GRUPA A

Ruby-A-3 Plik wejściowy zawiera linie postaci:
identyfikator = liczba
Po obu stronach znaku równości może być dowolnie wiele znaków odstępu lub tabulacji. Napisać program sumujący liczby przypisane poszczególnym identyfikatorom, które pojawiły się w pliku wejściowym. Przykład:

we:

```
alfa = 23
beta = 45
gamma = 65
alfa =10
beta=75
alfa=13
delta=  95
```

wy:

```
beta=110
alfa=46
delta=95
gamma=65
```

Kolejność identyfikatorów na wyjściu - w porządku alfabetycznym.

Ruby-A-5 Plik wejsciowy zawiera w każdym wierszu ciąg liczb całkowitych (dowolnej długości, być może pusty) oddzielonych znakami odstępu lub tabulacji. W niektórych liniach może opcjonalnie pojawić się również komentarz rozpoczynający się znakiem # i rozciągający się do końca linii. Napisać program, który przepisze do pliku wyjściowego tylko te wiersze, w których suma liczb jest większa niż 100 (ewentualnych liczb znajdujących się w komentarzu nie bierzemy pod uwagę). Pozostałe wiersze należy usunąć. Na końcu pliku wyjściowego powinna znaleźć się informacja, ile wierszy usunięto.
Przykład:
we:

```
1 3 4 15
56   22  9000
3 6 # 4000
770   77 # 10 10
```

wy:

```
56   22  9000
770   77 # 10 10
USUNIĘTO: 2
```

Flex/Bison-A-2 [Bez użycia Bisona] Plik wejściowy zawiera tekst, w którym występują m.in. liczby całkowite dodatnie i ujemne. Napisać program, który zastąpi piątą z kolei liczbę całkowita dodatnią - zerem, cały pozostały tekst przepisując bez zmian. Przykład:
we: `Ci mają 5, -1, 10, 900 i -2, A tamci mają 40 i 70 oraz 6 i -2.`
wy: `Ci mają 5, -1, 10, 900 i -2, A tamci mają 40 i 0 oraz 6 i -2.`

Flex/Bison-A-5 Stworzyć parser akceptujący ujęte w nawiasy okrągłe niepuste ciągi zer i jedynek oddzielonych białymi znakami. Wszelkie nadmiarowe białe znaki są ignorowane. Przykłady takich napisów:

a) `(0)`

b) `( 1 )`

c)
```
(1 1  0 1
 0 )
```

Jeśli na wejściu pojawi się znak inny niż cyfra 0,1, nawias okrągły, biały znak powinniśmy dostać komunikat "Błąd leksykalny.". Jeśli budowa napisu będzie nieodpowiednia, powinniśmy dostać komunikat "Błąd składniowy.".

Parser musi być zbudowany zgodnie ze sztuką: flex identyfikuje jednostki leksykalne, bison sprawdza poprawność budowy napisu.

OpenFst/Thrax-A-3 Skonstruować przetwornik, który przetwarza napis w taki sposób, że wymazuje wszystkie litery `a`, pozostały tekst pozostawiając bez zmian. W alfabecie uwzględnić tylko litery `a-e`. Przykład:
we: `dadae`
wy: `dde`
Rozwiązanie: plik thraxowy .grm

OpenFst/Thrax-A-5 Skonstruować przetwornik, który przetwarza napis wejściowy w taki sposób, że usuwa z niego literę jedną literę `e` (przetwornik nie przetwarza napisów wejściowych nie zawierających `e`). W alfabecie uwzględnić tylko litery `a-e`. Przykłady:
`ded` przetwarza na `dd`;
`edede` przetwarza na `dede`, `edde` i `eded`;
dla `abc` daje wynik pusty.

Rozwiązanie powinno zawierać kod thraxowy w pliku z rozszerzeniem .grm ORAZ przetwornik w formacie tekstowym OpenFst (wygenerowany automatycznie lub stworzony ręcznie) w pliku z rozszerzeniem .txt.