|  |
| --- |
| **Part A** |
| **Aim:** Program to calculate of Follow of a grammar. |
| **Prerequisite:** C |
| **Outcome:** Calculation of Follow. |
| **Procedure:**   1. The study should be in the given format |
| **Instructions:**   1. Use string array to store the strings. |
|  |

|  |
| --- |
| **Part B** |
| **Study:**  **package** edu.ashish.PCD.FirstAndFollow;  **import** sun.reflect.Reflection;  **import** java.util.\*;  */\*\*  \* Created by admin on 18-Feb-16.  \*/* **public class** NonTerminal {   String **value**;   **public** NonTerminal(String value) {  **this**.**value** = value;  }   **public** String getValue() {  **return value**;  }     **public void** setValue(String value) {  **this**.**value** = value;  }       **public** HashSet<Terminal> findFirst() {   NonTerminal targetNT = **this**;  HashSet<Terminal> set = **new** HashSet<>();   LinkedHashMap<NonTerminal , ArrayList<Object>> rules = **new** ProductionRules().makeProductionTable();   ArrayList<Object> RHS = rules.get(targetNT);    **int** index=0;  Object alphabet = **""**;  **while**(alphabet != **null**)  {  alphabet =RHS.get(index);  **if**(alphabet **instanceof** NonTerminal)  {  set = ((NonTerminal) alphabet).findFirst();  **return** set;  }  **else if** (alphabet **instanceof** Terminal)  {  set.add((Terminal) alphabet);  **return** set;  }   index = index+1;  }        **return** set ;  }    **public** HashSet<Terminal> findFollow( NonTerminal givenNT)  {  HashSet<Terminal> toReturn = **new** HashSet<Terminal>();  HashSet<NonTerminal> foundKeys = findInRHS(givenNT);   LinkedHashMap<NonTerminal , ArrayList<Object>> rules = **new** ProductionRules().makeProductionTable();    **for** (NonTerminal currentKey : foundKeys) {  ArrayList<Object> RHSOfFoundNTKey = rules.get(currentKey);    **if**(isLastIn(RHSOfFoundNTKey , givenNT ))  {  HashSet<Terminal> followsFound = findFollow(currentKey);   **for** (Terminal terminal : followsFound) {  toReturn.add(terminal);  }  }   **else** {  **int** nextIndex = RHSOfFoundNTKey.indexOf(givenNT) + 1;  Object nextObject = RHSOfFoundNTKey.get(nextIndex);   **if** (nextObject **instanceof** Terminal)  {  toReturn.add((Terminal) nextObject);  }   **else** {  NonTerminal nonTerminal = **new** NonTerminal(((NonTerminal) nextObject).getValue());   HashSet<Terminal> firstOfTheNT = nonTerminal.findFirst();   **for** (Terminal terminal : firstOfTheNT) {  toReturn.add(terminal);  }   }   }    }    **return** toReturn;  }      **private boolean** isLastIn(ArrayList<Object> rhsOfFoundNTKey, NonTerminal givenNT) {  **int** index= rhsOfFoundNTKey.indexOf(givenNT);   **if**(index == rhsOfFoundNTKey.size())  **return true**;   **else return false**;  }   **private** HashSet<NonTerminal> findInRHS(NonTerminal givenNT) {  HashSet<NonTerminal> founds = **new** HashSet<NonTerminal>();   LinkedHashMap<NonTerminal , ArrayList<Object>> rules = **new** ProductionRules().makeProductionTable();   **for** (Map.Entry<NonTerminal, ArrayList<Object>> entry : rules.entrySet()) {  ArrayList<Object> list = entry.getValue();   **if**(list.contains(givenNT))  founds.add(entry.getKey());  }   **return** founds;  }    @Override  **public boolean** equals(Object o) {  **if** (**this** == o) **return true**;  **if** (o == **null** || getClass() != o.getClass()) **return false**;   NonTerminal that = (NonTerminal) o;   **if** (**value** != **null** ? !**value**.equals(that.**value**) : that.**value** != **null**) **return false**;   **return true**;  }   @Override  **public int** hashCode() {  **return value** != **null** ? **value**.hashCode() : 0;  } } |
| **Observation & Learning:**  Write your Observations & Learning after performing task |
| **Conclusion:**  What is the use of Follow of a grammar. |