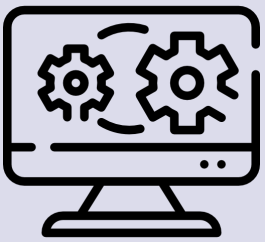
CASOS DE PRUEBA EG2



UNIVERSIDAD CARLOS III DE MADRID

Oscar Junhao Qiu Lin - 100516265

Josué David Hernández Arauz - 100557973

Grupo 07

Desarrollo de Software

2º Grupo.85

uc3m | Universidad Carlos III de Madrid

Índice

Introducción

En este segundo proyecto EG2, se nos ha solicitado definir y documentar los casos de prueba para tres funciones específicas. Para garantizar la cobertura de pruebas adecuada, se han aplicado distintos enfoques metodológicos en cada una de ellas:

1. **Primera función**: Se ha realizado una identificación de las clases de equivalencia y los valores límite, con el fin de validar la robustez y fiabilidad de los casos de prueba. Este enfoque permite detectar posibles errores en los bordes de los rangos definidos para cada entrada del sistema.
2. **Segunda función**: Se ha elaborado una gramática formal y su correspondiente árbol de derivación. Esta técnica permite analizar la estructura sintáctica de los datos de entrada, asegurando que cumplen con las reglas establecidas antes de ser procesados por el sistema.
3. **Tercera función**: Se ha construido el **gráfico de control de flujo**, identificando sus rutas básicas y los casos de prueba adicionales necesarios para evaluar los bucles. Este enfoque facilita el análisis del comportamiento del sistema bajo diferentes escenarios de ejecución.

Adicionalmente, como parte del proyecto, se ha desarrollado un manejador del flujo de datos de los documentos en formato JSON. Para ello, se ha implementado una clase denominada **json_manager**, la cual se encarga de gestionar la escritura y lectura de archivos JSON. Esto garantiza la persistencia de los datos y la correcta manipulación de la información durante la ejecución de las pruebas.

El presente documento detalla el análisis y la metodología aplicada para la ejecución de las pruebas en cada una de estas funciones, asegurando la validez de los resultados y la congruencia con los requisitos del sistema.

Función 1

Para la función 1 debemos de identificar las clases de equivalencia y los valores límite de la función.

Clases de equivalencia y valores límite:

Los parámetros de entrada y sus condiciones son las siguientes:

- **from_iban.** Código IBAN de una cuenta española desde la que se va a enviar el dinero. Usamos el método *validate_iban* del proyecto EG1 para comprobarlo
- **to_iban.** Código IBAN de una cuenta española que va a recibir el dinero. Usamos también el método *validate_iban* del proyecto EG1 para comprobarlo
- **concept.** Razón de la operación. Debe ser un texto libre entre 10 y 30 caracteres con al menos 2 cadenas con caracteres [a-z A-Z], separadas por un espacio blanco.
- **type.** Tipo de transferencia donde los posibles valores son ORDINARY, URGENT or INMEDIATE.
- **date.** Fecha de la operación que debe ir en formato "DD/MM/YYYY" (ejemplo (01/07/2024). Para validarlo habrá que comprobar lo siguiente:
    - DD tendrá un valor entre 01 y 31.
    - MM tendrá un valor entre 01 y 12.
    - YYYY será igual o mayor que 2025 y menor que 2051.
    - La fecha debe ser igual o posterior a la fecha en que se ha recibido la orden de transferencia.
- **amount.** Cantidad a transferir. Debe ser un número igual o mayor a 10.00 y menor o igual a 10000.00. Además el valor recibido debe tener como máximo 2 decimales.

Salidas esperadas por la función:

1. **Una cadena en MD5** correspondiente al Transfer Code.
2. **Un fichero** en el que se ha incluido los datos de la transferencia que se está registrando en el sistema. El fichero almacenará todas las transferencias que se vayan recibiendo.
3. **Una excepción** en los siguientes casos:
    1) Los números de cuenta (from o to) recibidos no son válidos.
    2) El concepto no tiene un valor válido.

3) El tipo de transferencia no es válido.
4) La fecha de la transferencia no es válida.
5) La cantidad no es válida.
6) Existe en el fichero de salida una transferencia con los mismos datos.
7) Se considera que un valor no es válido si no tiene el formato definido en el requisito o si el valor no cumple con las restricciones establecidas en el requisito.

En la siguiente página se adjunta la tabla de las clases de equivalencia y valores límite, sacadas del excel corregido del **grupo 08**.

