Міністерство освіти і науки України
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА
Факультет прикладної математики та інформатики

Кафедра програмування

ЛАБОРАТОРНА РОБОТА № 5

Зворотній польський запис

з курсу “Алгоритми та структури даних”

Виконав:

Студент групи ПМІ-16

Бевз Маркіян Юрійович

Львів – 2024

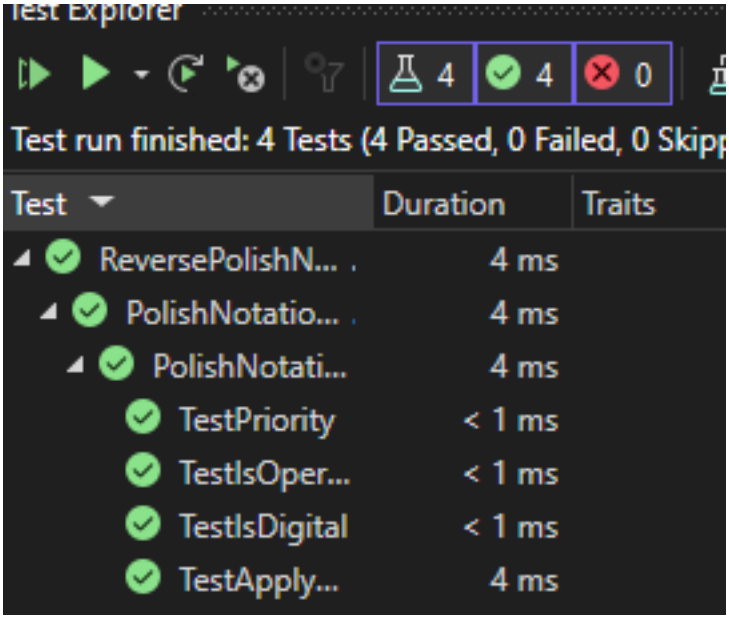
Мета:Ознайомитись з принципом роботи зворотнього польського запису (ЗПЗ) та його застосуванням у обчисленнях математичних виразів.

Принцип роботи зворотнього польського запису (ЗПЗ):

1. **Запис виразу:** Операнди та оператори записуються у послідовності, де оператор розміщується після своїх операндів.
2. **Обчислення виразу:** Вираз обчислюється під час його читання зліва направо.
3. **Застосування операторів:** Коли зустрічається оператор, він відразу ж застосовується до двох передніх операндів.
4. **Використання стеку:** Для обчислення виразів у ЗПЗ зазвичай використовується стек.
    - **Операнди:** Кожен операнд (число) додається до стеку.
    - **Оператори:** Коли зустрічається оператор, він вилучається зі стеку, і до результату застосовується операція з цим оператором до двох останніх операндів.
5. **Збереження результату:** Результат також додається до стеку, і процес продовжується, поки весь вираз не буде обчислений.

Цей принцип дозволяє обходити потребу в дужках у математичних виразах та забезпечує однозначне інтерпретування виразів.

Хід роботи: Після вивчення принципу роботи полського запису я реалізував усі необхідні методи й сам клас. Також було створено 4 тести для перевірки правильності роботи цих методів. Нижче буде прикріплено результат виконання програми, тестів, та їх код.



```
TEST_METHOD(TestPriority)
{
    Assert::AreEqual(3, priority('^'));
    Assert::AreEqual(2, priority('*'));
    Assert::AreEqual(2, priority('/'));
    Assert::AreEqual(1, priority('+'));
    Assert::AreEqual(1, priority('-'));
    Assert::AreEqual(0, priority('#'));
    Assert::AreEqual(0, priority('('));
    Assert::AreEqual(0, priority(')'));
    Assert::AreEqual(-1, priority('a'));
}

TEST_METHOD(TestIsDigital)
{
    Assert::IsTrue(isDigital('0'));
    Assert::IsTrue(isDigital('5'));
    Assert::IsTrue(isDigital('9'));
    Assert::IsFalse(isDigital('a'));
    Assert::IsFalse(isDigital('+'));
    Assert::IsFalse(isDigital('('));
    Assert::IsFalse(isDigital(')'));
}

TEST_METHOD(TestIsOperand)
{
    Assert::IsTrue(isOperand('+'));
    Assert::IsTrue(isOperand('-'));
    Assert::IsTrue(isOperand('*'));
    Assert::IsTrue(isOperand('/'));
    Assert::IsTrue(isOperand('^'));
    Assert::IsTrue(isOperand('('));
    Assert::IsTrue(isOperand(')'));
    Assert::IsFalse(isOperand('0'));
    Assert::IsFalse(isOperand('9'));
    Assert::IsFalse(isOperand('a'));
}

TEST_METHOD(TestApplyOperator)
{
    Assert::AreEqual(5.0, applyOperator(2.0, 3.0, '+'));
    Assert::AreEqual(-1.0, applyOperator(2.0, 3.0, '-'));
    Assert::AreEqual(6.0, applyOperator(2.0, 3.0, '*'));
    Assert::AreEqual(2.0 / 3.0, applyOperator(2.0, 3.0, '/'));
    Assert::AreEqual(pow(2.0, 3.0), applyOperator(2.0, 3.0, '^'));
    Assert::AreEqual(0.0, applyOperator(2.0, 3.0, '@'));
}
```

```
1. Enter infix notation
2. Enter postfix notation
3. Exit
Enter your choice: 1

Enter infix notation: (9 + 9) ^ 2 - (7 * 6)
Postfix notation: 99+2^76*-
Result: 282
```

Висновок: На цій лабораторні роботі я ознайомився з принципом роботи зворотнього польського запису (ЗПЗ) та його застосуванням у обчисленнях математичних виразів.