



Universidad Politécnica
de Madrid



Escuela Técnica Superior de
Ingenieros Informáticos

Máster Universitario en Inteligencia Artificial

Ingeniería Lingüística

Práctica 3: Extracción de información

Alumno 1: Alejandro Francisco Toral

Alumno 2: Enrique Martín López

Madrid, 10 de enero de 2021

Contenido

1.	Introducción	2
1.1.	Elección de textos	2
2.	Análisis de documentos y tabla de dimensiones	3
2.1.	Tabla de dimensiones.....	3
2.2.	Preguntas relacionadas	5
2.3.	Parafraseo de las preguntas.....	6
3.	Reglas de identificación.....	8
4.	Implementación	9
4.1.	Clases.....	9
4.1.1.	Receta.....	9
4.1.2.	Ingrediente	9
4.1.3.	QAModel	10
5.	Ejecución	14
6.	Experimentación y discusión de resultados	15
7.	Trabajo futuro	18
8.	Bibliografía y referencias.....	19

1. Introducción

1.1. Elección de textos

Para la realización de la práctica se han seleccionado dos recetas de cocina. Estas recetas son el pote gallego y el cochinillo asado [1]. Consideramos que contienen la suficiente información como para ser variada e interesante.

Para tratar los textos más fácilmente se ha trasladado toda la información de ambas recetas a dos ficheros txt distintos.

:: Cochinillo asado



Ingredientes (8 personas):

- 300 ccs de agua
- 4 kilos de cochinillo blanco entero
- 2 hojas de laurel
- 100 gramos de manteca de cerdo
- 1 pellizco de sal
- 1 chorreón de vinagre

Nacionalidad: Española
Dificultad: Difícil
Preparación: 300 min.
Vegetariana: No
Nº calorías: Medio

Instrucciones de elaboración:

Pedir en la carnicería que limpien y preparen el cochinillo, quitándole la asadura y practicándole sólo una pequeña abertura en el vientre, lo justo para poderlo vaciar y limpiar.

Salar y coser para que quede bien cerrado.

Calentar el horno a 220-230º centígrados y poner el cochinillo en un recipiente, con las manos hacia arriba y sobre unas tablitas o unos palitos de laurel, colocarlo de manera que se sostenga sin moverse.

Agregar agua y cocer durante 2 horas aproximadamente.

Transcurrido éste tiempo, sacar, darle la vuelta y pinchar con un tenedor para que suelte el agua que haya podido absorber. Barnizar con un poco de manteca de cerdo para que resulte brillante.

Proteger las orejas y el rabo con un poco de papel de aluminio para que no se quemen e introducir de nuevo en el horno 1 hora o más.

En el último momento rociar con el vinagre.

Debe quedar con la corteza dorada y crujiente, de manera que, apretando un poquito con el dedo, chasque como si fuera un pastel de hojaldre.



Figura 1. Captura de la receta del cochinillo asado [1]

:: Pote gallego



Ingredientes (8 personas):

- 150 gramos de cerdo, cabeza (carena) en trozos
- 4 chorizos
- 150 gramos de cordero, falda en trozos
- 750 gramos de garbanzos
- 1 kilo de grelos
- 1 hueso de caña
- 150 gramos de jamón, en trozos
- 300 gramos de patatas
- 150 gramos de tocino fresco en trozos
- 50 ccs de aceite
- 1 pizca de sal

Nacionalidad: Española
Dificultad: Media
Preparación: 180 min.
Vegetariana: No
Nº calorías: Alto

Instrucciones de elaboración:

Se ponen los garbanzos la noche anterior en remojo.

Se pone la carne en una olla con agua fría para desalar la noche anterior.

Se escurren los garbanzos y se ponen en una malla pero sin llenarla, por lo que crecen, para que no se mezcle con el resto de los condimentos.

Se limpian los grelos y se trocean y se ponen en una cacerola con hueso de caña, aceite y sal.

Cuando estén tiernos se retiran del fuego y se escurren y se reservan.

En una cacerola se pone la falda de ternera, el tocino, la careta, el jamón y el chorizo a fuego lento.

Cuando estén casi cocidos, la carne y los garbanzos, se les agrega las patatas enteras y los grelos.

Se deja cocer hasta que todo esté tierno y se sirven en una bandeja sin mezclar los ingredientes.



Figura 2. Captura de la receta del pote gallego [1]

2. Análisis de documentos y tabla de dimensiones

En este apartado se encuentran los resultados del análisis de los dos documentos seleccionados: la tabla de dimensiones, las preguntas principales y las preguntas reformuladas.

2.1. Tabla de dimensiones

La tabla de dimensiones está basada en aquella vista en clase. Se divide en tres columnas:

- **Nombre del concepto:** nombre de la familia de palabras o conceptos de aquellos encontrados en los textos.
- **Valores del concepto:** las instancias de la familia de conceptos.
- **Observaciones:** descripción del apartado y datos a tener en cuenta, como restricciones.