VARIABLE	Rule	VALID CLASSES	INVALID CLASSSES
from iban	tipo de dato	CEV1 - string	CENV1 - distinto de string
from iban	iban valido	CEV2 - iban valido (cumple algoritmo de validacion)	CENV2 - iban no cumple algoritmo de validacion (24 char)
from iban	ES	CEV3 - dos primeros chars son ES	CENV3 - 2 primeros caracteres no son ES
from iban	22 ultimos carateres IBAN = digitos(0-9)	CEV4 - 22 caracteres ultimos son dígitos(0-9)	CENV4 - 22 caracteres ultimos no son dígitos(0-9)
from iban	longitud	CEV5 - longitud = 24 chars VLV1 - 24 chars	CENV5 - longitud distinta de 24 chars VLNV1 - 23 chars VLNV2 - 25 chars
to iban	tipo de dato	CEV6 - string	CENV6 - distinto de string
to iban	iban valido	CEV7 - iban valido (cumple algoritmo de validacion)	CENV7 - iban no cumple algoritmo de validacion (24 char)
to iban	ES	CEV8 - dos primeros chars son ES	CENV8 - 2 primeros caracteres no son ES
to iban	22 ultimos carateres IBAN = digitos(0-9)	CEV9 - 22 caracteres ultimos son dígitos(0-9)	CENV9 - 22 caracteres ultimos no son dígitos(0-9)
to iban	longitud	CEV10 - longitud = 24 chars VLV2 - 24 chars	CENV10 - longitud distinto de 24 chars VLNV3 - 23 chars VLNV4 - 25 chars
concept	tipo de dato	CEV11 - string	CENV11 - distinto de string
concept	longitud>=10 y longitud <=30	CEV12 - debe tener entre 10 y 30 chars VLV3 - 10 chars VLV4 - 11 chars VLV5 - 29 chars VLV6 - 30 chars	CENV12 - menor que 10 chars CENV13 - mayor que 30 chars VLNV5 - 9 chars VLNV6 - 31 chars
concept	nº de cadenas de chars >=2	CEV13 - nº de cadenas de chars mayor o igual que 2 VLV7 - 2 cadenas de chars VLV8 - 3 cadenas de chars	CENV14 - nº de cadenas de chars menor que 2 VLNV7 - 1 cadena de chars
concept	concepto valido	CEV14 - string formado por caracteres de [a-z A-Z]	CENV15 - string formado por caracteres distintos a [a-z A-Z]
type	tipo de dato	CEV15 - string	CENV16 - distinto de string
type	tipo valido	CEV16 - ser "ORDINARY" CEV17 - ser "URGENT" CEV18 - ser "INMEDIATE"	CENV17 - ser distinto de "ORDINARY", "URGENT" o "INMEDIATE"
date	tipo de dato	CEV19 - string	CENV18 - distinto de string
date	formato valido	CEV20 - sigue el formato DD/MM/YYYY	CENV19 - no sigue el formato DD/MM/YYYY
date	valores validos para DD (debe tener valores de entre 01 y 31)	CEV21 - DD tiene que ser valores entre 01 y 31 VLV9 - el numero 01 VLV10 - el numero 02 VLV11 - el numero 30 VLV12 - el numero 31	CENV20 - DD es menor que 01 CENV21 - DD es mayor que 31 VLNV8 - el numero 00 VLNV9 - el numero 32
date	valores validos para MM (debe tomar valores de entre 01 y 12)	CEV22 - MM tiene que ser valores entre 01 y 12 VLV13 - el numero 01 VLV14 - el numero 02 VLV15 - el numero 11 VLV16 - el numero 12	CENV22 - MM es menor que 01 CENV23 - MM es mayor que 12 VLNV10 - el numero 00 VLNV11 - el numero 13
date	valores validos para YYYY (2025 <= YYYY < 2051)	CEV23 - YYYY debe ser entre 2025 y 2050 VLV17 - el numero 2025 VLV18 - el numero 2026 VLV19 - el numero 2049 VLV20 - el numero 2050	CENV24 - YYYY es menor que 2025 CENV25 - YYYY es mayor que 2050 VLNV12 - el numero 2024 VLNV13 - el numero 2051
date	fecha de transacción válida	CEV24 - fecha válida	CENV26 - Fecha no válida (es menor a la fecha de la orden de transferencia)
amount	tipo de dato	CEV25 - float	CENV27 - distinto de float
amount	amount >= 10.00 y amount <= 10000.00	CEV26- numero entre 10.00 y 10000.00 VLV21 - el numero 10.00 VLV22 - el numero 10.01 VLV23 - el numero 9999.99	CENV28 - numero menor que 10.00 CENV29 - numero mayor que 10000.00 VLNV14 - el numero 9.99 VLNV15 - el numero 10000.01
amount	float valido (2 decimales máximo)	CEV27 - tiene entre 0 y 2 decimales VLV24 - 0 deciamales VLV25 - 1 decimal VLV26 - 2 decimales	CENV30 - tiene mas de 2 decimales VLNV16 - 3 decimales
salidas	salidas	CEV28 - cadena en MD5 correspondiente al Tranfer Code CEV29 - fichero que contiene todas las transferencias con sus	CENV31- número de cuenta inválido CENV32 - excepcion: concept no valido CENV33 - excepcion: type no valido CENV34 - excepcion: date no valido CENV35 - excepcion: amount no valido CENV36 - excepcion: transferencia con mismos datos en el fichero de salida

