**ABHINAV TYAGI (DESD HYD – 50330001)**

Q5. Convert the following Infix Expressions to Post-fix Expressions.

**a. x^y/(5\*z)+2**

Solution:

Step1:

Expression: x

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

Step2: Push “^” into stack.

Expression: x

|  |  |  |  |
| --- | --- | --- | --- |
| ^ |  |  |  |

Step3: As “^” has higher precedence value than “/”. So it will be taken out of stack and “/” will be pushed in to the stack.

Expression: xy^

|  |  |  |  |
| --- | --- | --- | --- |
| / |  |  |  |

Step4: Push “(” into stack.

Expression: xy^

|  |  |  |  |
| --- | --- | --- | --- |
| / | ( |  |  |

Step5:

Expression: xy^5

|  |  |  |  |
| --- | --- | --- | --- |
| / | ( |  |  |

Step6: “\*” is pushed into the stack.

Expression: xy^5z

|  |  |  |  |
| --- | --- | --- | --- |
| / | ( | \* |  |

Step7: As “)” operator comes in the series. We will pop the operator “(“, “\*”,”/” and will use in the expression.

Expression: xy^5z\*/

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

Step8: Push”+” into stack.

Expression: xy^5z\*/2

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

Step9: Take out ”+” from the stack.

Final Expression: xy^5z\*/2+

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

b. **K + L - M\*N + (O^P) \* W/U/V \* T + Q**

Step1:

Expression: K

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

Step2: Push “+” into stack.

Expression: KL

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

Step3: Since “**-”** is an operator, is left-to-right associative, and has the same precedence as the “**+”** at the top of the stack, pop “**+”** from the stack and append it to the postfix expression.

Expression: KL+

|  |  |  |  |
| --- | --- | --- | --- |
| - |  |  |  |

Step4: Push “” into stack.

Expression: KL+M

|  |  |  |  |
| --- | --- | --- | --- |
| - | \* |  |  |

Step4: Since “**\*”** is an operator and has greater precedence than the “**-”** on the top of the stack, push “**\*”** to the top of the stack.

Expression: KL+MN

|  |  |  |  |
| --- | --- | --- | --- |
| - | \* |  |  |

Step5: Since “**+”** is an operator and has lower precedence than the “**\*”** on the top of the stack, pop “**\*”** from the stack and append it to the post-fix expression.

Expression: KL+MN\*

|  |  |  |  |
| --- | --- | --- | --- |
| - |  |  |  |

Step6: Since “**+”** is an operator, is left-to-right associative, and has the same precedence as the “**-”** at the top of the stack, pop “**-”** from the stack and append it to the post-fix expression.

Expression: KL+MN\*-

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

Step7: Since **(** is an opening parenthesis, push it to the top of the stack.

Expression: KL+MN\*-O

|  |  |  |  |
| --- | --- | --- | --- |
| + | ( |  |  |

Step8: Since **^** is an operator and has greater precedence than the **(** on the top of the stack, push **^** to the top of the stack.

Expression: KL+MN\*-OP

|  |  |  |  |
| --- | --- | --- | --- |
| + | ( | ^ |  |

Step9: Since the **)** is a closing parenthesis, pop each operator from the stack one at a time and append to the postfix expression. Keep popping from the stack until an opening parenthesis is encountered.

Pop ^ from the top of the stack and append to the postfix expression.

Expression: KL+MN\*-OP^

|  |  |  |  |
| --- | --- | --- | --- |
| + | ( |  |  |

Step10: Since **\*** is an operator and has greater precedence than the **+** on the top of the stack, push **\*** to the top of the stack.

Expression: KL+MN\*-OP^W

|  |  |  |  |
| --- | --- | --- | --- |
| + | \* |  |  |

Step11: Since **/** is an operator, is left-to-right associative, and has the same precedence as the **\*** at the top of the stack, pop **\*** from the stack and append it to the post-fix expression.

Expression: KL+MN\*-OP^W\*

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

Step12: Since **/** is an operator and has greater precedence than the **+** on the top of the stack, push **/** to the top of the stack.

Expression: KL+MN\*-OP^W\*U

|  |  |  |  |
| --- | --- | --- | --- |
| + | / |  |  |

Step13: Since **/** is an operator, is left-to-right associative, and has the same precedence as the **/** at the top of the stack, pop **/** from the stack and append it to the post-fix expression.

Expression: KL+MN\*-OP^W\*U/

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

Step13: Since **/** is an operator, is left-to-right associative, and has the same precedence as the **/** at the top of the stack, pop **/** from the stack and append it to the post-fix expression.

Expression: KL+MN\*-OP^W\*U/V

|  |  |  |  |
| --- | --- | --- | --- |
| + | / |  |  |

Step14: Since **\*** is an operator, is left-to-right associative, and has the same precedence as the **/** at the top of the stack, pop **/** from the stack and append it to the postfix expression.

Expression: KL+MN\*-OP^W\*U/V/

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

Step15: Since **\*** is an operator and has greater precedence than the **+** on the top of the stack, push **\*** to the top of the stack.

Expression: KL+MN\*-OP^W\*U/V/T

|  |  |  |  |
| --- | --- | --- | --- |
| + | \* |  |  |

Step16: Since **+** is an operator and has lower precedence than the **\*** on the top of the stack, pop **\*** from the stack and append it to the postfix expression.

Expression: KL+MN\*-OP^W\*U/V/T\*

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

Step17: Since **+** is an operator, is left-to-right associative, and has the same precedence as the **+** at the top of the stack, pop **+** from the stack and append it to the postfix expression.

Expression: KL+MN\*-OP^W\*U/V/T\*+

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

Step18: Push the **+** to the stack.

Expression: KL+MN\*-OP^W\*U/V/T\*+

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

Step19: Since **Q** is an operand, append it to the postfix expression.

Expression: KL+MN\*-OP^W\*U/V/T\*+Q

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

FINAL EXPRESSION: KL+MN\*-OP^W\*U/V/T\*+Q+

**c. A+(B\*C+D)/E**

Solution:

Step1: Take “A” in expression and Push “+” in stack.

Expression: A

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

Step2: Push “(” into stack. Take “B” in expression.

Expression: AB

|  |  |  |  |
| --- | --- | --- | --- |
| + | ( |  |  |

Step3: As “\*” has higher precedence value than “+”. So “\*” will be taken out of stack.

Expression: ABC\*

|  |  |  |  |
| --- | --- | --- | --- |
| + | ( | + |  |

Step4: As “)” operator comes in, pop each operator one by one until “(” reaches in the stack.

Expression: ABC\*D+

|  |  |  |  |
| --- | --- | --- | --- |
| + |  |  |  |

Step5: Push “/” on to the stack.

Expression: ABC\*D+5

|  |  |  |  |
| --- | --- | --- | --- |
| + | / |  |  |

Step 6: Final Expression is ABC\*D+5/+