Marsjańskie DNA

Rosja znana jest ze swoich sukcesów w dziedzinie eksploracji wszechświata. Niedawno rosyjscy naukowcy podczas analizy próbek marsjańskiej gleby znaleźli dziwne cząsteczki, które mogą być pewnym rodzajem DNA. W przeciwieństwie do normalnego, ziemskiego DNA, cząsteczka ta ma dwa bazowe elementy, zamiast czterech. Całą cząsteczkę można zatem zapisać jako słowo złożone z zer i jedynek.

Naukowcy zbadali długość cząsteczki i okazało się, że ma ona n bazowych elementów. Teraz chcą oni odkryć jej strukturę, tj. znaleźć ciąg zer i jedynek S kodujący elementy tego DNA. Aby to zrobić, mogą oni wykonywać testy w specjalnym analizatorze DNA. W każdym teście ustawia się słowo opisające ciąg bazowych elementów, zakodowany jako słowo P, a analizator sprawdza, czy słowo to pojawia się w marsjańskim DNA, tj. czy słowo P jest podsłowem słowa S.

Próbka jest bardzo mała, dlatego naukowcy mogą wykonać jedynie t testów. Pomóż im wykonać odpowiednie testy, aby określić strukturę DNA.

Szczegóły implementacji

Powinieneś napisać jedną funkcję (metodę):

- `string analyse(int n, int t)`. Ta funkcja powinna wykonywać testy, używając używając funkcji (metody) bibliotecznej `make_test` i rozkwikłać strukturę DNA.
  - `n`: długość DNA.
  - `t`: liczba dozwolonych testów.
  - funkcja powinna dawać w wyniku znalezione słowo `S` opisujące DNA.

Funkcje biblioteczne

- `make_test(string p)`. Ta funkcja sprawdza, czy słowo P jest podsłowem słowa S.
  - `p`: słowo, które zostanie sprawdzone na okoliczność bycia podsłowem.
  - funkcja zwraca `true`, jeżeli P jest podsłowem S, natomiast `false` w przeciwnym wypadku.

Przykład

Program sprawdzający wywołuje funkcję z następującymi parametrami:

- `analyse(3, 7)`. Długość słowa S wynosi 3. Dozwolona liczba testów wynosi natomiast 7.

Program zawodnika wykonuje następujące wywołania:

- `make_test("00")` zwraca `false`.
- `make_test("01")` zwraca `true`.
- `make_test("10")` zwraca `true`.
- `make_test("11")` zwraca `false`.
- `make_test("010")` zwraca `false`.

Teraz jedynym możliwym słowem jest `"101"`, zatem funkcja `analyse` zwraca `"101"`.

Podzadania

1. (11 punktów) $n \leq 5, t = 31$.
2. (25 punktów) $n \leq 100, t = 256$.
3. (64 punkty) $n \leq 1000, t = 1024$.

Przykładowy program sprawdzający

Przykładowy program sprawdzający wczytuje dane w następującym formacie:

- Wiersz 1: słowo S.
- Wiersz 2: liczba całkowita t.

Języki programowania

Szczegóły implementacji w Twoim języku programowania znajdują się w dostarczonych plikach z szablonami.