Universidad Carlos III

Desarrollo de Software

Ejercicio Final Extraordinario

Curso 2021-22

Práctica Final Extraordinaria

FECHA: **28/05/22** - Versión: **3**

GRUPO:**82** EQUIPO: **16**

JUAN DIEGO GOMEZ LABAJOS

100431310

INDICE

Introducción

En la siguiente memoria, se realiza el ejercicio de la práctica extraordinaria final de la asignatura de Desarrollo de software. Cómo

Valores límite y clases de equivalencia

Las técnicas de valores límite y clases de equivalencia sólo son aplicables en esta práctica a los parámetros que posee el fichero de entrada: *date_signature*, reactivation_type y *NewDate*:

Para *date_signature* obtenemos 3 clases de equivalencia:

- Cadena en hexadecimal de 64 caracteres (válido) → TEST_EC_ID_V1
- Cadena no hexadecimal (no válido) → TEST_EC_ID_NV1
- date_signature no es un string (no válido) → TEST_EC_ID_NV2

Y obtenemos 3 casos de valores límite a probar:

- Cadena en hexadecimal de 63 caracteres (no válido) → TEST_LV_ID_NV2
- Cadena en hexadecimal de 64 caracteres (válido) → TEST_LV_ID_V1
- Cadena en hexadecimal de 65 caracteres (no válido) → TEST_LV_ID_NV1

Para reactivation_type obtenemos 4 clases de equivalencia:

- reactivation_type = “Normal” (válido) → TEST_EC_CT_V1
- reactivation_type = “NewDate” (válido) → TEST_EC_CT_V2
- reactivation_type = tiene un valor distinto de los dos anteriores (no válido) →TEST_EC_CT_NV1
- reactivation_type no es un string (no válido) → TEST_EC_CT_NV2

Como este parámetro sólo admite dos valores concretos, no posee ningún valor límite que comprobar.

Para *NewDate* obtenemos 5 clases de equivalencia:

- NewDate == fecha existente cancelada formato ISO (válido) → TEST_EC_R_V1
- NewDate == no ISO (no válido) → TEST_EC_R_NV1
- NewDate == fecha no cancelada
- NewDate == fecha pasada
- NewDate == fecha inexistente

Y no obtenemos casos de valores límite que comprobar.

Tenmos casos de equivalencia de salida:

El fichero de entrada tiene algún problema de formato o de acceso.
La cita que se quiere reactivar no existe.
La cita que se quiere reactivar ya ha caducado o ya ha sido vacunado y, por tanto, no se puede reactivar.
La clave que se quiere reactivar se había desactivado previamente como final y no puede ser reactivada.

Algunos de estos casos ya están comprobados en otros test.

Árbol de derivación y análisis de sintaxis

El fichero de entrada posee la siguiente estructura:

```
{
    "date_signature": "<String having 64 hex. characters>",
    "reactivation_type":"<Normal|NewDate >",
    "new_date":"YYYY-MM-DD"
}
```

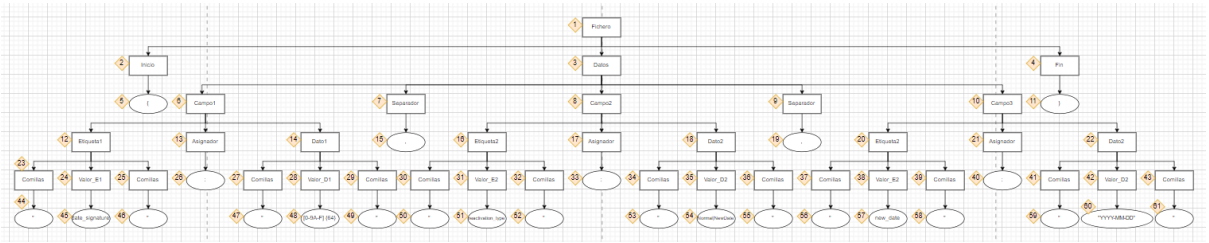
A partir de esta estructura obtenemos la gramática con las siguientes reglas:

(Los nodos terminales son los de color azul)

- Fichero → Inicio Datos Fin
- Inicio → {
- Fin → }
- Datos → Campo1 Separador Campo2 Separador Campo3 Separador
- Separador → ,
- Campo1 → Etiqueta1 Asignador Dato1
- Asignador → :
- Etiqueta1 → Comillas Valor_E1 Comillas

- Comillas → “
- Valor_E1 → date_signature
- Dato1 → Comillas Valor_D1 Comillas
- Valor_D1 → a|b|c|d|e|f|0|1|2|3|4|5|6|7|8|9 (64)
- Campo2 → Etiqueta2 Asignador Dato2
- Etiqueta2 → Comillas Valor_E2 Comillas
- Valor_E2 → reactivation_type
- Dato2 → Comillas Valor_D2 Comillas
- Valor_D2 → Normal|NewDate
- Campo3 → Etiqueta3 Asignador Dato3
- Etiqueta3 → Comillas Valor_E3 Comillas
- Valor_E3 → new_date
- Dato3 → Comillas Valor_D3 Comillas
- Valor_D3 → "YYYY-MM-DD"

