ZADANIE

Wielomiany (treść, rozwiązanie i ocena jakości)

Otwarto: poniedziałek, 8 listopada 2021, 08:30
Wymagane do: wtorek, 23 listopada 2021, 20:00

Wprowadzenie

Jednomian zmiennej `x` to wyrażenie postaci `ax^n`, czyli iloczyn *współczynnika* `a` i `n`-tej potęgi `x`. Wartość `n`, nazywana *stopniem jednomianu*, jest nieujemną liczbą całkowitą. Jednomian o współczynniku równym `0` to *jednomian zerowy*.

Wielomianem zmiennej `x` nazywamy sumę jednomianów zmiennej `x`. Wielomian, którego wszystkie jednomiany są zerowe, to *wielomian zerowy*.

Stopniem niezerowego wielomianu jest maksymalny stopień jego niezerowego jednomianu. Przyjmujemy, że stopniem wielomianu zerowego jest `-1`.

Na wielomianach określone są operacje *sumy* i *iloczynu*.

Polecenie

Zaimplementuj kalkulator liczący sumę i iloczyn wielomianów o współczynnikach całkowitych.

Kalkulator ma pamięć, nazywaną akumulatorem, która przechowuje jeden wielomian. Wartością początkową akumulatora jest wielomian zerowy.

Kalkulator wykonuje polecenia obliczenia sumy i iloczynu wartości akumulatora oraz wielomianu, który jest argumentem polecenia. Polecenie pisze obliczony wynik na wyjście i zapamiętuje go w akumulatorze.

Postać danych

Dane programu to ciąg wierszy z poleceniami, zakończony wierszem zaczynającym się od kropki `.`. Każde polecenie zajmuje jeden wiersz.

Wiersz polecenia obliczenia sumy zaczyna się od znaku `+` a wiersz polecenia obliczenia iloczynu zaczyna się od znaku `*`. Kolejne znaki, aż do końca wiersza, są zapisem argumentu polecenia.

Składnię zapisu wielomianu opisujemy gramatyką, z symbolem początkowym `<wielomian>`, w rozszerzonej notacji BNF:

```
<wielomian> ::= "0" | [ "-" ] <jednomian> { <operacja> <jednomian> }
<operacja> ::= "+" | "-"
<jednomian> ::= "1" | <dużo> | [ <dużo> ] "x" [ "^" <dużo> ]
<dużo> ::= "1" <cyfra> { <cyfra> } | <cyfra od 2 do 9> { <cyfra> }
<cyfra> ::= "0" | "1" | <cyfra od 2 do 9>
<cyfra od 2 do 9> ::= "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

Nawiasy klamrowe oznaczają tu powtórzenie zero lub więcej razy, nawiasy kwadratowe otaczają fragmenty opcjonalne a kreska pionowa to alternatywa. Symbole pomocnicze są ujęte w nawiasy kątowe a symbole końcowe są ujęte w cudzysłowy. W zapisie wielomianu symbolowi końcowemu odpowiada znak, który występuje między cudzysłowami.

Dodatkowo, jednomiany wielomianu są uporządkowane w kolejności malejących stopni.

W wierszu polecenia może być dowolna liczba spacji. Niepusty ciąg spacji nie występuje jednak ani na początku wiersza ani bezpośrednio pomiędzy dwiema cyframi.

?

Postać wyniku

Dla każdego wykonanego polecenia program pisze na wyjście jeden wiersz z jego wynikiem. Składnia zapisu wielomianu w wyniku programu jest taka sama, jak w danych. Przed i po wystąpieniu symbolu końcowego `"+"` lub `"-"`, w produkcjach symbolu pomocniczego `<operacja>`, jest po jednej spacji. Oprócz tego, żadnych innych spacji w wyniku programu nie ma.

Przykłady

Do treści zadania dołączone są pliki `.in` z przykładowymi danymi i pliki `.out` z wynikami wzorcowymi.

- Dla danych przyklad1.in poprawny wynik to przyklad1.out .
- Dla danych przyklad2.in poprawny wynik to przyklad2.out .
- Dla danych przyklad3.in poprawny wynik to przyklad3.out .

Walidacja i testy

- Rozwiązania podlegają walidacji, wstępnie badającej zgodność ze specyfikacją.

  Walidacja sprawdza działanie programu na przykładach dołączonych do treści zadania.

  Pomyślne przejście walidacji jest warunkiem dopuszczenia programu do testów poprawności. Program, który walidacji nie przejdzie, dostaje zerową ocenę poprawności.
- Walidacja i testy są prowadzone na komputerze `students`.
- Programy są kompilowane poleceniem:

```
gcc @opcje nazwa.c -o nazwa
```

  gdzie `nazwa.c` to nazwa pliku z kodem źródłowym a plik `opcje` ma zawartość:

```
-std=c17
-pedantic
-Wall
-Wextra
-Wformat-security
-Wduplicated-cond
-Wfloat-equal
-Wshadow
-Wconversion
-Wjump-misses-init
-Wlogical-not-parentheses
-Wnull-dereference
-Wvla
-Werror
-fstack-protector-strong
-fsanitize=undefined
-fno-sanitize-recover
-g
-fno-omit-frame-pointer
-O1
```

  Opcje `-std=c17`, `-pedantic` wskazują, że kompilator ma dbać o zgodność kodu z aktualnym standardem języka C.

  Dzięki opcjom `-Wall`, `-Wextra` kompilator zgłosi zauważone usterki.

  Opcje `-Wformat-security`, `-Wduplicated-cond`, `-Wfloat-equal`, `-Wshadow`, `-Wconversion`, `-Wjump-misses-init`, `-Wlogical-not-parentheses`, `-Wnull-dereference` umożliwiają wykrywanie dodatkowych usterek.

  Opcja `-Wvla` sprawia, że użycie tablic zmiennej długości jest uznawane za usterkę.

  Opcja `-Werror` wskazuje, że kompilator ma uznać usterki za błędy.

  Dzięki opcji `-fstack-protector-strong`, podczas wykonania programu zostaną wykryte niektóre błędne odwołania do pamięci na stosie.

  Opcje `-fsanitize=undefined`, `-fno-sanitize-recover` umożliwiają wykrywanie operacji, które mają efekt nieokreślony.

  Opcje `-g`, `-fno-omit-frame-pointer` poprawiają jakość komunikatów o błędach wykonania.

  Opcja `-O1` włącza optymalizacje, co zwiększa prawdopodobieństwo ujawnienia się błędów.

  Wymagane są wszystkie wymienione opcje kompilatora. Nie będą do nich dodawane żadne inne.