Casos de prueba 1:

Los casos de prueba correspondiente a la función 1 son las siguientes:

	I/O	TECHNIC	VALID/INVAL	ID TI	DESCRIPTIC	From_iban	to_iban	concept	type	date	amoui	RESULT	ECS, BV COVERED
1	INPUT	CE	VALID	TC1	caso valido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	10,00	TRUE	CEV1, CEV2, CEV3, CEV4, CEV5, CEV6, CEV7, CEV8, CEV9, CEV10, CEV11, CEV12, CEV13, CEV14, CEV15, CEV16C EV19, CEV20, CEV21, CEV22, CEV23, CEV24, CEV25, CEV26, CEV27, CEV28, CEV29, VLV1, VLV2, VLV3, VLV7, VLV9, VLV13, VLV17, VLV21, VLV24
2	INPUT	CE	VALID	TC2	caso valido	ES9121000418450200051332	ES4500817294770123456789	"concept val"	URGENT	02/02/2026	10,01	TRUE	CEV17, VLV4, VLV10, VLV14, VLV18, VLV22, VLV26
3	INPUT	CE	VALID	TC3	caso valido	ES9121000418450200051332	ES4500817294770123456789	"transferencia inmediata viaje"	INMEDIATE	30/11/2049	9999,99	TRUE	CEV18, VLV5, VLV8, VLV11, VLV15, VLV19, VLV23
4	INPUT	CE	VALID	TC4	caso valido	ES9121000418450200051332	ES4500817294770123456789	"transfere compra semana pasada"	ORDINARY	31/12/2050	400,3	TRUE	VLV6, VLV12, VLV16, VLV20, VLV25
5	INPUT	CE	INVALID	TC5	caso invalido	123	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	10,00	"Excepción: Los números de cuenta (from) recibidos no son válidos."	CENV1, CENV31
6	INPUT	CE	INVALID	TC6	caso invalido	ES3400491500950012345679	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	10,00	"Excepción: Los números de cuenta (from) recibidos no son válidos."	CENV2
7	INPUT	CE	INVALID	TC7	caso invalido	S3400491500950012345678O	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	10,00	"Excepción: Los números de cuenta (from) recibidos no son válidos."	CENV3
8	INPUT	CE	INVALID	TC8	caso invalido	ES23456789UYTR3456789654	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	10,00	"Excepción: Los números de cuenta (from) recibidos no son válidos."	CENV4
9	INPUT	CE	INVALID	TC9	caso invalido	ES1234567891234567891l2	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	10,00	"Excepción: Los números de cuenta (from) recibidos no son válidos."	CENV5, VLNV1
10	INPUT	CE	INVALID	TC10	caso invalido	ES3400491500950012345679l	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	10,00	"Excepción: Los números de cuenta (from) recibidos no son válidos."	VLNV2
11	INPUT	CE	INVALID	TC11	caso invalido	ES9121000418450200051332	123	"text valid"	ORDINARY	01/01/2025	10,00	"Excepcion: Los números de cuenta (to) recibidos no son válidos."	CENV6
12	INPUT	CE	INVALID	TC12	caso invalido	ES9121000418450200051332	ES3400491500950012345679	"text valid"	ORDINARY	01/01/2025	10,00	"Excepcion: Los números de cuenta (to) recibidos no son válidos."	CENV7
13	INPUT	CE	INVALID	TC13	caso invalido	ES9121000418450200051332	S3400491500950012345678O	"text valid"	ORDINARY	01/01/2025	10,00	"Excepcion: Los números de cuenta (to) recibidos no son válidos."	CENV8
14	INPUT	CE	INVALID	TC14	caso invalido	ES9121000418450200051332	ES23456789UYTR3456789654	"text valid"	ORDINARY	01/01/2025	10,00	"Excepcion: Los números de cuenta (to) recibidos no son válidos."	CENV9
15	INPUT	CE	INVALID	TC15	caso invalido	ES9121000418450200051332	ES1234567891234567891l2	"text valid"	ORDINARY	01/01/2025	10,00	"Excepcion: Los números de cuenta (to) recibidos no son válidos."	CENV10, VLNV3
16	INPUT	CE	INVALID	TC16	caso invalido	ES9121000418450200051332	ES3400491500950012345679l	"text valid"	ORDINARY	01/01/2025	10,00	"Excepcion: Los números de cuenta (to) recibidos no son válidos."	VLNV4