Con esta gramática, si la representamos en un árbol, obtenemos el siguiente árbol de derivación (se añade el fichero “Árbol de gramatica.png” junto a esta memoria para facilitar su visionado):



(Los nodos rectangulares son no terminales, y los redondeados son terminales)

Para obtener los casos de prueba, borramos, duplicamos y modificamos cada nodo para obtener un distinto caso de prueba. Cabe recordar que no es necesario modificar los nodos no terminales, así como tampoco es necesario duplicar ni borrar los nodos que generen un caso ya probado.

Gráficos de flujo de control

Analizando el código del método reactivate_appointment:

En este análisis no tenemos en cuenta la función *get_appointment_from_date_signature*, dado que es parte del código que se nos ha porporcionado de base.

```
def reactivate_appointment(self, input_file):
    """Reactivate the cancelled dates and can change its date"""
    appointment = VaccinationAppointment.reactivate_appointment_from_
    return appointment.date_signature
```

Nodo 1 en reactivate_appointment→ Llamada a reactivate_appointment_from_json_file

```
@staticmethod
def reactivate_appointment_from_json_file(json_file):
    """Check is appointment is cancelled and reactivates"""
    reactivation_key = ReactivateJsonParser(json_file).json_content
    appointment = VaccinationAppointment. \
        get_appointment_from_date_signature(reactivation_key[ReactivateJsonParser.DATE_SIGNATURE_KEY])
    appointment.reactivate(reactivation_key[ReactivateJsonParser.REACTIVATION_KEY], reactivation_key[ReactivateJsonParser.NEW_DATE_KEY])
    return appointment
```

Nodo 2 en ReactivateJsonParser

```
class ReactivateJsonParser(JsonParser):
    """Subclass of JsonParer for parsing inputs of get_v
    BAD_REACTIVATION_TYPE_ERROR = "Bad label reactivatio
    BAD_DATE_SIGNATURE_LABEL_ERROR = "Bad label date_sig
    DATE_SIGNATURE_KEY = "date_signature"
    REACTIVATION_KEY = "reactivation_type"
    NEW_DATE_KEY = "new_date"
    BAD_REACTIVATION_TYPE_ERROR = "JSON Decode Error - W
    BAD_DATE_SIGNATURE_LABEL_ERROR = "JSON Decode Error
    BAD_NEW_DATE_LABEL_ERROR = "JSON Decode Error - Wron

    _JSON_KEYS = [DATE_SIGNATURE_KEY,
                  REACTIVATION_KEY,
                  NEW_DATE_KEY]
    _ERROR_MESSAGES = [_BAD_DATE_SIGNATURE_LABEL_ERROR,
                       BAD_REACTIVATION_TYPE_ERROR,
                       BAD_NEW_DATE_LABEL_ERROR ]
```

Nodo 3 en get_appointment_from_date_signature que viene de codigo base

Nodo 4 es reactivate

```
def reactivate(_self, reactivation_type, newdate ):
    """reactivate the appoinmtent"""
    today = datetime.today().date()
    #cambiamos formato a timestamph
    newdate = datetime.fromisoformat(newdate).date()
    if self.__cancelled != "Final":
        today = datetime.today().date()
        if reactivation_type == "Normal":
            date_patient = datetime.fromtimestamp(self.__appointment_date).date()
            if today < date_patient:
                self.__cancelled = "active"
                self.__reason_to_cancel = ""
                self.update_appointment()
            else:
                raise VaccineManagementException("The appointment cannot be retrieved today")
        else:
            if today < newdate:
                #newdate = datetime.fromtimestamp(newdate)
                self.__appointment_date = newdate
                self.__cancelled = "active"
                self.__reason_to_cancel = ""
                self.update_appointment()
    else:
        raise VaccineManagementException("Vaccine not temporally cancelled")
```

Nodo 5 en today (extrae fecha actual)