Hemos querido sacar toda la información posible de los documentos. Los apartados de “instrucciones”, “tipo de preparación” e “instrumentos de cocina”, se han obtenido de los pasos de cada rectea.

Nombre del concepto	Valores del concepto	Observaciones
<i>Nombre del plato</i>	Cochinillo Asado, pote gallego	
<i>Ingrediente</i>	300 ccs de Agua, 4 kilos de cochinillo blanco entero, 2 hojas de laurel, 100 gramos de manteca de cerdo, 1 pellizco de sal, 1 chorreón de vinagre, 150 gramos de cerdo, cabeza en trozos, 4 chorizos, 150 gramos de cordero, falda en trozos, 750 gramos de garbanzos, 1 kilo de grelos, 1 hueso de caña, 150 gramos de jamón, en trozos, 300 gramos de patatas, 150 gramos de tocino fresco en trozos, 50 ccs de aceite, 1 pizca de sal	Ingredientes de una receta de cocina
<i>Origen del plato</i>	Gallega, segoviana	Lugar de origen de la receta.
<i>Dificultad</i>	Difícil, media	Nivel de dificultad de la receta, repartidos en “Fácil”, “Media”, “Difícil”
<i>Tiempo de preparación</i>	300 min., 180 min.	Tiempo estimado de la receta. En minutos siempre
<i>Número de Calorías</i>	Medio, alto	Identifica el nivel calórico del plato a preparar.
<i>Plato Vegetariano</i>	No	Identifica si una receta es apta para vegetarianos con un valor binario: “sí”, “no”
<i>Instrumentos de cocina</i>	Horno, recipiente de horno, barniz, papel de aluminio, olla, malla, cacerola, bandeja	Instrumentos utilizados en la preparación de la receta

<i>Tipo de preparación</i>	Cocer, hornear, dejar en remojo", desalar, escurrir, mezclar, limpiar, trocear, escurrir, reservar, cocinar a fuego lento, servir	Acciones específicas en la preparación del plato
<i>Instrucciones de elaboración</i>	<p>Pedir en la carnicería que limpien y preparen el cochinillo, quitándole la asadura y practicándole sólo una pequeña abertura en el vientre, lo justo para poderlo vaciar y limpiar. Salar y coser para que quede bien cerrado.</p> <p>Calentar el horno a 220-230º centígrados y poner el cochinillo en un recipiente, con las manos hacia arriba y sobre unas tablitas o unos palitos de laurel, colocarlo de manera que se sostenga sin moverse.</p> <p>Agregar agua y cocer durante 2 horas aproximadamente.</p> <p>Transcurrido este tiempo, sacar, darle la vuelta y pinchar con un tenedor para que suelte el agua que haya podido absorber.</p> <p>Barnizar con un poco de manteca de cerdo para que resulte brillante.</p> <p>Proteger las orejas y el rabo con un poco de papel de aluminio para que no se quemen e introducir de nuevo en el horno 1 hora o más.</p> <p>En el último momento rociar con el vinagre. Debe quedar con la corteza dorada y crujiente, de manera que, apretando un poquito con el dedo, chasque como si fuera un pastel de hojaldre.</p> <p>Se ponen los garbanzos la noche anterior en remojo.</p> <p>Se pone la carne en una olla con agua fría para desalar la noche anterior.</p>	<p>Frases significativas encontradas en la receta que especifican los pasos de su preparación.</p>

	<p>Se escurren los garbanzos y se ponen en una malla, pero sin llenarla, por lo que crecen, para que no se mezcle con el resto de los condimentos.</p> <p>Se limpian los grelos y se trocean y se ponen en una cacerola con hueso de caña, aceite y sal.</p> <p>Cuando estén tiernos se retiran del fuego y se escurren y se reservan.</p> <p>En una cacerola se pone la falda de ternera, el tocino, la careta, el jamón y el chorizo a fuego lento.</p> <p>Cuando estén casi cocidos, la carne y los garbanzos, se les agrega las patatas enteras y los grelos.</p> <p>Se deja cocer hasta que todo esté tierno y se sirven en una bandeja sin mezclar los ingredientes.</p>	
Número de personas	8	Número de personas estimado que podrán comer del plato preparado

2.2. Preguntas relacionadas

En esta tabla se encuentran las preguntas principales que están relacionadas con los conceptos de la tabla de dimensiones. Estas preguntas se utilizarán como oraciones base que serán reformuladas para obtener más preguntas y dar variedad al programa.

