UNIVERSIDAD DE SANTIAGO DE CHILE

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INFORMÁTICA

INFORME CHATBOT

Programación en Scheme

Nombre Alumno:	Francisco Guajardo
Profesores:	Daniel Gacitúa
Ayudantes:	Javier Vasquez
Fecha de Entrega:	23 – 04 - 2018

Tabla de contenido

Introducción

En la actualidad muchas empresas y compañías de bienes y servicios implementan en algún momento la entidad chatbot.
El chatbot es una entidad conversacional artificial. Son básicamente programas computacionales que son capaces de mantener o generar una conversación fluida con un usuario.
En este laboratorio se nos encomendó la tarea de realizar un chatbot que logre los estándares de este, basando nuestro algoritmo en el paradigma funcional, usando lenguaje de programación Scheme.
Este chatbot esta diseñado para mantener una conversación corta con un usuario que desea comprar una película, juego o manga de la tienda X, indicando su precio y pidiendo forma de pago al usuario.
Este programa funciona en DrRacket, que es un entorno de desarrollo integrado programado en Racket, que nos facilitará la tarea de programar en Scheme con las herramientas que ofrece este lenguaje.

Marco teórico

- **Lenguaje de programación:** Un lenguaje de programación es un vocabulario, un conjunto de reglas gramaticales que tiene como objetivo dar instrucciones al computador para que este realice tareas especificas.

- **Scheme:** Es un lenguaje de programación de sintaxis simple basado en Lips, que esta débilmente tipado. Se usa como lenguaje del paradigma funcional.

- **Paradigma de programación:** Es un estilo de programar, un camino a seguir, una filosofía para diseñar soluciones a problemas relacionados con la programación.

- **Recursión:** Es el proceso de definir una solución a un problema en términos de ella misma. Teniendo elementos mínimos como condición de parada y llamada recursiva. Existen diferentes tipos de recursión, como la recursión natural, recursión arbórea y recursión de cola.

- **Algoritmo:** Un algoritmo es un procedimiento esquemático que comprende un conjunto de pasos secuenciales ordenados, para realizar una actividad especifica

- **TDA:** Un tipo de dato abstracto consiste en la representación de un objeto, proceso o cualquier elemento real o imaginario para que sea operado por la maquina. Para formar un TDA tenemos que tener en consideración su arquitectura de seis elementos los cuales son Operadores, modificadores, selectores, funciones de pertenencia, constructores y destructores. Como mínimo de esa arquitectura se requiere una función constructora, función selectora y funciones para operar el TDA.

Descripción problema

El problema que nos fue planteado es el de crear un chatbot basado en el paradigma funcional utilizando el lenguaje de programación Scheme.
Para este problema había que diseñar un contexto o temática general para el chatbot, el cual se usaría para guiar y definir las actitudes que este va a tomar con las respuestas que el usuario le entregue, además este contexto se deberá de mantener durante todos los laboratorios.
El chatbot debería ser capaz de poder responder al usuario de forma empática para lograr una mayor conexión con el, este a su vez deberá de contar con líneas de mensajes protocolares y de estilo libre.
Un ejemplo de conversación podría ser el siguiente:

- Chatbot: ¿Cuál de nuestros servicios desea contratar?
- Usuario: cable
- Chatbot: Registrado, ¿Qué hay de internet?
- Usuario: no
- Chatbot: Entendido, ¿y telefonía?
- Usuario: si
- Chatbot: Perfecto, ya he registrado sus servicios

(ejemplo rescatado del enunciado del problema desde USACH Virtual)

Así también, se plantea como problema lograr que nuestro chatbot logre tener memoria de corto y largo plazo, así también poseer capacidad de aprender del contexto y del usuario.
Todo esto debe de estar implementado en base al paradigma funcional, el cual se describirá brevemente en este informe.

Paradigma funcional

El paradigma funcional es una forma de programar basado en la programación declarativa, ya que depende de declaraciones o expresiones en vez de estados, y el uso de funciones matemáticas para realizar y tratar los cálculos de nuestros algoritmos.
La programación funcional nace del calculo lambda, la aplicación de las funciones y la recursión.

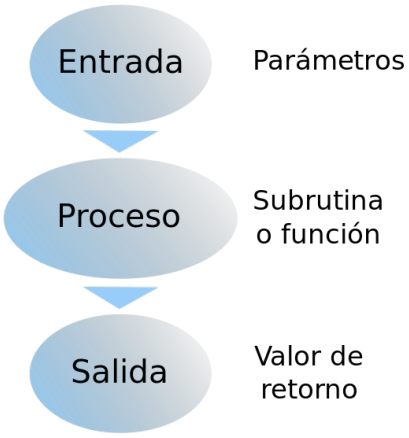


Ilustración 1, Diagrama del funcionamiento de una subrutina.

