**Zadanie Wielomiany**

**Szczegółowe wymagania dotyczące implementacji klas.**

* **w rozwiązaniach nie wolno korzystać z kolekcji STL**, np. z <vector>. Rozwiązania wykorzystujące pliki nagłówkowe z biblioteki standardowej inne niż
  + **<iostream>**,
  + **<sstream>**,
  + **<string>**,
  + **<cstdlib>** oraz
  + **<cstring>**

będą automatycznie odrzucone.

* Wszystkie klasy powinny poprawnie zarządzać pamięcią, w szczególności konstruktory kopiujące, destruktory i operatory przypisania powinny być zdefiniowane wszędzie tam, gdzie jest to konieczne.
* Wszędzie gdzie ma to sens należy akceptować stałe obiekty oraz tablice obiektów stałych jako argumenty funkcji. W szczegółowych opisach poniżej nie będzie to sprecyzowane.

**Wymagany publiczny interfejs klasy Z2.**

Klasa **Z2** powinna dostarczać następujących publicznych metod:

**Konstruktor domyślny**

tworzy liczbę zero

**Konstruktor z argumentem typu short int**

tworzy liczbę Z2 na podstawie podanego argumentu

**Operator konwersji do short int**

zwraca liczbę naturalną ze zbioru {0,1}

**operator+=**

realizuje operację **a+=b** dla liczb w ciele Z2. Zwraca referencję do uaktualnionego obiektu **a**.

**operator\*=**

realizuje operację **a\*=b** dla liczb w ciele Z2. Zwraca referencję do uaktualnionego obiektu **a**.

**operator/=**

realizuje operację **a/=b** dla liczb w ciele Z2. Zwraca referencję do uaktualnionego obiektu **a**. W przypadku dzielenia przez zero należy wypisać na standardowe wyjście łańcuch

**"Dzielenie przez zero\n"**

i zwrócić niezmieniony obiekt **a**.

Ponadto należy zdefiniować następujące operatory jako **funkcje globalne**:

**operator+**

zwraca obliczony wynik **a+b** jako obiekt **Z2**.

**operator\***

zwraca obliczony wynik **a\*b** jako obiekt **Z2**.

**operator/**

zwraca obliczony wynik **a/b** jako obiekt **Z2**. W przypadku dzielenia przez zero należy wypisać na standardowe wyjście łańcuch

**"Dzielenie przez zero\n"**

i zwrócić niezmieniony obiekt **a**.

**operator<<**

realizuje wypisanie liczby **Z2** do strumienia **std::ostream** jako liczby całkowitej ze zbioru {0,1}

**Wymagany publiczny interfejs klasy Wielomian.**

Klasa **Wielomian** powinna dostarczać następujących publicznych metod:

**Konstruktor domyślny**

tworzy wielomian stopnia zero równy zero

**Konstruktor z argumentem typu unsigned int oraz tablicą obiektów Z2**

tworzy Wielomian o długości podanej jako pierwszy argument o współczynnikach z podanej tablicy. Należy przyjąć założenie, że tablica zawiera wystarczającą liczbę elementów. Zakładamy również, że współczynniki są podane w kolejności od wyrazu wolnego, do współczynnika przy najwyższej potędze.

**operator[]**

zwraca współczynnik wielomianu przy **i**-tej potędze. Indeks jest liczbą typu **unsigned int**. W przypadku, gdy podany indeks jest niepoprawny (większy od stopia wielomianu), należy wypisać na standardowe wyjście łańcuch

**"Niepoprawny indeks wielomianu\n"**

oraz zwrócić współczynnik przy zerowej potędze wielomianu.

**operator+=**

realizuje operację **a+=b** dla wielomianów. Zwraca referencję do uaktualnionego obiektu **a**.

**operator\*=**

realizuje operację **a\*=b** dla wielomianów. Zwraca referencję do uaktualnionego obiektu **a**.

**degree**

bezargumentowa funkcja zwracająca stopien wielomianu jako liczbę typu **unsigned int**.

**toString**

funkcja o argumencie **xVar** typu **std::string** i wartości typu **std::string**. Zwraca wzór wielomianu w postaci sumy kolejnych potęg zmiennej **xVar** w kolejności od wyrazu wolnego do najwyższej potęgi. Ponadto należy:

* wielomian zerowy wypisać jako **0**. Jest to wyjątek, poza którym nie wypisujemy współczynników zerowych.
* niezerowy wyraz wolny wypisać jako liczbę **1**
* pozostałe potęgi wypisać jako **xVar^i**

Przykłady poprawnych wartości dla wywołania **toString("x")**:

* **0**
* **1**
* **x^1**
* **1+x^7+x^1234**

Ponadto należy zdefiniować następujące **funkcje globalne**:

**operator+, operator\***

zwracają sumę i iloczyn wielomianów **a+b** oraz **a\*b**.

**mod**

czteroargumentowa funkcja globalna o argumentach kolejno **u,v,q,r**. Jej zadaniem jest obliczenie wyniku z dzielenia i reszty z dzielenia wielomianów **u** i **v**. Wyniki obliczeń są zapisane w argumentach **q (wynik), r (reszta)** przesłanych do funkcji przez referencję. Obliczone wielomiany **q,r** mają spełniać tożsamość  
**u = q\*v + r**  
przy czym stopień **r** jest mniejszy niż stopień **v**. Wyjątkiem jest dzielenie przez wielomian **v=1**, gdzie wynik i reszta to odpowiednio **q=u, r=0**.  
W przypadku dzielenia przez wielomian zerowy należy wysłać do standardowego strumienia łańcuch

**"Dzielenie przez zero\n"**

i pozostawić argumenty **q,r** niezmienione.

**operator/**

zwraca wynik z dzielenia dwóch wielomianów **u/v** (jak w funkcji **mod**). W przypadku dzielenia przez wielomian zerowy należy wysłać do standardowego strumienia łańcuch

**"Dzielenie przez zero\n"**

i zwrócić obiekt **u**.

**operator%**

zwraca resztę z dzielenia dwóch wielomianów **u/v** (jak w funkcji **mod**). W przypadku dzielenia przez wielomian zerowy należy wysłać do standardowego strumienia łańcuch

**"Dzielenie przez zero\n"**

i zwrócić wielomian zerowy.

**operator<<**

realizuje wypisanie wielomianu **a** do strumienia w następującym formacie

**{a0,a1,...,an}**

gdzie **n** jest stopniem wielomianu, **ai** to współczynnik przy **i**-tej potędze.

**operator>>**

realizuje wczytanie wielomianu **a** z podanego strumienia. Operator ma ignorować wszystkie znaki strumienia aż do pierwszego napotkanego nawiasu **'{'**. Wtedy rozpoczyna czytanie kolejnych współczynników wielomianu aż do napotkania zamykającego nawiasu **'}'**. Zakładamy, że w strumieniu wejściowym jest wielomian w formacie zgodnym z tym, co zostało wydrukowane za pomocą **operator<<** (z dokładnością do pewnej liczby początkowych nieokreślonych znaków poprzedzających **'{'**).

**Test jawny.**

Kod źródłowy testu jawnego: [link](https://baca.ii.uj.edu.pl/progmk2020/problems/14/test.cpp)  
Wyjście testu jawnego: [link](https://baca.ii.uj.edu.pl/progmk2020/problems/14/test.out)

**Wskazówka.**

Do testowania poprawności Państwa implementacji można użyć Mathematica. Załączam przykładowy [Notebook](https://baca.ii.uj.edu.pl/progmk2020/problems/14/polynomial.nb).