Nombre del concepto	Pregunta Tipo
<i>Nombre del plato</i>	¿Qué recetas hay disponibles?
<i>Ingrediente</i>	¿Qué ingredientes tiene x?
<i>Cantidades</i>	¿Cuáles son las cantidades necesarias para hacer x?
<i>Nacionalidad del plato</i>	¿De dónde procede x?
<i>Dificultad</i>	¿Cuál es la dificultad de hacer x?
<i>Tiempo de preparación</i>	¿Cuánto se tarda en hacer x?
<i>Número de Calorías</i>	¿Cuál es la cantidad de calorías de x?
<i>Plato Vegetariano</i>	¿x es una receta vegetariana?
<i>Herramientas de cocina</i>	¿Cuáles son las herramientas de cocina necesarias para hacer x?
<i>Tipo de preparación</i>	¿Cuáles son los métodos de cocinar empleados al hacer x?
<i>Instrucciones de elaboración</i>	¿Cómo se hace x?
<i>Número de personas</i>	¿Para cuántas personas es la receta de x?

2.3. Parafraseo de las preguntas

En esta tabla se encuentran las preguntas anteriores parafraseadas para tener en cuenta todo tipo de variaciones en la forma de demandar la información. Esta información es fundamental para tener en cuenta todas las variables que el programa se puede encontrar cuando tenga que procesar la información.

Pregunta Tipo	Variaciones
<i>¿Qué recetas hay disponibles?</i>	<ul style="list-style-type: none"> • Nombres de los platos disponibles • Platos disponibles • ¿Cuáles son los platos disponibles?
<i>¿Qué ingredientes tiene x?</i>	<ul style="list-style-type: none"> • ¿Cuáles son los ingredientes de x? • ¿Qué ingredientes lleva x? • Dime la lista de ingredientes de x • Ingredientes de x
<i>¿Cuáles son las cantidades necesarias para hacer x?</i>	<ul style="list-style-type: none"> • ¿Cuántos ingredientes se necesitan para hacer x? • ¿Cuántos gramos de “ingrediente” se necesitan para hacer x? • Cantidades de los ingredientes de x
<i>¿De dónde procede x?</i>	<ul style="list-style-type: none"> • ¿De qué lugar es el plato x? • ¿De dónde es originario x? • Lugar de origen de x • ¿De dónde viene x? • ¿Cuál es la procedencia de x? • ¿Cuál es el lugar de procedencia de x? • Lugar de procedencia de x
<i>¿Cuál es la dificultad de hacer x?</i>	<ul style="list-style-type: none"> • ¿Cuánto de difícil es hacer x? • Dificultad del plato x • ¿Cómo es de difícil hacer x?
<i>¿Cuánto se tarda en hacer x?</i>	<ul style="list-style-type: none"> • Tiempo de preparación de x • ¿Cuánto tiempo tarda en hacerse x?
<i>¿Cuál es la cantidad de calorías de x?</i>	<ul style="list-style-type: none"> • Nivel calórico de x • Nivel de calorías de x • ¿Cuál es el nivel calórico de x? • ¿Cuántas calorías tiene x?
<i>¿x es una receta vegetariana?</i>	<ul style="list-style-type: none"> • ¿x es apto para vegetarianos? • ¿x es un plato vegetariano? • ¿Pueden los vegetarianos tomar x?
<i>¿Cuáles son las herramientas de cocina necesarias para hacer x?</i>	<ul style="list-style-type: none"> • ¿Qué instrumentos se necesitan para hacer x? • ¿Qué utensilios de cocina se necesitan para hacer x? • Instrumentos de cocina para hacer x
<i>¿Cuáles son los métodos de cocinar empleados al hacer x?</i>	<ul style="list-style-type: none"> • ¿Qué métodos culinarios se emplean en x? • ¿Qué técnicas se utilizan en x? • ¿Cuáles son las técnicas culinarias que se utilizan para hacer x?

<i>¿Cómo se hace x?</i>	<ul style="list-style-type: none"> • Pasos para hacer x • ¿Qué pasos hay que seguir para hacer x? • ¿Cuáles son los pasos de la receta de x?
<i>¿Para cuántas personas es la receta de x?</i>	<ul style="list-style-type: none"> • ¿Cuántas personas pueden comer x?

3. Reglas de identificación

Tal y como se especifica en el enunciado, los datos de las recetas están escritos exclusivamente en la manera que aparece en los textos encontrados.

En un principio, dejamos como idea de ampliación de la práctica añadir las variaciones lingüísticas de cada uno de los elementos de la receta, pero debido a la falta de tiempo y sobre todo, a ser un proyecto centrado en dos recetas muy específicas, decidimos no implementar dicha funcionalidad al final.

Dicho esto, las preguntas del usuario devolverán como respuesta las palabras como están, sin ningún tipo de variación, independientemente de cómo el usuario formule la pregunta. A continuación, se expone un ejemplo:

```
Introduzca una pregunta o escriba 'Salir' para cerrar el programa.  
  
¿de dónde proviene el pote gallego?  
Es: Gallega  
  
Introduzca una pregunta o escriba 'Salir' para cerrar el programa.  
  
¿Cuál es la procedencia del pote gallego?  
Es: Gallega  
  
Introduzca una pregunta o escriba 'Salir' para cerrar el programa.
```

Figura 3. Ejemplo de ejecución

Como se puede observar, no hay variabilidad lingüística alguna. El programa, en el caso de la procedencia, trata en femenino cualquier pregunta relacionada con el origen del plato. La respuesta adecuada a la primera pregunta será “de Galicia” o “Es: gallego”, pero sin embargo contesta con la palabra origen del texto: “Es: Gallega”, refiriéndose a la receta.