Al usar la programación funcional, se trata a los valores de salida o retorno de las funciones como valores que dependen solo de los elementos o valores de entrada, así se evitan efectos secundarios, cambios en el estado que no dependen en las entradas de la función.
En este paradigma cabe resaltar que no existen conceptos como el de variable actualizable, iteración o estado. Por lo mencionado anterior se evade este tipo de sentencias con el uso de funciones.
Como esta todo definido en base a cálculos matemáticos, las funciones o subprocesos de la programación funcional, poseen dominio y recorrido fijo.

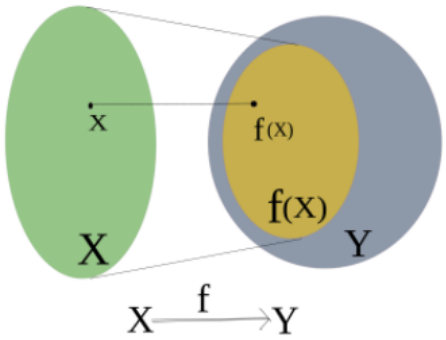


Ilustración 2, Dominio y recorrido.

Análisis del problema

Nos piden utilizar el lenguaje Scheme y no Racket para la realización de este proyecto, por lo cual las funciones que se usaran se limitan a la versión R6RS de Scheme.

Como requerimientos nos solicitan trabajar el chatbot en base a una estructura con distintos parámetros, con la materia vista en clases se puede asumir que este chatbot será una estructura de datos abstracta, TDA para simplificar, la cual constara de una arquitectura de 6 niveles.

También se solicita poder manejar un registro histórico de conversación, sin el uso de variables, ya que se trabaja en paradigma funcional, al no poder usar variables actualizables nos queda la opción de ofrecer como parámetros de una función el retorno de otra, de esta forma solo se usaran funciones para la creación de este registro LOG.

Existe también un elemento llamado *seed* el cual será el encargado de definir el comportamiento de nuestro chatbot con reglas ya predefinidas por el programa.

Para las funciones *endDialog* y *beginDialog* se tiene que usar una forma de pensar similar, puesto que ambas funciones delimitan la conversacion con el user y la “guardan” con un identificador único, ambas deben de usar el Log.

Las funciones *sendMessage* y *rate* necesitan una analogía distinta, pues, en el caso de enviar un mensaje este recibe un parámetro desde el usuario, el cual es desconocido y debe de hacer un tratamiento de este dato para incorporarlo al Log, o en el caso al evaluar el chatbot editar este ultimo, para futuras referencias.

La función *test* es un recopilatorio de todo lo anterior, solo que el mensaje ya esta predicho y solo se utilizar para verificar el buen funcionamiento de nuestro chatbot.

Diseño y aspectos de implementación de la solución

Para el desarrollo de la solución se pensó en base al paradigma funcional, usando la siguiente forma de definir un TDA (posee el TDA base, su función constructora, una función selectora y una función de pertenencia)
En si el proyecto esta basado en funciones R6RS, teniendo que importar librerías como

```
(rnrs lists (6))
(rnrs base (6))
(rnrs io simple (6))
(srfi :19)
(srfi :27)
```

Siendo estas dos ultimas correspondientes a la librería tiempo, y la librería de random respectivamente.
Se parte de la premisa que todo debe de estar basado en programación funcional, se tiene la encapsulación de funciones, esto es lo que el usuario hace al ingresar los comando por pantalla.

```
> (define log1 (beginDialog chatbot log 1))
> (define log2 (sendMessage "The Last Jedi" chatbot log1 1))
> (define log3 (sendMessage "BlueRay" chatbot log2 1))
> (define log4 (sendMessage "Efectivo" chatbot log3 1))
> (define log5 (endDialog chatbot log4 1))
```

Ilustración 3, ejemplo ingreso datos por el usuario

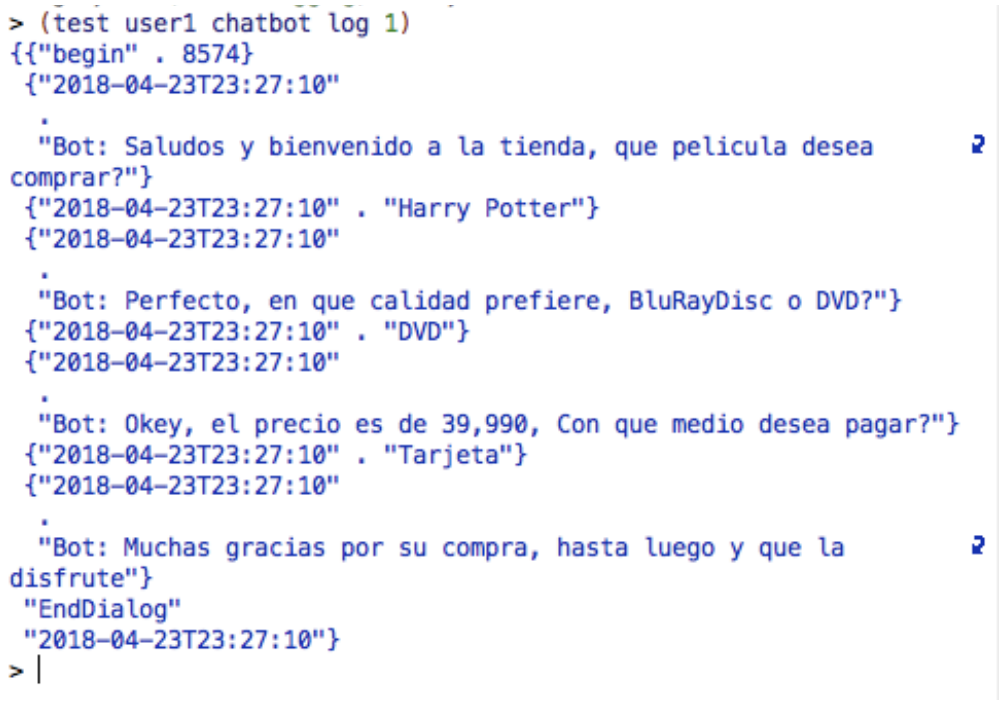
Aquí el usuario ingresa como parámetros el resultado de la función anterior, por lo cual se ahorra el uso de variables que están prohibidas en este desarrollo y se logra una no perdida de datos, logrando hacer una especie de BackTracking bien básico.

