도전 과제 11

마감일시: 2023년 5월 28일 오후 11시

사칙연산만을 수행할 수 있는 아주 간단한 계산기 프로그램을 작성하려고 한다. 이 계산기는 라인(line)별로 사칙연산을 텍스트로 입력 받고 결과를 텍스트로 출력한다. 계산기는 한 사칙연산식의 결과를 저장하기 위한 변수 result를 가진다. 연산을 수행하기 전에 result는 0으로 초기화된다. 사용자는 result에 더하거나 빼거나 곱하거나 나누는 연산만을 수행할 수 있다. 계산기의result에 한 사칙연산을 수행하기 위해 한 라인(줄)에 다음과 같이 입력한다.

연산자 숫자

위에서 연산자는 +, -, \*, / 중의 하나여야 한다. 다른 연산자가 입력되면 연산을 수행할 수 없어 적절한 예외 처리를 해야 한다. 예를 들면, 계산기의result에 3.5를 더하기 위해 다음과 같이 입력한다.

+ 3.5

그러면 계산기는 다음 연산을 수행한 후 결과인 3.5를 result에 저장하고 출력한다.

result + 3.5 = 0 + 3.5 = 3.5

사용자는 여러 연산들을 계속해서 수행할 수 있다. 다음은 사용자와 프로그램사이의 상호작용을 보여주는 예이다.

프로그램: **result = 0.0**

사용자: *+ 8.2*

프로그램: **result + 8.2 = 8.2**

**갱신된 result = 8.2**

사용자: *- 3.4*

프로그램: **result – 3.4 = 4.8**

**갱신된 result = 4.8**

위의 예에서 이태릭체로 쓰여진 부분은 사용자 입력이고 고딕체로 쓰여진 부분은 프로그램의 출력이다. 사용자가 연산을 끝내기 위해서는 e 또는 E를 입력하면 된다.

이 계산기 프로그램에서는 두 가지 유형의 사용자 입력 오류를 처리해야 한다. 첫 번째 입력 오류는 나눗셈 연산에서 0으로 나누려고 시도하는 경우이다. 이 경우에는 나눗셈을 수행할 수 없으므로 오류 메시지를 출력하고 프로그램을 강제로 중단시켜야 한다. 다음은 이 경우의 프로그램 실행 예이다.

프로그램: **result = 0.0**

사용자: *+ 3.5*

프로그램: **result + 3.5 = 3.5**

**갱신된 result = 3.5**

사용자: */ 0*

프로그램: **0 혹은 0에 아주 가까운 값으로 나누려고 시도하여**

**프로그램이 중단되었습니다.**

두 번째 입력 오류는 사칙연산이 아닌 연산을 시도하는 경우이다. 이 경우에는 적절한 오류 메시지를 출력하고 사용자에게 처음부터 다시 시작하라고 요구해야 한다. 다음은 이런 경우의 프로그램 실행 예이다.

