the requirements of a Boolean algebra. * The set of real numbers is closed under the two operations that is, the sum or the product of two real numbers also is a real number identity elements exist—0 for addition and 1 for multiplication that is, a + 0 = a and a × 1 = a for any [real number](https://www.britannica.com/science/real-number) a and multiplication is distributive over addition (that is, a × [b + c] = [a × b] + [a × c]); but addition is not distributive over multiplication (that is, a + [b × c] does not, in general, equal [a + b] × [a + c]). * The advantage of Boolean algebra is that it is valid when [truth-values](https://www.britannica.com/topic/truth-value)—i.e., the truth or falsity of a given proposition or logical statement are used as variables instead of the numeric quantities employed by ordinary algebra. * It lends itself to manipulating propositions that are either true with [truth-value](https://www.britannica.com/topic/truth-value) 1 or false with truth-value 0). Two such propositions can be combined to form a [compound](https://www.merriam-webster.com/dictionary/compound) proposition by use of the logical connectives, or operators, AND or OR. * The standard symbols for these connectives are ∧ and ∨, respectively. The truth-value of the resulting proposition is dependent on the truth-values of the components and the [connective](https://www.britannica.com/topic/connective-logic) employed. For example, the propositions a and b may be true or false, independently of one another. * The connective AND produces a proposition, a ∧ b, that is true when both a and b are true, and false otherwise.  **Boolean Laws** There are six types of Boolean Laws. Commutative law Any binary operation which satisfies the following expression is referred to as commutative operation.  Commutative Law  Commutative law states that changing the sequence of the variables does not have any effect on the output of a logic circuit. Associative law This law states that the order in which the logic operations are performed is irrelevant as their effect is the same.  Associative Law Distributive law Distributive law states the following condition.  Distributive Law AND law These laws use the AND operation. Therefore they are called as **AND** laws.  AND Law OR law These laws use the OR operation. Therefore they are called as **OR** laws.  OR Law INVERSION law This law uses the NOT operation. The inversion law states that double inversion of a variable results in the original variable itself.  NOT Law **BCD to 7 Segment Display** A seven-segment display is an electronic display device for displaying decimal numerals. Seven-segment displays are widely used in digital clocks, electronic meters and other electronic devices that display numerical information. **7 Segment Display** A 7 Segment LED display generally has 8 input connections, one for each LED segment and one that acts as a common terminal. There are 2 types of 7 Segment LED digital display.   * Common Cathode Display – all the cathode connections of the LEDs are connected to ground. A logic '1' applied to the anode terminal of the individual segment illuminates it. * Common Anode Display – all the anode connections of the LEDs are connected to VCC. A logic '0' applied to the cathode terminal of the individual segment illuminates it. |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Date:** | | **28-05-2020** | **Name:** | **Karthik J** |  |
| **Course:** | | [Programming with Python: Hands-on Introduction for Beginners](https://www.udemy.com/course/python-programming-beginners/) | **USN:** | **4AL16EC030** |  |
| **Topic:** | |  | **Semester & Section:** | **8th A** |  |
|  | **AFTERNOON SESSION DETAILS** | | | | |
|  | **Image of session** | | | | |
|  | Python - Tuples A tuple is an immutable sequence of Python objects. Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.  Creating a tuple is as simple as putting different comma-separated values. Optionally you can put these comma-separated values between parentheses also.  For example −  tup1 = ('physics', 'chemistry', 1997, 2000);  tup2 = (1, 2, 3, 4, 5 );  tup3 = "a", "b", "c", "d"; Accessing Values in Tuples To access values in tuple, use the square brackets for slicing along with the index or indices to obtain value available at that index.  For example −  tup1 = ('physics', 'chemistry', 1997, 2000);  tup2 = (1, 2, 3, 4, 5, 6, 7 );  print "tup1[0]: ", tup1[0];  print "tup2[1:5]: ", tup2[1:5]; Updating Tuples Tuples are immutable which means you cannot update or change the values of tuple elements. You are able to take portions of existing tuples to create new tuples  Example  tup1 = (12, 34.56);  tup2 = ('abc', 'xyz');  # Following action is not valid for tuples  # tup1[0] = 100;  # So, let's create a new tuple as follows  tup3 = tup1 + tup2;  print tup3; Delete Tuple Elements Removing individual tuple elements is not possible. There is, of course, nothing wrong with putting together another tuple with the undesired elements discarded.  To explicitly remove an entire tuple, just use the **del** statement.  example −  tup = ('physics', 'chemistry', 1997, 2000);  print tup;  del tup;  print "After deleting tup : ";  print tup;  This produces the following result. Note an exception raised, this is because after **del tup** tuple does not exist any-more − Basic Tuples Operations Tuples respond to the + and \* operators much like strings; they mean concatenation and repetition here too, except that the result is a new tuple, not a string. Indexing, Slicing, and Matrixes Because tuples are sequences, indexing and slicing work the same way for tuples as they do for strings. Assuming following input −  L = ('spam', 'Spam', 'SPAM!') Python Lists The list is a most versatile datatype available in Python which can be written as a list of comma-separated values (items) between square brackets. Important thing about a list is that items in a list need not be of the same type.  Creating a list is as simple as putting different comma-separated values between square brackets. For example −  list1 = ['physics', 'chemistry', 1997, 2000];  list2 = [1, 2, 3, 4, 5 ];  list3 = ["a", "b", "c", "d"]  Similar to string indices, list indices start at 0, and lists can be sliced, concatenated and so on Accessing Values in Lists To access values in lists, use the square brackets for slicing along with the index or indices to obtain value available at that index.  **Example**  list1 = ['physics', 'chemistry', 1997, 2000];  list2 = [1, 2, 3, 4, 5, 6, 7 ];  print "list1[0]: ", list1[0]  print "list2[1:5]: ", list2[1:5] Updating Lists You can update single or multiple elements of lists by giving the slice on the left-hand side of the assignment operator, and you can add to elements in a list with the append() method.  **Example**  list = ['physics', 'chemistry', 1997, 2000];  print "Value available at index 2 : "  print list[2]  list[2] = 2001;  print "New value available at index 2 : "  print list[2] | | | | |