Para mantener el Log con sus elementos bien identificados se logro construir una lista de pares, donde cada par es el mensaje con su hora. Cada elemento de esta lista corresponde a una línea de dialogo, comenzando este log con una lista con un par que incluye el identificador del chatbot junto con la palabra clave Begin, que indicara el inicio de la conversación.

La función *sendMessag*e recibirá el log actualizado, y con cada “iteración” que haga el usuario, este log se actualizara, usando el largo de este lograremos “comandar” a nuestro chatbot y le ordenaremos que línea debe de decir en la siguiente actualización.

Resultados

Utilizando la función test llegamos a lo siguiente:



```
> (test user1 chatbot log 1)
{{"begin" . 8574}
 {"2018-04-23T23:27:10"
  .
  "Bot: Saludos y bienvenido a la tienda, que pelicula desea
comprar?"}
 {"2018-04-23T23:27:10" . "Harry Potter"}
 {"2018-04-23T23:27:10"
  .
  "Bot: Perfecto, en que calidad prefiere, BluRayDisc o DVD?"}
 {"2018-04-23T23:27:10" . "DVD"}
 {"2018-04-23T23:27:10"
  .
  "Bot: Okey, el precio es de 39,990, Con que medio desea pagar?"}
 {"2018-04-23T23:27:10" . "Tarjeta"}
 {"2018-04-23T23:27:10"
  .
  "Bot: Muchas gracias por su compra, hasta luego y que la
disfrute"}
 "EndDialog"
 "2018-04-23T23:27:10"}
>
```

Ilustración 4, ejemplo respuesta función test

Lo cual nos indica los limites de nuestro chatbot, el cual solo puede pedir que producto comprar, alguna cualidad de este y que medio de pago el usuario escogerá para comprar, luego se despide.
En si es muy limitado puesto que solo se logra una conversación de 5 a 6 líneas, este no es el resultado esperado, pero tampoco se consideraría un producto final.

Falto implementar la función *rate,* siendo la razón de esto el tiempo y la falta de investigación con respecto al uso correcto del TDA, falto también un orden mayor con lo que respecta al código y una mejor comprensión del enunciado entregado.

Conclusiones

No estando conforme con el resultado final, tampoco se puede decir que no se logro nada, puesto que el chat mantiene una conversación, aunque limitada, con un usuario, siendo esta lo justo y necesario con lo que respecta a la venta de productos, el objetivo de este chatbot.
Las dificultades que se presentaron en el camino recaen la mayoría en el uso del lenguaje Scheme, la limitación de no poder usar variables afecto directamente el desempeño en el presente laboratorio, el acostumbrarse a un lenguaje débilmente tipado y de simple sintaxis no fue tarea fácil.
La no comprensión completa del paradigma también fue un elemento en contra, puesto que, como el programa se basaba en este tipo de programación, no se tenia conocimiento de como plasmar una idea rescatada del paradigma imperativo en base de funciones.

Referencias

1. http://proposicionesdefuncionescalculo.blogspot.cl/2016/11/en-matematicas-logica-una-funcion.html
2. https://es.wikipedia.org/wiki/Programaci%C3%B3n_funcional
3. López, José (21/07/17) <Programación Funcional> http://archive.fo/nqMru
4. http://monads.haskell.cz/html/index.html/html/
5. PTTs Paradigmas (2018) https://drive.google.com/drive/u/1/folders/1HbFEr1f5MCntqecuqNRPfEsoVowyYssy
6. https://groups.csail.mit.edu/mac/projects/scheme/
7. http://cs.lmu.edu/~ray/notes/paradigms/
8. https://users.dcc.uchile.cl/~bebustos/apuntes/cc30a/TDA/
9. Dominio y recorrido. http://proposicionesdefuncionescalculo.blogspot.cl/2016/11/en-matematicas-logica-una-funcion.html
10. Diagrama subrutina: https://commons.wikimedia.org/wiki/File:Subprograma.svg

Tabla de ilustraciones