프로그램: **result = 0.0**

사용자: *+ 2.5*

프로그램: **result + 2.5 = 2.5**

**갱신된 result = 2.5**

사용자: *% 3*

프로그램: **%는 허용되지 않는 연산자이다.**

**처음부터 다시 시작하세요.**

이 프로그램을 작성하기 위해 다음과 같이 하라. **클래스 설계는 UML클래스 다이어그램으로 해야 한다.**

1. 나눗셈에서 0으로 나누려고 시도하여 발생하는 예외를 모델하는 DivideByZeroException이라는 예외 클래스를 설계하고 구현하라. 이 클래스는 다음과 같은 생성자 메소드들을 포함한다.
2. Exception 클래스의 생성자 메소드에 ‘0으로 나누려고 시도!’라는 메시지를 넘겨주는 기본 생성자 메소드
3. 예외가 발생한 이유를 매개변수로 갖는 생성자 메소드.
4. 사칙연산에서 허용되지 않는 연산자로 인해 발생하는 예외를 모델하는 UnknownOpException이라는 예외 클래스를 설계하고 구현하라. 이 클래스는 다음과 같은 생성자 메소드들을 포함한다.
5. Exception 클래스의 생성자 메소드에 ‘허용되지 않는 연산자로 인한 예외’라는 메시지를 넘겨주는 기본 생성자 메소드
6. 허용되지 않는 연산자를 문자형 매개변수 op로 갖고 Exception 클래스의 생성자 메소드에 ‘op는 허용되지 않는 연산자이다.’라는 메시지를 넘겨주는 생성자 메소드
7. 예외가 발생한 이유를 매개변수로 갖는 생성자 메소드
8. 사칙연산을 수행할 수 있는 계산기를 나타내는 Calculator클래스를 설계하고 구현하라. 이 클래스는 사칙연산의 결과를 저장하기 위한 변수 result를 가진다. 이 클래스는 다음과 같은 일들을 수행하는 메소드들을 포함한다.
9. result를 0으로 초기화하면서 Calculator 객체를 생성한다.
10. result를 반환한다.
11. result의 값을 넘겨 받은 값으로 변경한다.
12. 사용자로부터 ‘연산자 숫자’의 형태로 연산식을 입력 받거나 ‘e’(혹은 ‘E’)를 입력 받는다. 연산자가 입력되면 숫자를 피연산자로 하여 대응하는 연산을 수행하고 갱신된 결과를 출력하고 사용자로부터 계속 입력을 받는다. ‘e’(혹은 ‘E’)가 입력되면 메소드 수행을 종료한다.
13. 연산자를 나타내는 문자 op, 첫 번째 피연산자 oprd1과 두 번째 피연산자 oprd2를 넘겨 받아 연산 ‘oprd1 op oprd’를 수행하고 결과를 반환한다. 허용되는 연산자는 '+', '-', '\*', 혹은 '/'이다. 만약 op가 허용되지 않는 연산자라면 op를 매개변수로 갖는 UnknownOpException이라는 예외를 발생시켜야 한다. 만약 op가 /이고 oprd2가 0 이거나 0.0001보다 작고 -0.0001보다 크다면 DivideByZeroException이라는 예외를 발생시켜야 한다.
14. DivideByZeroException이라는 예외를 처리하기 위해 ‘0 혹은 0에 아주 가까운 값으로 나누려고 시도하여 프로그램이 중단되었습니다.’를 출력하고 프로그램을 종료시킨다.
15. UnknownOpException이라는 예외를 처리하기 위해 다음을 순서대로 출력한다.
    * 처음부터 다시 시작하세요.
    * 연산식을 나타내는 각 라인(line)의 포맷:
    * 연산자 숫자
    * 예: + 3
    * 끝내려면 e(혹은 E)를 입력하세요.

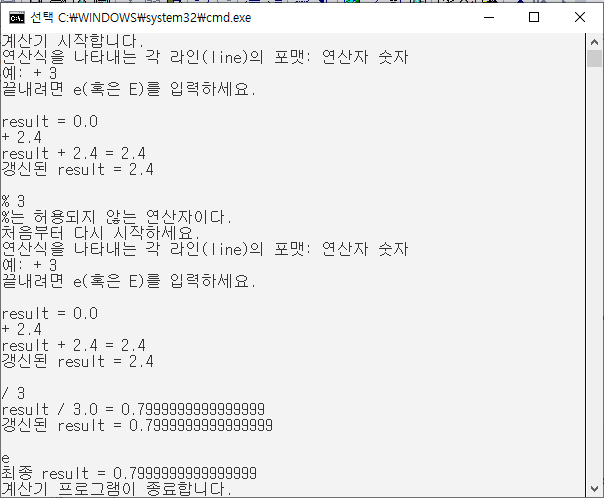
다음으로 (4)번을 수행하는 메소드를 호출한다.

1. Calculator 객체를 생성한다. 그리고 나서 다음을 순서대로 출력한다.
   * 계산기 시작합니다.
   * 연산식을 나타내는 각 라인(line)의 포맷:
   * 연산자 숫자
   * 예: + 3
   * 끝내려면 e(혹은 E)를 입력하세요.

다음으로 (4)번을 수행하는 메소드를 호출한다. 만약 호출된 메소드에서 UnknownOpException이나 DivideByZeroException이라는 예외가 발생하면 각각 (6) 번이나 (7) 번을 수행하는 메소드를 호출하여 예외를 처리해야 한다. 예외가 발생하지 않으면 최종 결과값(result의 값)과 ‘계산기 프로그램이 종료합니다.’를 출력하고 프로그램을 종료해야 한다.

(8)번을 수행하는 메소드가 main 메소드이다. (4), (5), (6), (7)과 (8)번을 수행하는 메소드의 설계를 해야 한다.

다음은 사칙연산에서 허용되지 않는 연산식을 포함하는 경우의 프로그램 실행 예이다.



다음은 나눗셈에서 0으로 나누려고 시도하는 연산식을 포함하는 경우에 프로그램 실행이 중단된 예이다.