4. Implementación

El código de la práctica se ha implementado en lenguaje Java y se ha ejecutado utilizando distintos IDEs: Eclipse e IntelliJ IDEA.

La formulación principal a la solución del problema consiste en lo siguiente: una vez se tenga toda la información de los textos procesada e insertada en las estructuras del programa, se propondrán una serie de frases modelos -las correspondientes a las que se encuentran en el apartado de preguntas relacionadas- que servirán como enlace común a todas las preguntas que el usuario introduzca.

El programa lee la pregunta del usuario y busca el nombre de la receta. Cuando encuentre el nombre de la receta, el programa buscará la frase modelo que más se parezca a la pregunta introducida y lanzará por pantalla la respuesta programada.

4.1. Clases

En este apartado se encuentran descritas las clases Java del proyecto. Se han implementado un total de tres clases.

4.1.1. Receta

Clase que define una receta con todas sus propiedades características sacadas de la anterior [tabla de dimensiones](#). Una receta consta de varias características:

- **Nombre:** nombre de la receta
- **Ingredientes:** lista de ingredientes necesarios para hacer la receta. La lista contendrá elementos de la clase Ingrediente que será posteriormente explicada.
- **Nacionalidad:** lugar de origen de la receta
- **Dificultad:** nivel de dificultad de la realización de la receta
- **MinPreparacion:** tiempo que se tarda en realizar una determinada receta (en minutos)
- **CantidadCalorias:** nivel calórico de cada plato
- **EsVegetariano:** booleano que identifica si el plato es apto o no para vegetarianos
- **HerramientasDeCocina:** lista de utensilios que se utilizan en una determinada receta
- **TipoDePreparación:** lista de técnicas de cocinado que se utilizan durante el cocinado del plato
- **InstruccionesDeElaboracion:** lista de las instrucciones de cada receta
- **NumeroDePersonas:** número de personas que pueden comer del plato preparado

La clase Receta consta de sus correspondientes *getters* y *setters* para poder obtener la información e insertar nuevos datos, respectivamente.

4.1.2. Ingrediente

La clase Ingrediente define los elementos de la lista de ingredientes. Un ingrediente estará formado por las siguientes características:

- **Nombre:** nombre completo del ingrediente encontrado en la receta
- **Cantidad:** cantidad necesaria del ingrediente especificada en la receta en sus correspondientes unidades

- **UnidadDeMedida:** unidad en la que la cantidad de un ingrediente concreto estará medida

La clase Ingrediente, al igual que la clase Receta, consta de sus correspondientes *getters* y *setters* para poder obtener la información e insertar nuevos datos, respectivamente.

4.1.3. QAModel

Es la clase principal de nuestro proyecto. Consiste en un modelo básico de inserción y extracción de información.

Al comienzo del código se encuentran definidas las frases modelos. Estas frases aquellas definidas en el apartado de [preguntas relacionadas](#). Sirven para tener una pregunta base con la que relacionar todas aquellas consultas que demanden los usuarios. Todas disponen de la variable *x*, que será el nombre de cualquier plato.

```

26 // Definir las frases modelo
27 // Se usarán para determinar las respuestas clave y redirigir las preguntas que hagan los usuarios
28 private final static String FRASE_NOMBRE = "¿Qué recetas hay disponibles?";
29 private final static String FRASE_INGREDIENTES = "¿Qué ingredientes tiene x?";
30 private final static String FRASE_CANTIDADES = "¿Cuáles son las cantidades necesarias para hacer x?";
31 private final static String FRASE_ORIGEN = "¿De dónde procede x?";
32 private final static String FRASE_DIFICULTAD = "¿Cuál es la dificultad de hacer x?";
33 private final static String FRASE_TIEMPO = "¿Cuánto se tarda en hacer x?";
34 private final static String FRASE_CALORIAS = "¿Cuál es la cantidad de calorías de x?";
35 private final static String FRASE_VEGETARIANA = "¿x es una receta vegetariana?";
36 private final static String FRASE_HERRAMIENTAS = "¿Cuáles son las herramientas de cocina necesarias para hacer x?";
37 private final static String FRASE_METODOS = "¿Cuáles son los métodos de cocinar empleados al hacer x?";
38 private final static String FRASE_INSTRUCCIONES = "¿Cómo se hace x?";
39 private final static String FRASE_PERSONAS = "¿Para cuántas personas es la receta de x?";
40

