Laura Carratalá Castillo

**Pruebas de caja negra y caja blanca**

A continuación, se reflejarán de forma clara y concisa los pasos para la elaboración de las pruebas de caja negra y caja blanca de dos métodos pertenecientes a dos de nuestras aplicaciones que componen la Navaja Suiza. En nuestro caso, hemos elegido la aplicación 1, que consiste en la determinación de si un número positivo introducido es primo o no, y la aplicación 2, que consiste en mostrar los múltiplos de 3 y 5 encontrados hasta un número introducido, estando este comprendido entre el 1 y el 100.

**Pruebas para la aplicación 1:**

Como hemos indicado anteriormente, la aplicación 1 consiste en determinar si un número positivo introducido es primo o no. El método con el cual haremos las pruebas se llama **EsPrimo** y su código es el siguiente:



En primer lugar, realizaremos las pruebas de caja negra. Para ello, definiremos las clases de equivalencia elegidas para cubrir el mayor número de pruebas posible. Cada prueba tendrá un identificador, en nuestro caso consistirá en la abreviatura “App”, más el número de la aplicación con la que estemos haciendo las pruebas (1 o 2), más un punto, más el número de la prueba, más la letra “N” o “B” en función de si estamos haciendo las pruebas de caja negra (N) o de caja blanca (B). P.ej: App1.1N, App1.1B.

Clases de equivalencia:

-  **numIntroducido:** Es un int, deberá ser mayor que 0.

App1.1N) numIntroducido >= 3 (que sea primo): Introducción de cualquier número entero positivo mayor o igual que 3 que sea primo. Valor límite: 3.

App1.2N) numIntroducido >= 3 (que no sea primo): Introducción de cualquier número entero positivo mayor o igual que 3 que no sea primo. Valor límite: 4.

App1.3N) 0 < numIntroducido < 3: Introducción de valores positivos menores que 3. Son valores especiales puesto que son susceptibles de dar algún error. Valores límite: 1, 2.

App1.4N) numIntroducido == 0: Es un valor especial. Ya no entra en el rango de valores.

App1.5N) numIntroducido < 0: Introducción de valores negativos. Valores límite: -1, - 2, -3, -4.

App1.6N) numIntroducido == decimal: Introducción de un número decimal, ya sea positivo o negativo.

App1.7N) numIntroducido == letra: Introducción de letras.

App1.8N) numIntroducido == símbolo: Introducción de caracteres no alfanuméricos.

App1.9N) NumIntroducido == “”: No introducir ningún carácter.

