|  |
| --- |
| **Part A**  **Name:-**  **Roll No:-**  **Subject:- Data Structures and Algorithms**  **Program: B Tech/MBA Tech CE 2nd Year** |
| **Aim:**  To study and implement concept of Stack data structure |
| **Prerequisite:** C++ Programming |
| **Outcome:** To implement parenthesis matching using stack. |
| * **Theory:** Stack is a linear data structure which stores the elements in an ordered manner. * The elements in a stack are added and removed only from one end which is called top. * The policy is LIFO, the element that was inserted last is the first one to be taken out.   Operations-   1. Push(element)- inserts an element at top of the stack 2. Pop() – removes the topmost element 3. Peek()- returns the topmost element without removing 4. isEmpty() – checks if stack is empty 5. isFull()- checks if stack is full   **Parenthesis matching using stack:**  Stacks can be used to check the validity of parentheses in any algebraic expression.  Algebraic expression is valid if for every open bracket there is a corresponding closing bracket.  For example, the expression (A+B} is invalid but an expression {A + (B – C)} is valid.  Algorithm  Parenthesischeck( string)  {  valid= true;  s= the empty stack;  while ( entire string is not read)  {  symb=next symbol of the string;  if (symb == ‘(‘ || symb == ‘{‘|| symb == ‘[‘)  push (symb)  else if (symb == ‘)‘ || symb == ‘}‘|| symb == ‘]‘)  {  i= pop ();  if ( i is not the matching opener of symb)  valid=false  } *//end else*  } *//end while*  if (stack s not empty)  valid=false  if (valid) print (“the string is valid”);  else print (“the string is not valid”);  } |
| **Procedure:**   1. Open CodeBlock editor or visual studio editor and write the code in C++. 2. Complile and run the code |
| **Instructions:**   1. Copy code & paste in code section and output of Part B. |
|  |

|  |
| --- |
| **Part B** |
| **Code:** |
| **Output:** |
| **Observation & Learning:**  Write your Observations & Learning after performing task |
| **Answer Following Question**  Simulate the parenthesis checker algorithm for each of the following strings by showing the contents of the stack at each point-   1. {A + (B – C)} 2. (A + B)) \* C |
| **Conclusion:**  We successfully implemented C++ program for parenthesis check using stack |