```

Fragmento de código 1. Preguntas tipo

Seguidamente se inicializan todas las recetas. Mediante la creación de las clases anteriormente mencionadas, definimos todos los platos. A continuación, se muestra un fragmento de código donde se definen el nombre, la dificultad, las herramientas de cocina y se determina si es vegetariano o no:

```

44 // Inicializar recetas
45 // Cochinitillo asado
46 Receta cochinilloAsado = new Receta();
47 cochinilloAsado.setNombre("Cochinillo asado");
48 cochinilloAsado.setDificultad("Difícil");
49 cochinilloAsado.setEsVegetariano(false);
50 List<String> herramientasCocina = new LinkedList<>();
51 herramientasCocina.add("Horno");
52 herramientasCocina.add("Recipiente de horno");
53 herramientasCocina.add("Barniz");
54 herramientasCocina.add("Papel de aluminio");
55 cochinilloAsado.setHerramientasDeCocina(herramientasCocina);

```

Fragmento de código 2. Ejemplo de inicialización del cochinillo asado

Después de introducir las recetas completas en la estructura del programa, se añaden una serie de palabras clave a cada una de las preguntas tipo o preguntas claves ya inicializadas. Se trata de una batería de palabras para poder tener en cuenta cualquier variación que el usuario pueda hacer a una misma pregunta clave. Así pues, por ejemplo, cuando el usuario pregunte: “¿de dónde viene el cochinillo asado?”, las palabras “dónde” y “viene” estarán asociadas a la pregunta tipo “¿de dónde procede x?”, por lo que se devolverá al usuario el lugar de origen del cochinillo asado. De esta forma no tenemos que programar cada una de las frases variantes, sino únicamente una batería de palabras relacionadas.

```

160 // Palabras clave de FRASE_ORIGEN
161 keywordList = new LinkedList<>();
162 keywordList.add("de donde");
163 keywordList.add("sitio");
164 keywordList.add("lugar");
165 keywordList.add("region");
166 keywordList.add("pais");
167 keywordList.add("origen");
168 keywordList.add("originario");
169 keywordList.add("procedencia");
170 keywordList.add("procede");
171 keywordList.add("viene");
172 keywords.put(FRASE_ORIGEN, keywordList);

```

Fragmento de código 3. Ejemplo de lista de palabras clave relacionadas con la pregunta tipo "¿de dónde proviene x?"

Después de procesar los datos, se inicia el programa donde el usuario podrá introducir una pregunta o escribir la palabra “salir” para finalizar la ejecución. Cada vez que el usuario inicie el programa verá en su pantalla la lista de recetas disponibles por las que puede preguntar. Así se le proporciona la información necesaria si tener que preguntar por las recetas disponibles. Si quiere volver a conocer las recetas, basta por preguntar al programa por qué recetas hay disponibles.

```

283 System.out.println("Introduzca una pregunta sobre alguna receta disponible o escriba 'Salir' para cerrar el programa.");
284 for (Receta rec : recetas) { r += rec.getNombre() + ", "; }
285 r = r.substring(0, r.length()-2);
286 System.out.println("Recetas disponibles: "+r);
287 System.out.println();

```

Fragmento de código 4. Frases de la primera ejecución

Cuando el usuario pregunte por consola, la frase se escanea, normalizando todos sus elementos (esto es, quitando signos de puntuación, tildes...).

```

276 // Escaneo de la pregunta del usuario
277 //Parafraseo
278 String cadenaNormalize = Normalizer.normalize(query.toLowerCase().trim().replaceAll( regex: "\\?", replacement: "" ).replaceAll( regex: "¿", replacement: "" ), Normalizer.Form.NFD);
279 query = cadenaNormalize.replaceAll( regex: "[^\\p{ASCII}]", replacement: "" );
280
281 modelQuery = getFraseTipo(query);
282 x = getRecetaQuery(query);
283 answer = "";
284 //Enlace de las preguntas tipo con los datos
285 Receta receta = getReceta(x);

```

Fragmento de código 5. Escaneo y normalizado de las palabras de la frase

Cuando la frase es escaneada y normalizada se llama a la función *getFraseTipo(query)*, pasándole como argumento la frase recién leída. Esta función sencillamente devuelve la frase tipo que más se parezca a la frase introducida. Mediante un contador y comparando con la lista de palabras clave de cada frase tipo, dicho contador aumenta en 1 cada vez que encuentra una palabra coincidente. La frase con la que más palabras coincida será devuelta. En el fragmento de código 5 se encuentra una captura del código.

```

365 // Compara la frase obtenida mediante el escaneo con la frase tipo y devuelve aquella que más se parezca
366 private static String getFraseTipo(String query) {
367     String res = "";
368     int maxVal = 0;
369     // Contador de coincidencias
370     int counter = 0;
371     // Se devuelve aquella frase que más coincidencias tenga
372     for(String fraseTipo : keywords.keySet()) {
373         counter = 0;
374         for(String palabraAComprobar : keywords.get(fraseTipo)) {
375             if(query.contains(palabraAComprobar)) {
376                 counter++;
377             }
378         }
379         if(counter > maxVal && counter != 0) {
380             res = fraseTipo;
381             maxVal = counter;
382         }
383     }
384     return res;
385 }

