Assignment 4

**Assignment Guidelines:**

1. Deadlines should be kept in mind. No extension in assignment dates would be given. No late submission will be accepted.
2. This is an individual assignment. **PLAGIARISM IS NOT ACCEPTABLE AT ALL!** Zero marks will be given in case of plagiarism.

Deadline = 29-04-25, Deadline is hard and firm.

|  |
| --- |
| **Question no 1:** |
| Given the production rule **A → B C D E,** and the following semantic actions:  C.i = f(A.i, B.s)  D.i = f(C.s)  A.s = f(E.s)  Determine if this SDT is **L-attributed**. Justify your answer by identifying which attributes are synthesized and which are inherited. |

|  |
| --- |
| **Question no 2:** |
| Modify the following SDT so that it becomes L-attributed:  A → X Y Z  Y.i = f(Z.s)  A.s = f(Y.s) |

|  |
| --- |
| **Question no 3:** |
| Analyze the following C code snippet and identify all **semantic errors**, including:   * **Flow of control** issues * **Type compatibility** problems * **Uniqueness violations**   void process() { int value = 10; break;  float value = 5.5;  int result = value + "text"; if (result) {  continue;  }  return value;  }  Clearly state the **line number** of each semantic issue.  Classify each issue as a **flow of control error**, **type error**, or **uniqueness check error**. Suggest how to **correct** each problem. |

|  |
| --- |
| **Question no 4:** |
| Implement a mini type checker in C that supports:   * Basic types (int, float, char) * Assignment, arithmetic (+, -, \*, /), and conditional statements * Suggests explicit casts where needed |

|  |
| --- |
| **Question no 5:** |
| For the following expression  **abc+\*de+/**   * Convert into Three Address code * Draw its Syntax Tree. |

|  |
| --- |
| **Question no 6:** |
| For the following expression  **a−bcde↑↑/f\*g/+**   * Convert into Three Address code * Draw its Syntax Tree. |

|  |
| --- |
| **Question no 7:** |
| For the following C code, identify the **type of TAC** used for each line (assignment, unary, conditional, function call, etc.)  x = y + z; w = -x;  if (w > 0) goto L2; call print(w);   * List each statement * Mention the TAC **statement type** used |

|  |
| --- |
| **Question no 8:** |
| Given the expression:  a = b \* -(c - d) + b \* -(c - d);   1. Generate the **Quadruple** representation (op, arg1, arg2, result). 2. Generate the **Triple** representation (index-based). |