17	INPUT	CE	INVALID	TC17	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"1111111111"	ORDINARY	01/01/2025	10,00	"Excepción: El concepto no tiene un valor válido."	CENV11, CENV32
18	INPUT	CE	INVALID	TC18	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"Pay Final"	ORDINARY	01/01/2025	10,00	"Excepción: El concepto no tiene un valor válido."	CENV12, VLNV5
19	INPUT	CE	INVALID	TC19	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"transferencia para javier perez"	ORDINARY	01/01/2025	10,00	"Excepción: El concepto no tiene un valor válido."	CENV13, VLNV6
20	INPUT	CE	INVALID	TC20	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"pagoparapablo"	ORDINARY	01/01/2025	10,00	"Excepción: El concepto no tiene un valor válido."	CENV14, VLNV14
21	INPUT	CE	INVALID	TC21	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"Pago para **pablo"	ORDINARY	01/01/2025	10,00	"Excepción: El concepto no tiene un valor válido."	CENV15
22	INPUT	CE	INVALID	TC22	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	123	01/01/2025	10,00	"Excepción: El tipo de transferencia no es válido."	CENV16, CENV33
23	INPUT	CE	INVALID	TC23	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	OTHER	01/01/2025	10,00	"Excepción: El tipo de transferencia no es válido."	CENV17
24	INPUT	CE	INVALID	TC24	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	123	10,00	"Excepción: La fecha de la transferencia no es válida."	CENV18, CENV34
25	INPUT	CE	INVALID	TC25	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	11-12-2027	10,00	"Excepción: La fecha de la transferencia no es válida."	CENV19
26	INPUT	CE	INVALID	TC26	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	00/02/2026	10,00	"Excepción: La fecha de la transferencia no es válida."	CENV20, VLNV8
27	INPUT	CE	INVALID	TC27	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	32/02/2026	10,00	"Excepción: La fecha de la transferencia no es válida."	CENV21, VLNV9
28	INPUT	CE	INVALID	TC28	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	02/00/2026	10,00	"Excepción: La fecha de la transferencia no es válida."	CENV22, VLNV10
29	INPUT	CE	INVALID	TC29	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	02/13/2026	10,00	"Excepción: La fecha de la transferencia no es válida."	CENV23, VLNV11
30	INPUT	CE	INVALID	TC30	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	02/02/2024	10,00	"Excepción: La fecha de la transferencia no es válida."	CENV24, CENV26 VLNV12
31	INPUT	CE	INVALID	TC31	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	02/02/2051	10,00	"Excepción: La fecha de la transferencia no es válida."	CENV25, VLNV13
32	INPUT	CE	INVALID	TC32	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	num	"Excepción: La cantidad no es válida."	CENV27, CENV35
33	INPUT	CE	INVALID	TC33	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	9,99	"Excepción: La cantidad no es válida."	CENV28, VLNV14
34	INPUT	CE	INVALID	TC34	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	10000,01	"Excepción: La cantidad no es válida."	CENV29, VLNV15
35	INPUT	CE	INVALID	TC35	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	123,123	"Excepción: La cantidad no es válida."	CENV30, VLNV16
36	INPUT	CE	INVALID	TC36	caso invalido	ES9121000418450200051332	ES4500817294770123456789	"text valid"	ORDINARY	01/01/2025	10	"Error, la transferencia ya existe"	CENV36

Función 2

Para la función 2 debemos de realizar un análisis sintáctico. Debemos de realizar una gramática, el árbol de derivación y definir los casos de prueba.