```

Fragmento de código 6. getFraseTipo()

Después de obtener la frase tipo a a partir de la pregunta del usuario, es el turno de conocer de qué plato se está haciendo la pregunta. Para eso tenemos la función *getRecetaQuery(query)*, que nos devolverá el nombre de la receta cuando encuentre alguna palabra relacionada con ella.

```

387 // Comprueba y devuelve el nombre de la receta incluida en la pregunta
388 private static String getRecetaQuery(String query) {
389     if (query.contains("pote gallego") || query.contains("pote"))
390         return "pote gallego";
391     else if (query.contains("cochinillo asado") || query.contains("cochinillo"))
392         return "cochinillo asado";
393     else
394         return "";
395 }
396

```

Fragmento de código 7. getRecetaQuery()

Cuando obtengamos el nombre de la receta por la que le usuario pregunta, se llama a *getReceta(nombre_receta)* para obtener todos los datos del plato. La función busca la receta con el nombre que le corresponda y la devuelve.

```

387 // Comprueba y devuelve el nombre de la receta incluida en la pregunta
388 private static String getRecetaQuery(String query) {
389     if (query.contains("pote gallego") || query.contains("pote"))
390         return "pote gallego";
391     else if (query.contains("cochinillo asado") || query.contains("cochinillo"))
392         return "cochinillo asado";
393     else
394         return "";
395 }
396

```

Fragmento de código 8. getReceta()

Cuando tengamos todos los datos, una estructura switch-case ayudará a determinar cuál es la respuesta adecuada para caso y se devolverá por pantalla. El programa no finaliza hasta que el usuario no lo especifique con la palabra “salir”, por lo que puede seguir preguntando cosas.

```

286 // Cada caso busca los datos correspondientes dependiendo de cada receta y los saca por pantalla con una frase predeterminada
287 if(receta!=null) {
288     switch(modelQuery) {
289         case FRASE_NOMBRE:
290             for (Receta rec : recetas) {
291                 answer += rec.getNombre() + ", ";
292             }
293             answer = answer.substring(0, answer.length()-2); break;
294
295         case FRASE_INGREDIENTES:
296             for (Ingrediente ingrediente : receta.getIngredientes()) {
297                 answer += ingrediente.getNombre() + "\n";
298             }
299             break;
300
301         case FRASE_CANTIDADES:
302             for (Ingrediente ingrediente : receta.getIngredientes()) {
303                 answer += ingrediente.getCantidad() + " " + ingrediente.getUnidadDeMedida() + " de " + ingrediente.getNombre() + "\n";
304             }
305             break;
306
307         case FRASE_ORIGEN:
308             answer = "Nacionalidad: " + receta.getNacionalidad(); break;
309
310         case FRASE_DIFICULTAD:
311             answer = "Dificultad: " + receta.getDifficultad(); break;
312
313         case FRASE_TIEMPO:
314             answer = "Tiempo de preparación: " + receta.getMinPreparacion() + " minutos."; break;
315
316         case FRASE_CALORIAS:
317             answer = "Cantidad de calorías: " + receta.getCantidadCalorias(); break;

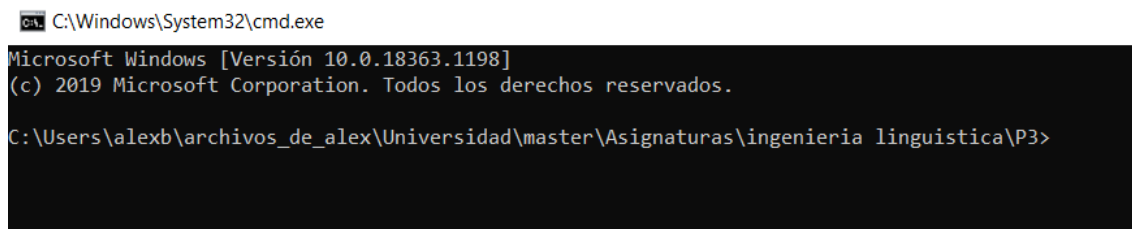
```

Fragmento de código 9. Fragmento del switch-case del programa

5. Ejecución

El programa se ha convertido en un archivo jar para una ejecución más sencilla del mismo. Dentro de la carpeta entregada se encuentra el archivo jar compilado listo para su ejecución. Pasos:

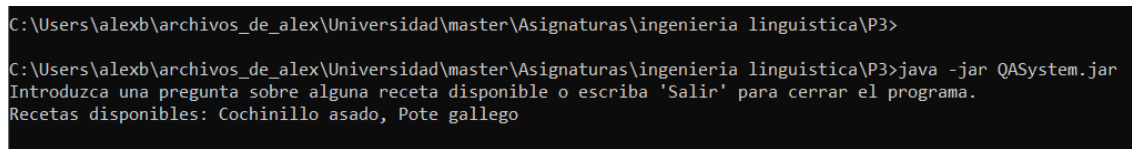
- Abrir la consola de comandos desde la carpeta del proyecto, donde está ubicado el archivo QASystem.jar.



