Duże zadanie, część 2

Jako drugą część zadania należy zaimplementować kalkulator działający na wielomianach i stosujący odwrotną notację polską.

Kalkulator

Program kalkulatora czyta dane wierszami ze standardowego wejścia. Wiersz zawiera wielomian lub polecenie do wykonania.

Wielomian reprezentujemy jako stałą, jednomian lub sumę jednomianów. Stała jest liczbą całkowitą. Jednomian reprezentujemy jako parę `(coeff,exp)`, gdzie współczynnik `coeff` jest wielomianem, a wykładnik `exp` jest liczbą nieujemną. Do wyrażenia sumy używamy znaku `+`. Jeśli wiersz zawiera wielomian, to program wstawia go na stos. Przykłady poprawnych wielomianów:

```
0
1
-2
(0,0)
(1,0)
(-2,0)
(1,1)
(1,0)+(1,2)
(1,2)+(1,0)
(1,2)+(-1,2)
(1,2)+(-2,2)
((1,2),15)+(-7,8)
(3,1)+(((4,4),100),2)
```

Kalkulator wykonuje następujące polecenia:

- `ZERO` – wstawia na wierzchołek stosu wielomian tożsamościowo równy zeru;
- `IS_COEFF` – sprawdza, czy wielomian na wierzchołku stosu jest współczynnikiem – wypisuje na standardowe wyjście 0 lub 1;
- `IS_ZERO` – sprawdza, czy wielomian na wierzchołku stosu jest tożsamościowo równy zeru – wypisuje na standardowe wyjście 0 lub 1;
- `CLONE` – wstawia na stos kopię wielomianu z wierzchołka;
- `ADD` – dodaje dwa wielomiany z wierzchu stosu, usuwa je i wstawia na wierzchołek stosu ich sumę;
- `MUL` – mnoży dwa wielomiany z wierzchu stosu, usuwa je i wstawia na wierzchołek stosu ich iloczyn;
- `NEG` – neguje wielomian na wierzchołku stosu;
- `SUB` – odejmuje od wielomianu z wierzchołka wielomian pod wierzchołkiem, usuwa je i wstawia na wierzchołek stosu różnicę;
- `IS_EQ` – sprawdza, czy dwa wielomiany na wierzchu stosu są równe – wypisuje na standardowe wyjście 0 lub 1;
- `DEG` – wypisuje na standardowe wyjście stopień wielomianu (−1 dla wielomianu tożsamościowo równego zeru);
- `DEG_BY idx` – wypisuje na standardowe wyjście stopień wielomianu ze względu na zmienną o numerze `idx` (−1 dla wielomianu tożsamościowo równego zeru);
- `AT x` – wylicza wartość wielomianu w punkcie `x`, usuwa wielomian z wierzchołka i wstawia na stos wynik operacji;
- `PRINT` – wypisuje na standardowe wyjście wielomian z wierzchołka stosu;
- `POP` – usuwa wielomian z wierzchołka stosu.

Wypisywany poleceniem `PRINT` wielomian powinien mieć jak najprostszą postać. Wykładniki wypisywanych jednomianów nie powinny się powtarzać. Jednomiany powinny być posortowane rosnąco według wykładników.

Podane wyżej wielomiany powinny zostać wypisane następująco:

```
0
1
-2
0
1
-2
(1,1)
(1,0)+(1,2)
(1,0)+(1,2)
0
(-1,2)
(-7,8)+((1,2),15)
(3,1)+(((4,4),100),2)
```

Sprawdzanie poprawności danych wejściowych i obsługa błędów

Program nie powinien zakładać maksymalnej długości wiersza. Poprawny wiersz nie zawiera żadnych dodatkowych białych znaków oprócz pojedynczej spacji separującej parametr poleceń `AT` i `DEG_BY` od polecenia. Program wypisuje komunikaty o błędach na standardowe wyjście błędów. Poniżej `w` oznacza numer wiersza, a `\n` – znak przejścia do nowego wiersza. Wiersze i kolumny numerujemy od 1.

Ignorujemy wiersze zaczynające się znakiem `#` i puste. Jeśli wiersz zaczyna się małą lub wielką literą alfabetu angielskiego, to uznajemy, że zawiera polecenie. W pozostałych przypadkach uznajemy, że wiersz opisuje wielomian.

Jeśli program wykryje niepoprawną nazwę polecenia, wypisuje:

```
ERROR w WRONG COMMAND\n
```

Jeśli w poleceniu `DEG_BY` nie podano parametru lub jest on niepoprawny, program wypisuje:

```
ERROR w DEG BY WRONG VARIABLE\n
```

Jeśli w poleceniu `AT` nie podano parametru lub jest on niepoprawny, program wypisuje:

```
ERROR w AT WRONG VALUE\n
```

Jeśli na stosie jest za mało wielomianów, aby wykonać polecenie, program wypisuje:

```
ERROR w STACK UNDERFLOW\n
```

Jeśli program wykryje błąd podczas parsowania wielomianu, wypisuje

```
ERROR w WRONG POLY\n
```

Wartość współczynnika jednomianu lub parametru polecenia `AT` uznajemy za niepoprawną, jeśli jest mniejsza od `-9223372036854775808` lub większa od `9223372036854775807`.

Wartość wykładnika jednomianu uznajemy za niepoprawną, jeśli jest mniejsza od `0` lub większa od `2147483647`.

Wartość parametru polecenia `DEG_BY` uznajemy za niepoprawną, jeśli jest mniejsza od `0` lub większa od `18446744073709551615`.

Program może, ale nie musi, akceptować liczby z dodatkowymi wiodącymi zerami. Znak `+` służy tylko do wyrażania sumy jednomianów i nie może poprzedzać liczby.

Przykłady użycia

Do treści zadania dołączone jest archiwum z przykładami użycia.

Rozwiązanie

Rozwiązanie części 2 zadania powinno korzystać z własnego rozwiązania części 1. Jako rozwiązanie części 2 zadania wymagamy:

- umieszczenia kodu źródłowego implementacji w katalogu `src`,
- uzupełnienia dokumentacji dla programu `doxygen`,
- dostosowania pliku konfiguracyjnego dla programu `cmake`.

Obowiązują wymagania sformułowane w punkcie „Wymagamy” w treści pierwszej części zadania. Zmieniamy jedno wymaganie: teraz funkcja `main` powinna być zawarta w pliku `src/calc.c`. Plik `src/poly_example.c` może pozostać niezmieniony. Pozostawiamy natomiast wymaganie, że w wyniku kompilacji powinien powstać plik wykonywalny `poly`.

Obowiązują wymagania sformułowane w punkcie „Obsługa błędów krytycznych” w treści pierwszej części zadania.