Gramática:

Para realizar la gramática debemos de tener en cuenta que el archivo de entrada JSON debe cumplir con el siguiente formato:

{
"IBAN":"<SPANISH VALID IBAN CODE>",
"AMOUNT": "EUR ####.##"
}

donde:

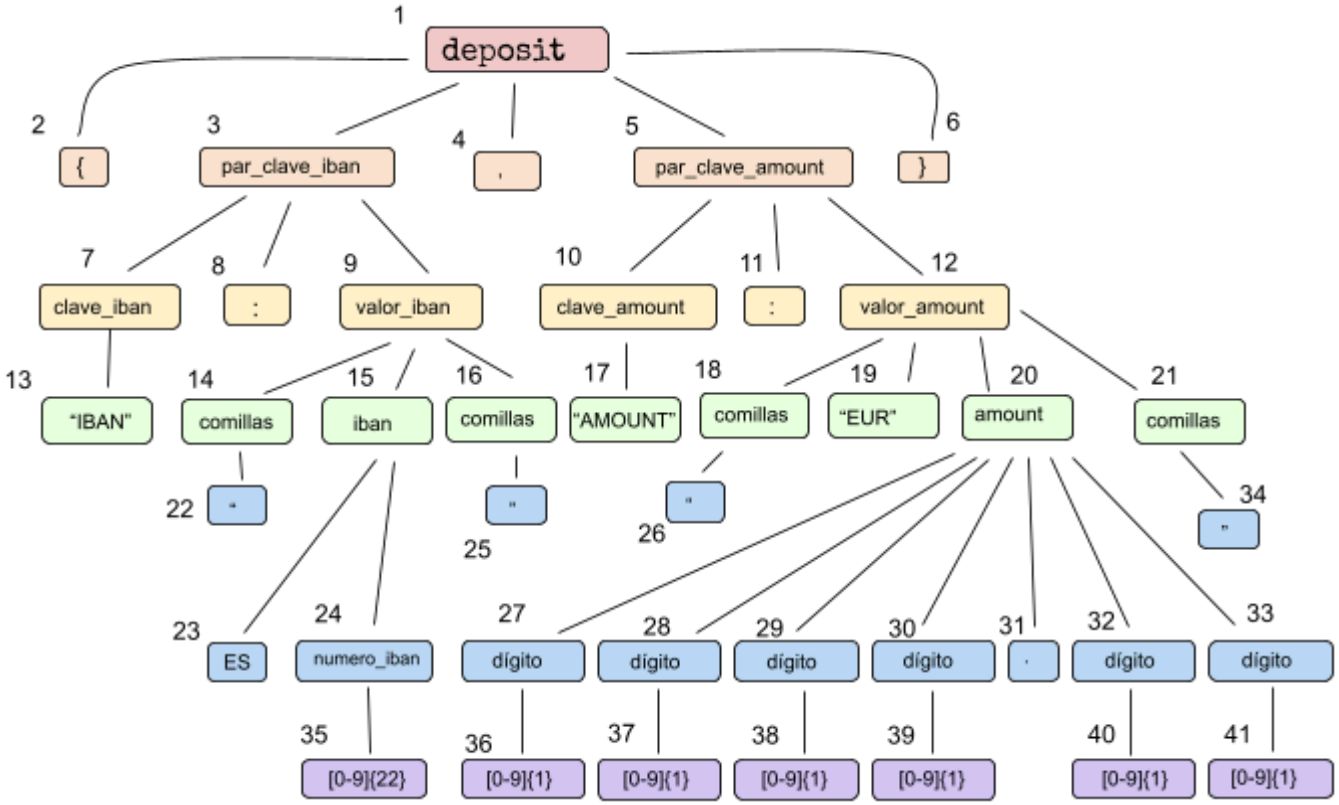
- **EUR** es la moneda y es un valor constante que en el futuro podría tomar otros valores.
- # es un carácter numérico con valor **0-9**

Una gramática posible para esta función podría ser la siguiente:

<**deposit**> ::= “ { ” <par_clave_iban> “ , ” <par_clave_amount> “ } ”
<**par_clave_iban**> ::= <clave_iban> “ : ” <valor_iban>
<**clave_iban**> ::= “IBAN”
<**valor_iban**> ::= <comillas> <iban> <comillas>
<**comillas**> ::= “
<**iban**> ::= “ES” <numero_iban>
<**numero_iban**> ::= [0-9]{22}
<**par_clave_amount**> ::= <clave_amount> “ : ” <valor_amount>
<**clave_amount**> ::= “AMOUNT”
<**valor_iban**> ::= <comillas> “ EUR ” <amount> <comillas>
<**amount**> ::= <digito> <digito> <digito> <digito> “ . ” <digito><digito>
<**digito**> ::= [0-9]{1}