Nodo 6 en cambio de formato de newdate

Nodo 7 checkeo de cancelación-

Nodo 8 checkeo de reactivation_type–

Nodo 9 date_patient

Node 10 checkeo de condicion—

Node 11 actualización variables

Node 12 Excepcion—

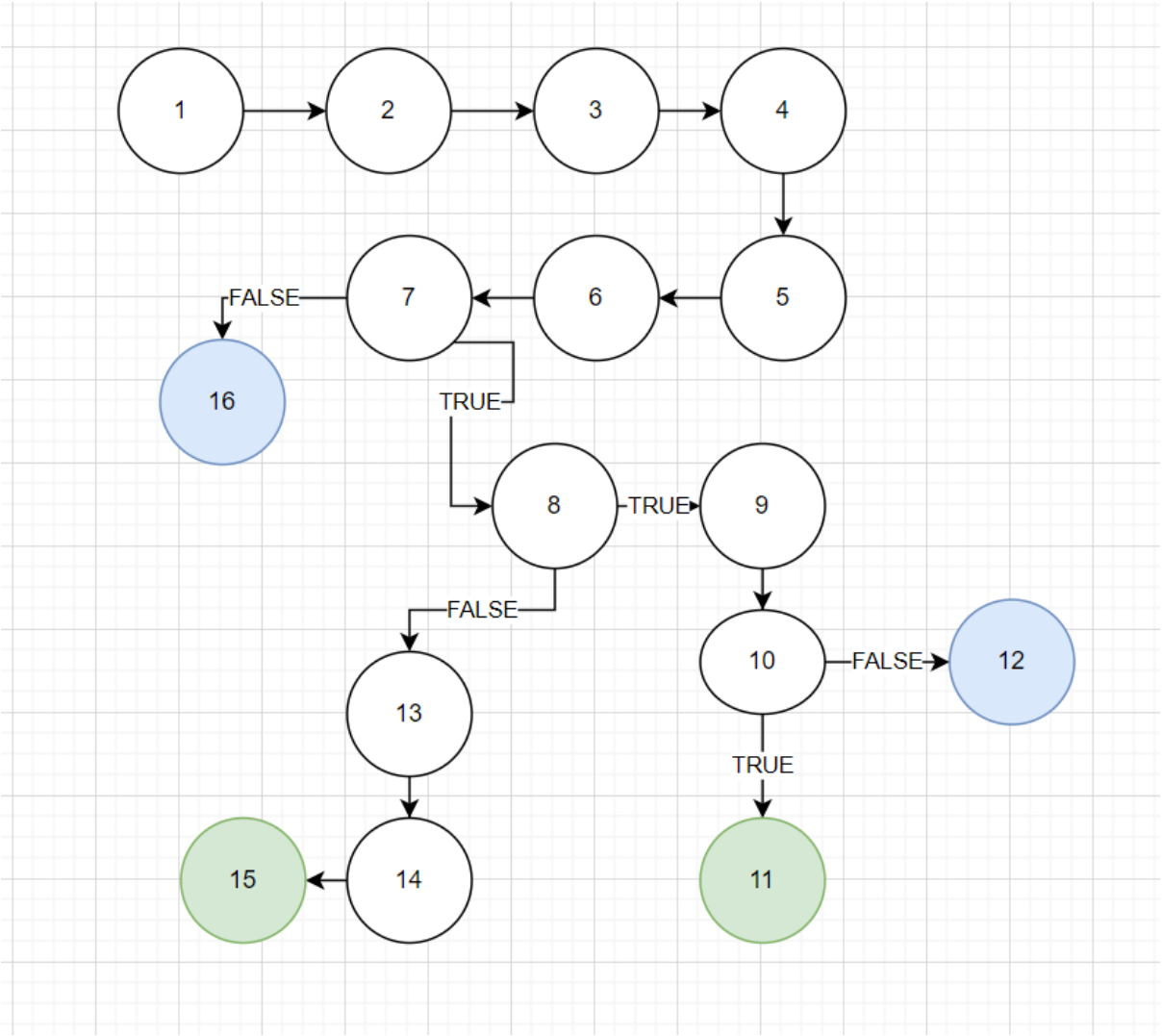
Node13 else–

Node 14 checkeo condicion newdate

Node 15 actualización variables

Node 16 excepción-

De esta forma, obtenemos el siguiente grafo de flujo de control:



Complejidad 2

De esta forma, obtenemos 2 caminos a probar en los tests:

- 1,2,3,4,5,6,7,8,9,10,11
- 1,2,3,4,5,6,8,13,14,15

Ambos son los caminos correctos, excluyendo los caminos que llevan a excepciones. Que son:

- 1,2,3,4,5,6,7,16
- 1,2,3,4,5,6,7,8,9,10,12

Llevaremos tales a los test de comprobación en en fichero: test_reactivate_appointment. Conseguimos por lo menos 5 test

Aclaraciones

Durante la realización de pa práctica, he encontrado muchos problemas con github que he debido soulucionar. No he conseguido pasar todos los test debido a errores que desconocía sobre todo con los de NewDate, me gustaría asistir a una revisión para comprobarlo. No he metido todos los json que debía por tiempo. Creo que me he esforzado bastante y trabajado acorde a mis capacidades. No he podido refactorizar nombres ni adaptar a formato pylint de gran manera. Me ha enseñado bastante y creo que la funcionalidad está conseguida.