Асирян Александр Камоевич 428

Функция **(visit-doctor name count)** является стартовой и принимает на вход имя и количество пациентов. В ней доктор приветствует пациента и начинает работу.

*Функции, не относящиеся к 7 упражнению:*

**(pick-random lst)**

Выбирает случайный элемент из списка.

**(push lst elem)**

Добавляет элемент в список, если его там не было.

**(random\_with\_weights good sum x)**

Распределение с весами. Сумма – длина всего отрезка, x – случайное число из отрезка [0,1], good – список пар (функция вес). Чем больше вес, тем больше вероятность попадания х в отрезок, соответствующий данной функции.

**(findrep pairs word)**

Ищет совпадение слова word с левой или правой частью в парах замен pairs. При нахождении меняется на противоположную часть.

**(replace pairs lst)**

Изменяет фразу lst, заменяя каждое слово в ней с помощью функции findrep, передавая ей список пар замен pairs.

**(qualifier)**

Выбор одного из стандартных ответов.

**(default responses)**

Обычный ответ, совмещающий некую реплику из qualifier с фразой, измененной с помощью replace. Список пар замен '((i you) (me you) (am are) (my your)).

**(too\_few responses)**

Выбирается фраза, соответствующая тому, что пациент произнес слишком короткую фразу (<2 слов)

**(too\_much responses)**

Выбирается фраза, соответствующая тому, что пациент произнес слишком длинную фразу (>15 слов)

**(hedge responses)**

Выбор одной из фразы, которая не зависит ни от каких параметров.

**(find key)**

Находит первое вхождение ключа из списка key в фразе пациента (содержится в объемлющей функции keyans.

**(keyans keys phrase pr)**

Находит самое приоритетное вхождение ключевых слов из тройки (приоритет (список\_ключевых\_слов) (список \_ответных\_фраз)) в фразе phrase. Возвращает нужную тройку с первым подобранным ключевым словом, найденное с помощью findkeys.

**(answer keys phrase)**

Возвращает измененную фразу, если в ней присутствовало ключевое слово. Замена происходит с помощью функции replace, в качестве пары (\* первое\_ключевое\_слово).

**(repeat responses)**

Возвращает измененную случайную фразу из словаря фраз пациента.

**(reply responses handlers)**

Принимает словарь фраз пациента и функций-обработчиков, отбирает только те функции, которые удовлетворяют условиям. Далее с помощью функции random\_with\_weights выбирается обработчик, на вход которому посылается словарь.

*Все остальное и упражнение 7:*

В функции visit-doctor доктор спрашивает пациента, знает ли тот, кто он (доктор) такой. После двух часов дня врач имеет способность мотивировать пациента как Шайа ЛаБаф ([https://ru.wikipedia.org/wiki/Лабаф,\_Шайа](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D0%B1%D0%B0%D1%84%2C_%D0%A8%D0%B0%D0%B9%D0%B0)) в отрывке из своего мотивационного видео (<http://www.youtube.com/watch?v=ZXsQAXx_ao0>). Если ответ пациента отрицателен и время подходит, то в функцию doctor-driver-loop, которая обрабатывает пациента, передается соответствующий флаг. Иначе доктор заявляет о том, что он такой же простой, как и все остальные.

**(doctor-driver-loop dict shia\_labeouf\_mode)**

Функция обрабатывает пациента, параметр dict отвечает за словарь фраз пациента и изначально пуст, о флаге shia\_labeouf\_mode было оговорено выше.

При обычных обстоятельствах выбирается функция-обработчик и выдается ответ.

К обработчикам добавлена функция **(shia responses)**, которая выбирает случайную фразу из мотивационной речи. Имеет очень большой вес, по сравнению с другими обработчиками.

Но если количество различных фраз у одного пациента превысит 5, доктор начнет злиться и вызывается функция angry\_doctor.

**(angry\_doctor)**

Выбирает фразу, которая передает все эмоции уставшего от пациента врача. Для фраз с неоднозначным переводом приведен цензурный перевод в комментариях к коду.

Если время перевалило за 6 вечера, то врач не просто злой, а очень злой и может так же применить некоторое действие по отношению к пациенту.

**(angry\_phrase)**

Выбор фразы.

**(angry\_move)**

Выбор действия.

Работа с пациентом оканчивается по фразе (goodbye).

Далее вызывается функция **(ask-patient-name)**, которая спрашивает имя очередного пациента и вызывает функцию visit-doctor с новым именем и увеличенным количеством.

Если количество пациентов превысило 5, то врач начинается лениться и у него пропадает желание работать. Он не будет обслуживать пациента в течение 5 минут по личным очень важным причинам. Не стоит спорить с доктором, он все равно будет игнорировать пациента ровно 5 минут. Это реализовано в функции **(go\_away name)**. Когда доктор отдохнет он вызывает пациента.

Окончательное завершение работы происходит при передаче слова “suppertime” в качестве имени очередного пациента.

**Седьмое упражнение сделано исключительно в целях “оригинальной идеи” и не несет смысловой нагрузки.**