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Versión 10.0.18363.1198]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.
C:\Users\alexb\archivos_de_alex\Universidad\master\Asignaturas\ingenieria linguistica\P3>
```

Figura 4. Ubicación del archivo en el PC del alumno

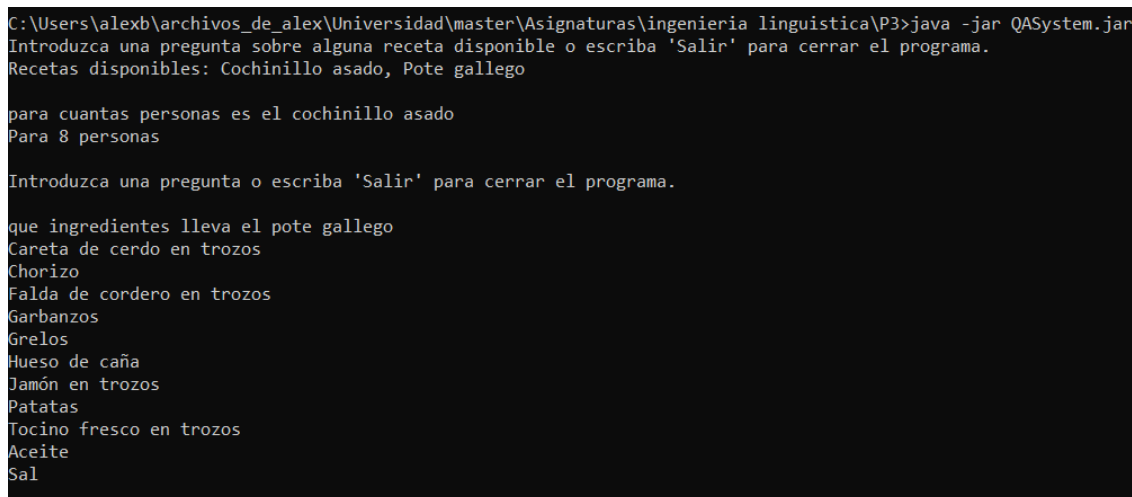
- Ejecutar el archivo mediante la secuencia de comandos “java -jar QASystem.jar”



```
C:\Users\alexb\archivos_de_alex\Universidad\master\Asignaturas\ingenieria linguistica\P3>java -jar QASystem.jar
Introduzca una pregunta sobre alguna receta disponible o escriba 'Salir' para cerrar el programa.
Recetas disponibles: Cochinitillo asado, Pote gallego
```

Figura 5. Cómo ejecutar el programa

Comenzará el programa y se podrá preguntar lo que se desee.



```
C:\Users\alexb\archivos_de_alex\Universidad\master\Asignaturas\ingenieria linguistica\P3>java -jar QASystem.jar
Introduzca una pregunta sobre alguna receta disponible o escriba 'Salir' para cerrar el programa.
Recetas disponibles: Cochinitillo asado, Pote gallego

para cuantas personas es el cochinitillo asado
Para 8 personas

Introduzca una pregunta o escriba 'Salir' para cerrar el programa.

que ingredientes lleva el pote gallego
Carena de cerdo en trozos
Chorizo
Falda de cordero en trozos
Garbanzos
Grellos
Hueso de caña
Jamón en trozos
Patatas
Tocino fresco en trozos
Aceite
Sal
```

Figura 6. Ejemplo de la ejecución del programa

6. Experimentación y discusión de resultados

La fase de experimentación comenzó tras desarrollar la primera versión final del código. Consistió en la ejecución repetida del programa, insertándole primeramente las preguntas base o preguntas tipo encontradas en el apartado de [preguntas relacionadas](#). Esta primera ronda de ejecuciones, hubo respuestas que no casaban con las preguntas, por ejemplo, cuando se pregunta el tiempo de preparación, devuelve la lista de ingredientes con sus cantidades. Esto es por la falta de variedad dentro de los diccionarios de cada pregunta clave: el contador determinaba que las coincidencias de palabras entre una frase y otra es la misma, por lo que se envía directamente el primer resultado encontrado. La solución fue añadir más palabras clave.