Árbol de derivación:

El árbol de derivación obtenido es el siguiente:



Casos de prueba 2:

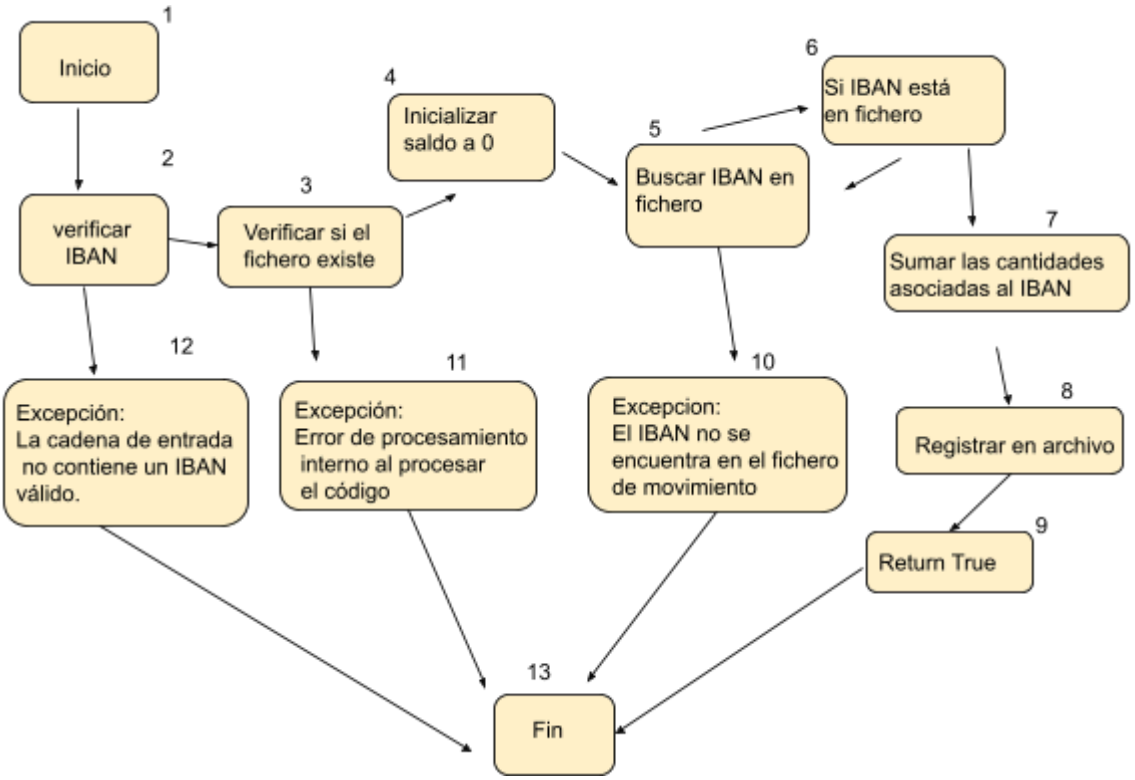
En la página siguiente se adjunta los casos de prueba de la segunda función

