

# Логическое программирование

Занятие 7 / 7.06.22

# List Comprehension

[ Expression || Generator ]

Generator – nondeterm выражение, которое «генерирует» решения

Expression – то, что собирается в список, используя переменные, которые были конкретизированы в Generator

# List Comprehension

```
student("Peter", 201).  
student("Mike", 202).  
student("Helen", 202).  
student("John", 201).  
student("Anna", 203).
```

```
L = [Group || student(_, Group), Group > 201]  
      yields
```

~~[202, 203]~~ [201, 202, 203]

# List Comprehension

Примеры!

# Предикаты, возвращающие значения

- Предикаты могут «возвращать» значения – вести себя как «классические» функции из других языков
- Точнее:
  - Если процесс доказательства такого предиката заканчивается неуспехом, начинается процесс возврата (backtracking) без возврата конкретного значения этим предикатом
  - Если процесс доказательства заканчивается успехом, он возвращает некоторое значение, и процесс доказательства движется дальше; полученное значение может быть записано в переменную или проброшено в аргумент другого предиката

# Предикаты, возвращающие значения

- То есть предикат как бы возвращает сразу 2 значения:
  - Истинность – удалось ли согласовать его с базой знаний
  - Некоторое конкретное значение – строка, число, структура и т.п.

# Предикаты, возвращающие значения

```
class predicates
  double_int : (integer Number) -> integer Doubled determ (i).
clauses
  double_int(Number) = 2 * Number :-
    1) Number >= 0.
    2) double_int(Number/2) = Number :-
```

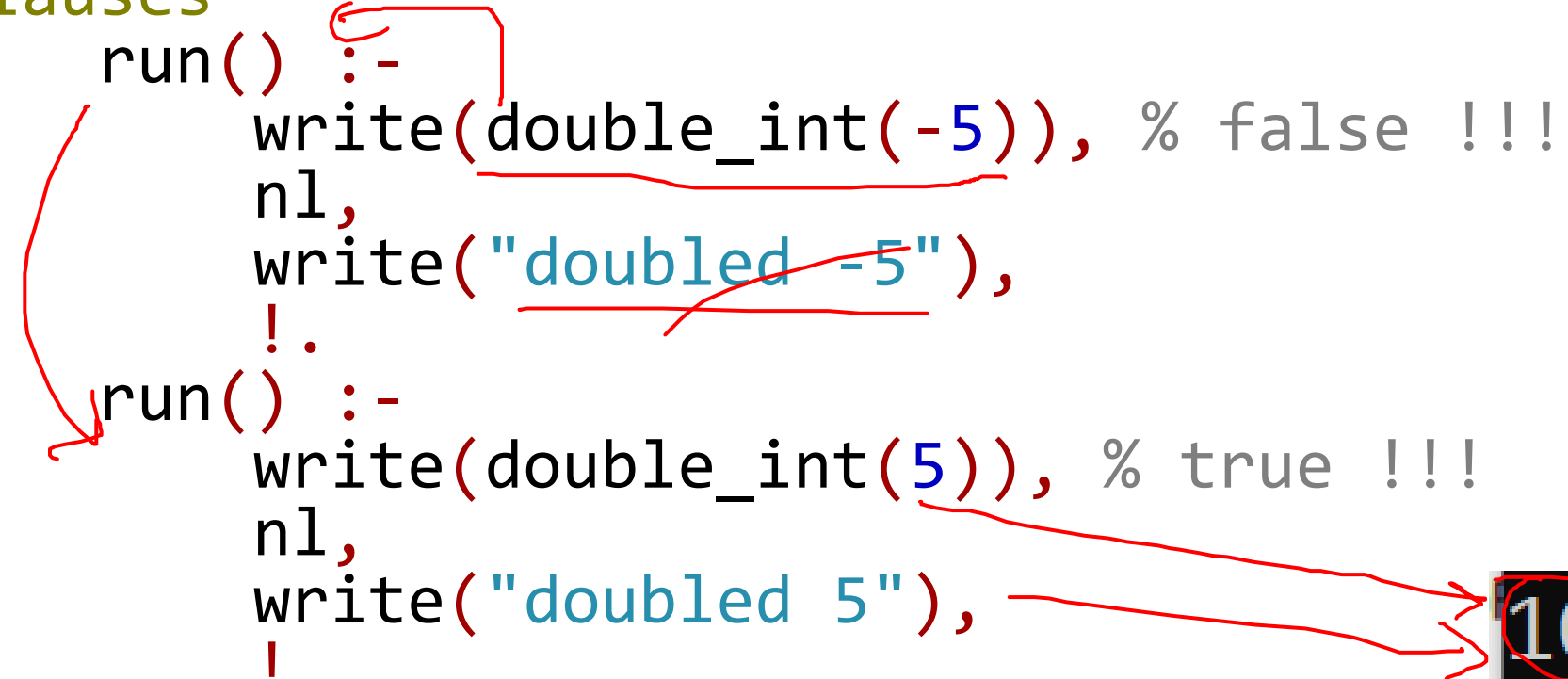
Handwritten red annotations on the code:

- A bracket connects `Doubled` in the type signature to `Doubled` in the clause body.
- A bracket under `determ` is labeled "common".
- A bracket under `double_int(Number/2)` is labeled "2)".
- A bracket under `Number >= 0.` is labeled "1)".

# Предикаты, возвращающие значения

clauses

```
run() :-  
    write(double_int(-5)), % false !!!  
    nl,  
    write("doubled -5"),  
    !.  
run() :-  
    write(double_int(5)), % true !!!  
    nl,  
    write("doubled 5"),  
    !.  
run().
```



```
10  
doubled 5
```



# Предикаты, возвращающие значения

Предикат, который **возвращает длину списка**, который подается на вход:

Как реализовать?

# Предикаты, возвращающие значения

```
class predicates
  length : (A*) -> integer Length procedure (i).
clauses
  length([]) = 0.
  length([_ | T]) = length(T) + 1.
```

# Предикаты, возвращающие значения

```
class predicates
  length : (A*) -> integer.
clauses
  length([]) = 0.
  length([_ | T]) = length(T) + 1.
```

# Предикаты, возвращающие значения

Примеры!

# Лабораторная работа 3

Реализовать необходимые правила, применяя предикаты по работе со списками:

- Функционал List Comprehension при генерации решений в виде списков
- Вывод данных предметной области как поэлементный вывод списка
- Поиск данных предметной области как поиск элементов в списке
- Вычисление количества и максимального / среднего / минимального значений характеристик при помощи обработки списков данных
- Получение значений реализовать через предикаты, возвращающие значения

# Лабораторная работа 3

Пример!

# Любые вопросы по темам курса

Ответы (скорее всего) будут в виде кода занятия