```
¿Cuánto se tarda en hacer el cochinitillo asado?
```

```
300 ccs de Agua  
4 kilos de Cochinitillo Blanco Entero  
2 hojas de Laurel  
100 gramos de Manteca De Cerdo  
1 pellizco de Sal  
1 chorreón de Vinagre
```

```
¿Cuánto se tarda en hacer el cochinitillo asado?
```

```
Tiempo de preparación: 300 minutos.
```

A continuación, se encuentran capturas de las ejecuciones exitosas del programa después de hacer los primeros cambios:

¿Qué recetas hay disponibles?

Cochinillo asado, Pote gallego

Introduzca una pregunta o escriba 'Salir' para cerrar el programa.

¿Qué ingredientes tiene el cochinillo asado?

Agua

Cochinillo Blanco Entero

Laurel

Manteca De Cerdo

Sal

Vinagre

Introduzca una pregunta o escriba 'Salir' para cerrar el programa.

¿Cuáles son las cantidades necesarias para hacer el pote gallego?

150 gramos de Careta de cerdo en trozos

4 unidades de Chorizo

150 gramos de Falda de cordero en trozos

750 gramos de Garbanzos

1 kilo de Grellos

1 unidad de Hueso de caña

150 gramos de Jamón en trozos

300 gramos de Patatas

150 gramos de Tocino fresco en trozos

50 ccs de Aceite

1 pizca de Sal

Introduzca una pregunta o escriba 'Salir' para cerrar el programa.

¿De dónde procede el cochinillo asado?

Es: Segoviana

Introduzca una pregunta o escriba 'Salir' para cerrar el programa.

¿Cuál es la dificultad de hacer el pote gallego?

Dificultad: Media

¿Cuanto se tarda en hacer el cochinillo asado?

Tiempo de preparación: 300 minutos.

Introduzca una pregunta o escriba 'Salir' para cerrar el programa.

¿Cuál es la cantidad de calorías del pote gallego?

Cantidad de calorías: Media

Introduzca una pregunta o escriba 'Salir' para cerrar el programa.

¿El cochinillo asado es una receta vegetariana?

No es un plato vegetariano

Introduzca una pregunta o escriba 'Salir' para cerrar el programa.

¿Cuáles son las herramientas de cocina necesarias para hacer el pote gallego?

Olla

Malla

Cacerola

Bandeja

Introduzca una pregunta o escriba 'Salir' para cerrar el programa.

¿Cuáles son los métodos de cocinar empleados al hacer el cochinillo asado?

Hornear, Cocer

```
¿Que se hace al ante salmón?
Paso 1: Se ponen los garbanzos la noche anterior en remojo.
Paso 2: Se pone la carne en una olla con agua fría para desalar la noche anterior.
Paso 3: Se escurren los garbanzos y se ponen en una malla pero sin llenarla, por lo que crecen, para que no se mezcle con el resto de los condimentos.
Paso 4: Se limpian los grelos y se trocean y se ponen en una cacerola con hueso de caña, aceite y sal.
Paso 5: Cuando estén tiernos se retiran del fuego y se escurren y se reservan.
Paso 6: En una cacerola se pone la falda de ternera, el tocino, la careta, el jamón y el chorizo a fuego lento.
Paso 7: Cuando estén casi cocidos, la carne y los garbanzos, se les agrega las patatas enteras y los grelos.
Paso 8: Se deja cocer hasta que todo esté tierno y se sirven en una bandeja sin mezclar los ingredientes.

Introduzca una pregunta o escriba 'Salir' para cerrar el programa.

¿Que muchas palabras es el cambalillo verde?
Para 8 personas
```

La segunda parte de la experimentación consistió en realizar preguntas aleatorias relacionadas con las preguntas tipo, todas ellas redactadas en el apartado de [parafraseo de preguntas](#). En un primer lugar, la implementación de este apartado iba a ser algo diferente: en lugar de preparar una batería de conceptos relacionada con las preguntas base, tendríamos una serie de preguntas enteras relacionadas con cada una de las preguntas base. Pensamos que esto último carecería de sentido ya que los usuarios pueden preguntar de muchas formas distintas y puede que alguna no la tuviéramos en cuenta. Por ello, optamos por la primera opción.

Después de haber añadido las suficientes palabras a la batería de palabras clave, se probó el programa con las preguntas parafraseadas. Al haber una gran variedad de palabras, todas las ejecuciones resultaron exitosas y conseguimos que le programa funcionara correctamente con cualquier variación de las preguntas tipo.

7. Trabajo futuro

Pensamos que el programa está demasiado centrado en los dos textos. Como trabajo futuro se propone un analizador de recetas en general, donde se procesen a mano los textos a un formato que pueda ser fácilmente legible para un programa. Tras leer la información, esta sería procesada y guardada en estructuras de datos parecidas a las ya implementadas.

En cuanto a la definición de reglas de identificación, se añadirían a mano todas las variaciones lingüísticas de todas las palabras procesadas para hacer un extractor de información más variado.

Como mejora de la práctica, se pueden añadir preguntas específicas sobre ingredientes que lleva una receta en concreto. Por ejemplo, en lugar de preguntar por los ingredientes siempre que se quiera saber la cantidad de uno sólo, se podía implementar una función para cada ingrediente en concreto.

8. Bibliografía y referencias

[1] https://www.lupaonline.com/media/recetas/Cocina_espanola.pdf