]	NODE	TERM	TYPE (DUPLICATION / DELETION	ID TEST	DESCRIPTION	FILE PATH	FILE CONTENT	EXPECTED RESULT	OBSERVATIONS
1	2,4,6,13,22,23,31, 34,35,36,37,38,39, 40,41	T	VALIDO	test_valid_tc1	test valido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	valido	
2	1	NT	BORRADO	test_invalid_tc1	el fichero esta vacio	deposit_rquests.json	fichero vacio	Excepción: El JSON no tiene la estructura esperada.	
3	1	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" } { "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
4	2	T	BORRADO	test_invalid_tc3	estructura inválida	deposit_rquests.json	"IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
5	2	T	DUPLICADO	test_invalid_tc4	estructura inválida	deposit_rquests.json	{{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
6	2	T	MODIFICADO		estructura inválida	deposit_rquests.json	$"IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
7	3	NT	BORRADO	test_invalid_tc5	estructura inválida	deposit_rquests.json	{ , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
8	3	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
9	4	NT	BORRADO		estructura inválida	deposit_rquests.json	{"IBAN" : "ES4500817294770123456789" "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
10	4	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" ,, "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
11	4	NT	MODIFICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" / "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
12	5	NT	BORRADO	test_invalid_tc6	estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , }	Excepción: El JSON no tiene la estructura esperada.	
13	5	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23""AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
14	6	T	BORRADO	test_invalid_tc2	estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23"	Excepción: El JSON no tiene la estructura esperada.	
15	6	T	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }}	Excepción: El JSON no tiene la estructura esperada.	
16	6	T	MODIFICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" &	Excepción: El JSON no tiene la estructura esperada.	
17	7,13	NT	BORRADO		estructura inválida	deposit_rquests.json	{ : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	Borrado del nodo 7 igual que el borrado del nodo 13
18	7,13	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN""IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	Duplicado del nodo 7 igual que el duplicado del nodo 13
19	8	NT	BORRADO		estructura inválida	deposit_rquests.json	{ "IBAN" "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
20	8	T	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" :: "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
21	8	T	MODIFICADO		estructura inválida	deposit_rquests.json	{ "IBAN" % "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
22	9	NT	BORRADO		estructura inválida	deposit_rquests.json	{ "IBAN" : , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
23	9	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : ES4500817294770123456789""ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
24	10,17	NT	BORRADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	Borrado del nodo 10 igual que el borrado del nodo 17
25	10,17	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT""AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	Duplicado del nodo10 igual que el duplicado del nodo 17
26	11	T	BORRADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
27	11	T	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" :: "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
28	11	T	MODIFICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" % "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
29	12	NT	BORRADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : }	Excepción: El JSON no tiene la estructura esperada.	
30	12	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23""EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
31	13	T	MODIFICADO		estructura inválida	deposit_rquests.json	{ "CASA" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
32	14,16,22,25	NT	BORRADO		estructura inválida	deposit_rquests.json	{ "IBAN" : ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	Borrados de los nodos 14,16 iguales que los borrados de los nodos 22,25
33	14,16,22,25	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : ""ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	Duplicados de los nodos 14,16 iguales que los duplicados de los nodos 22,25
34	15	NT	BORRADO		estructura del IBAN inválido	deposit_rquests.json	{ "IBAN" : "" , "AMOUNT" : "EUR1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	
35	15	NT	DUPLICADO		estructura del IBAN inválido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	
36	17	T	MODIFICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "CASA" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
37	18,21,26,34	NT	BORRADO	test_invalid_tc8	estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	Borrados de los nodos 18,21 iguales que los borrados de los nodos 26,34
38	18,21,26,34	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : ""EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	Duplicados de los nodos 18,21 iguales que los duplicados de los nodos 26,34
39	19	T	BORRADO		estructura del amount inválido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	
40	19	T	DUPLICADO		estructura del amount inválido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUREUR1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	
41	19	T	MODIFICADO		estructura del amount inválido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "DOLLAR1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	
42	20	NT	BORRADO		estructura del amount inválido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR" }	Excepción: Los datos del JSON no tienen valores válidos.	
43	20	NT	DUPLICADO		estructura del amount inválido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.231200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	
44	22,25	T	MODIFICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : %ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	Modificado del nodo 22 igual que el modificado del nodo 25
45	23	T	BORRADO	test_invalid_tc12	estructura del IBAN inválido	deposit_rquests.json	{ "IBAN" : "450081729477012345678945008172947701234567890" , "AMOUNT" : "EUR1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	
46	23	T	DUPLICADO		estructura del IBAN inválido	deposit_rquests.json	{ "IBAN" : "ESES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	
47	23	T	MODIFICADO		estructura del IBAN inválido	deposit_rquests.json	{ "IBAN" : "GB4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	
48	24,35	NT	BORRADO	test_invalid_tc10	estructura del IBAN inválido	deposit_rquests.json	{ "IBAN" : "ES" , "AMOUNT" : "EUR1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	Borrado del nodo 24 igual que el borrado del nodo 35
49	24,35	NT	DUPLICADO	test_invalid_tc11	estructura del IBAN inválido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789450081729477012345678 9" , "AMOUNT" : "EUR1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	Duplicado del nodo 24 igual que el duplicado del nodo 35
50	26,34	T	MODIFICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : %EUR1200.23" }	Excepción: El JSON no tiene la estructura esperada.	Modificado del nodo 26 igual que el modificado del nodo 34
51	27,28,29,30,32,33,35, 36,37,38,39,40,41	NT	BORRADO	test_invalid_tc15	estructura del amount inválido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	Borrados de los nodos 27,28,29,30,32,33 iguales que los borrados de los nodos 35,36,37,38,39,40,41
52	27,28,29,30,32,33,35, 36,37,38,39,40,41	NT	DUPLICADO	test_invalid_tc14	estructura del amount inválido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	Duplicados de los nodos 27,28,29,30,32,33 iguales que los duplicados de los nodos 35,36,37,38,39,40,41
53	31	NT	BORRADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR120023" }	Excepción: El JSON no tiene la estructura esperada.	
54	31	NT	DUPLICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200..23" }	Excepción: El JSON no tiene la estructura esperada.	
55	31	T	MODIFICADO		estructura inválida	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EUR1200%23" }	Excepción: El JSON no tiene la estructura esperada.	
56	35	T	MODIFICADO	test_invalid_tc9	estructura del IBAN inválido	deposit_rquests.json	{ "IBAN" : "ESABCD" , "AMOUNT" : "EUR1200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	
57	36,37,38,39,40,41	T	MODIFICADO	test_invalid_tc13	estructura del amount inválido	deposit_rquests.json	{ "IBAN" : "ES4500817294770123456789" , "AMOUNT" : "EURA200.23" }	Excepción: Los datos del JSON no tienen valores válidos.	Modificado del nodo 35 igual que los modificados de los nodos 36,37,38,39,40,41

