Тема 13. Упражнения

В последующих упражнениях, полагайтесь на эти определения:

```
(defrel (student° name group)
  (matche (cons name group)
          [(Дарья     . 2)]
          [(Максим    . 1)]
          [(Кирилл    . 2)]
          [(Александр . 1)]
          [(Владимир  . 2)]))

(defrel (friend° name1 name2)
  (matche (cons name1 name2)
          [(Дарья     . Максим)]
          [(Дарья     . Александр)]
          [(Максим    . Кирилл)]
          [(Максим    . Владимир)]
          [(Кирилл    . Александр)]
          [(Александр . Владимир)]))

(defrel (parent° parent-name child-name)
  (matche (cons parent-name child-name)
          [(Мардж . Барт)]
          [(Мардж . Лиза)]
          [(Мардж . Мэгги)]
          [(Гомер . Барт)]
          [(Гомер . Лиза)]
          [(Гомер . Мэгги)]
          [(Абрахам . Гомер)]
          [(Мона . Гомер)]
          [(Жаклин . Мардж)]
          [(Жаклин . Пэтти)]
          [(Жаклин . Сельма)]
          [(Клэнси . Мардж)]
          [(Клэнси . Пэтти)]
          [(Клэнси . Сельма)]
          [(Сельма . Линг)]))

(defrel (unary° n)
  (conde
   [(== 'z n)]
   [(fresh (m)
           (== `(s ,m) n)
           (unary° m))]))

(defrel (direct-train° from to)
  (let [(ft (cons from to))]
    (matche ft
            [(Мытищи . Химки)]
            [(Люберцы . Мытищи)]
            [(Одинцово . Люберцы)]
            [(Красногорск . Одинцово)]
            [(Балашиха . Красногорск)]
            [(Видное . Балашиха)]
            [(Коломна . Видное)])))
```

Упражнение 13.1 (Простые отношения)**.** Реализуйте следующие простые отношения:

1. Отношение `groupmates°` (одногрупники):

```
(run 1 (q) (groupmates° 'Дарья 'Максим))       ; '()
(run 1 (q) (groupmates° 'Дарья 'Владимир))    ; '(_.0)
```

2. Отношение `relative°` (родственники с общим предком):

```
(run 1 (q) (relative° 'Сельма 'Пэтти))    ; '(_.0)
(run 1 (q) (relative° 'Лиза 'Линг))       ; '(_.0)
(run 1 (q) (relative° 'Лиза 'Сельма))     ; '(_.0)
(run 1 (q) (relative° 'Гомер 'Сельма))    ; '()
```

Упражнение 13.2 (Слова)**.** Определите следующие отношения для списков символов:

1. Отношение `binary°`, определяющее двоичные числа:

```
(run* (x) (binary° `(1 ,x 0)))
; '(0 1)

(run 6 (xs)
     (binary° xs)
     (fresh (ys) (== `(1 . ,ys) xs)))
; '((1) (1 0) (1 1) (1 0 0) (1 1 0) (1 0 1))

(run* (q)
      (fresh (d1 d2)
             (== q `(,d1 ,d2))
             (binary° q)))
; '((0 0) (1 0) (0 1) (1 1))
```

2. Отношение `word°`, определяющее слова в алфавите:

```
(run* (x) (word° '(a b c) `(a ,x c)))
; '(a b c)

(run 1 (x) (word° x '(a b c)))
; '(a b c . _.0)

(run* (x y z) (word° '(a b) `(,x ,y ,z)))
; '((a a a) (a a b) (a b a) (a b b) (b a a) (b a b) (b b a) (b b b))
```

Упражнение 13.3 (Унарные числа)**.** Реализуйте следующие отношения для унарных чисел:

1. Отношение `double°`, позволяющее удвоить число:

```
(run 1 (q) (double° '(s (s z)) '(s (s (s (s z))))) ; '(_.0)
(run 1 (x) (double° '(s (s z)) x))                 ; '((s (s (s (s z)))))
(run 1 (x) (double° x '(s (s (s (s z))))))         ; '((s (s z)))
(run 1 (x) (double° x '(s (s (s z)))))             ; '()
```

2. Отношение `leq°`, позволяющее убедиться, что одно число меньше либо равно другому:

```
(run 1 (q) (leq° '(s (s z)) '(s (s (s z)))))          ; '(_.0)
(run 1 (q) (leq° '(s (s (s (s z)))) '(s (s (s z))))) ; '()

(run 1 (q) (leq° q '(s (s (s z)))))
; '(z (s z) (s (s z)) (s (s (s z))))
```

3. Отношение `mult°`, позволяющее умножать и делить целые унарные числа:

```
(run 1 (x) (mult° '(s (s z)) '(s (s (s z))) x))
; '((s (s (s (s (s (s z)))))))

(run 1 (x) (mult° x '(s (s (s z))) '(s (s (s (s (s (s z)))))))
; '((s (s z)))
```

4. Отношение `power-of-2°`, такое что `(power-of-2° n m)` верно при $m = 2^n$:

```
(run 1 (q) (power-of-2° '(s (s z)) '(s (s (s (s z))))))   ; '(_.0)
(run 1 (x) (power-of-2° x '(s (s (s z)))))                ; '()
(run 1 (x) (power-of-2° '(s z) x))                        ; '((s (s z)))
(run 3 (x y) (power-of-2° x y))
; '((z (s z))
;   ((s z) (s (s z)))
;   ((s (s z)) (s (s (s (s z))))))
```

Подсказка: важно указать верхнюю границу для второго аргумента, например через `leq°`*!*

Упражнение 13.4. Реализуйте следующие отношения на графе путей:

1. Отношение `train°`, определяющее, что существует поезд между двумя станциями:

```
(run 1 (q) (train° 'Люберцы 'Химки))  ; '(_.0)
(run* (q) (train° 'Люберцы q))        ; '(Мытищи Химки)
(run* (q) (train° q 'Люберцы))
; '(Одинцово Красногорск Балашиха Видное Коломна)

(run* (from to) (train° from to))         ; ...
```

2. Отношение `train-path°`, определяющее маршруты между двумя станциями:

```
(run 1 (path) (train-path° 'Люберцы 'Химки path))
; '((Люберцы Мытищи Химки))

(run* (path) (fresh (to) (train-path° 'Люберцы to path)))
; '((Люберцы Мытищи)
;   (Люберцы Мытищи Химки))

(run* (q path) (train-path° q 'Люберцы path))    ; ...
(run* (from to path) (train-path° from to path)) ; ...
```