```
Enter the infix expression: A+B*C+D
Expression Tree (Horizontally):


                                    [  +],1
        [  +],2                                 [  D],2
[  A],3                 [  *],3
                [  B],4         [  C],4

=== TRAVERSE TREE ===
Breadth-first: + + D A * B C
Inorder Traversal: A + B * C + D
Preorder Traversal: + + A * B C D
Postorder Traversal: A B C * + D +

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter the infix expression: (A+B)*(C+D)
Expression Tree (Horizontally):


                    [  *],1
       [  +],2                          [  +],2
[  A],3        [  B],3         [  C],3         [  D],3

=== TRAVERSE TREE ===
Breadth-first: * + + A B C D
Inorder Traversal: A + B * C + D
Preorder Traversal: * + A B + C D
Postorder Traversal: A B + C D + *

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter the infix expression: A*B+C*D
Expression Tree (Horizontally):


                    [  +],1
       [  *],2                          [  *],2
[  A],3        [  B],3         [  C],3         [  D],3

=== TRAVERSE TREE ===
Breadth-first: + * * A B C D
Inorder Traversal: A * B + C * D
Preorder Traversal: + * A B * C D
Postorder Traversal: A B * C D * +

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter the infix expression: A+B+C+D
Expression Tree (Horizontally):


                                    [  +],1
                        [  +],2                 [  D],2
            [  +],3                 [  C],3
[  A],4             [  B],4

=== TRAVERSE TREE ===
Breadth-first: + + D + C A B
Inorder Traversal: A + B + C + D
Preorder Traversal: + + + A B C D
Postorder Traversal: A B + C + D +

...Program finished with exit code 0
Press ENTER to exit console.
```