Función 3

En la función 3 debemos de realizar un diagrama de control de flujo, definir las rutas básicas, realizar los casos adicionales para probar los bucles y los casos de prueba.

Diagrama de control de flujo:



Podemos calcular la complejidad ciclomática sin incluir métodos a los que llama:

1. Enlaces = 16
2. Nodos= 13
3. Sabemos que **V(G) = E-N+2**
4. Luego V(G) = 16-13+2 = 5

Definición de rutas básicas:

Ruta 1

Camino →1,2,3,4,5,6,7,8,9,13

Camino de cuando el IBAN está en el archivo, y se suma todas las cantidades asociadas al IBAN

Ruta 2

Camino→ 1,2,12,13

Camino de cuando el IBAN es inválido

Ruta 3

Camino →1,2,3,11,13

Camino de cuando el fichero de movimiento no existe

Ruta 4

Camino →1,2,3,4,5,6,5,10,13

Camino de cuando el IBAN no se encuentra en el fichero

Ruta 5

Camino →1,2,3,4,5,10,13

Camino de cuando el fichero existe pero está vacío

Casos adicionales para probar los bucles:

Para asegurar que el bucle cubre todo, consideraremos los siguientes casos:

1. **Caso 1:** No entrar al bucle (No hay transacciones en el fichero de movimientos)
2. **Caso 2:** Entrar 1 vez. Hay una sola transacción
3. **Caso 3:** Hay 2 veces. Hay dos transacciones
4. **Caso 4:** Entrar máx veces. No haría falta ya que el bucle siempre recorre n veces. Podemos probar con el archivo transactions.json que contiene 20 transacciones

Nota: Entrar máx-1 o máx+1 no tiene sentido ya que el bucle se recorre siempre n veces.

Casos de prueba 3:

Estos fueron los casos de prueba resultantes:

#	PATH	ID TEST	DESCRIPTION	FIELD	EXPECTED RESULT	OBSERVATIONS
1	1,2,3,4,5,6,7,8,9,13 Caso 4 bucle	test_valid_tc1	Camino de cuando el IBAN está en el archivo, y se suma todas las cantidades asociadas al IBAN		TRUE	Suponemos que le maximo de veces sea el tamaño de transactions.json, es decir 20 iteaciones
2	1,2,12,13	test_invalid_tc1	Camino de cuando el IBAN es inválido		Exception: La cadena de entrada no contiene un IBAN válido.	Estructura de IBAN inválido
3	1,2,3,13	test_invalid_tc2	Camino de cuando el fichero de movimiento no existe		Excepción:Error de procesamiento interno al procesar el código"	
4	1,2,3,4,5,6,5,10,13 Caso 2 bucle	test_invalid_tc3	Camino de cuando el IBAN no se encuentra en el fichero		Excepcion: El IBAN no se encuentra en el fichero de movimiento	solo hay 1 IBAN dentro de transactions.json pero no encuentra el IBAN
5	1,2,3,4,5,10,13 Caso 1 bucle	test_invalid_tc4	Camino de cuando el fichero existe pero está vacío		Excepcion: El IBAN no se encuentra en el fichero de movimientos	El fichero existe pero está vacío, no entra al bucle
6	Caso 3 bucle	test_valid_tc22	Entra dos veces al bucle, y si encuentra el IBAN		TRUE	Solo hay 2 IBANs dentro de transactions.json