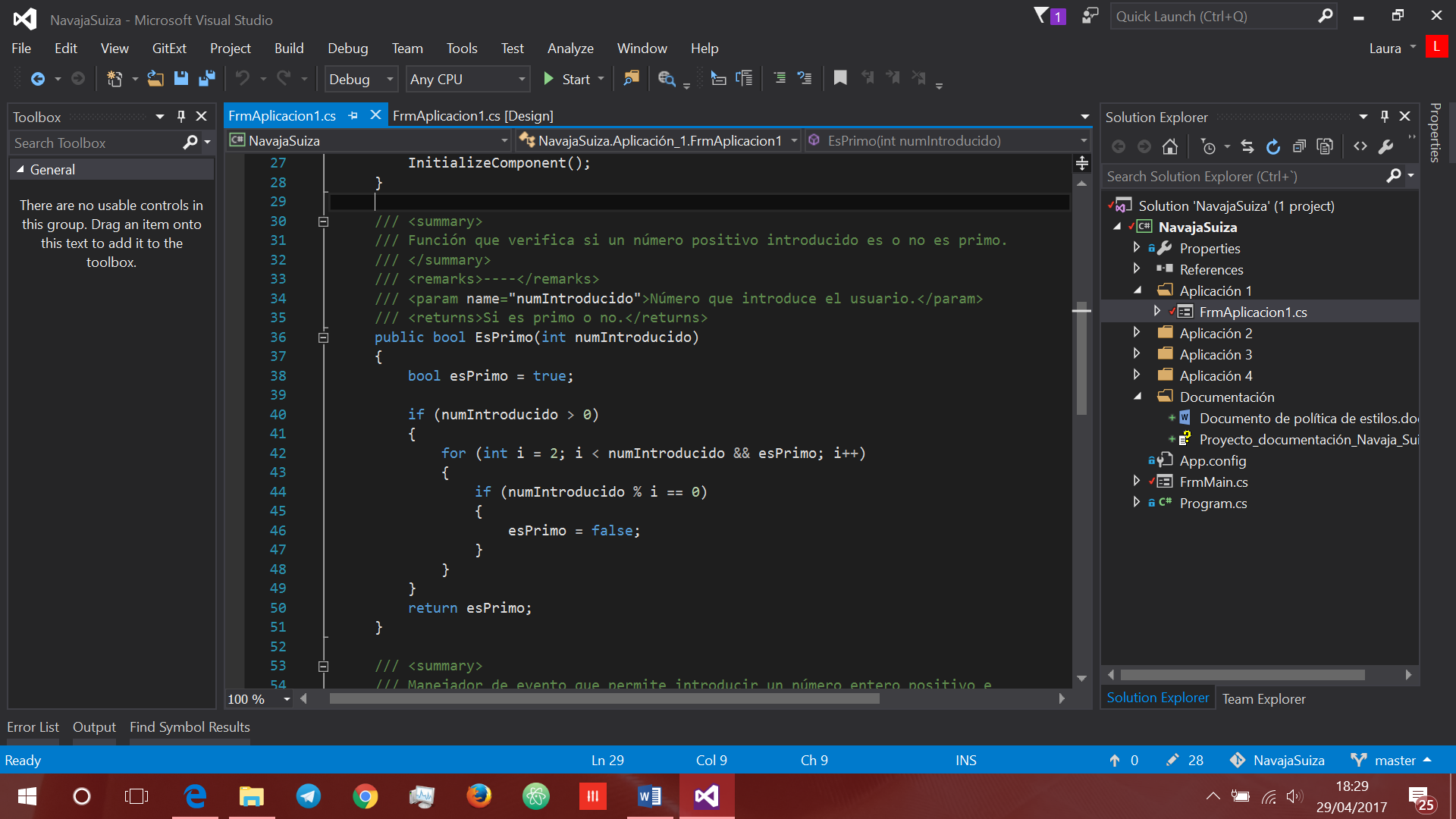
App1.10N) NumIntroducido == Máximo valor (999999999): Introducción del máximo valor que se puede introducir en una caja de texto, controlando también los límites. En nuestro caso, como hemos limitado el número máximo de caracteres que se pueden introducir a 9, el máximo valor que podemos introducir es el 999999999. Valores límite: 999999998, 999999999, 1000000000.

App1.11N) NumIntroducido == Mínimo valor (-99999999): Introducción del mínimo valor que se puede introducir en la caja de texto, controlando también los límites. En nuestro caso, como hemos limitado el número máximo de caracteres que se pueden introducir a 9, el máximo valor que podemos introducir es el -99999999. Valores límite: -99999998, -99999999, -100000000.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Prueba** | **Clase de equivalencia** | **Valores de entrada** | **Resultado** | **Comentarios** |
| App1.1N | numIntroducido >= 3 (que sea primo) | 3, 89 | Válido | Los números son primos. |
| App1.2N | numIntroducido >= 3 (que no sea primo) | 4, 100 | Válido | Los números no son primos. |
| App1.3N | 0 < numIntroducido < 3 | 1, 2 | Para 2: Válido  Para 1: No Válido | El 2 es primo, pero el 1 no se considera primo y el programa indica que si lo es |
| App1.4N | numIntroducido == 0 | 0 | No válido | El 0 no es primo y el programa indica que si lo es. |
| App1.5N | numIntroducido < 0 | -1, -2, -3, -4,  -50 | No válido | El programa indica que son primos cuando no lo son. |
| App1.6N | numIntroducido == decimal | 1.5, -1.5 | Válido | El programa indica que se introduzca un elemento válido, puesto que solo acepta números enteros. |
| App1.7N | numIntroducido == letra | A, a, aa | Válido | El programa indica que se introduzca un elemento válido, puesto que solo acepta números enteros. |
| App1.8N | numIntroducido == simbolo | >, \* | Válido | El programa indica que se introduzca un elemento válido, puesto que solo acepta números enteros. |
| App1.9N | numIntroducido == “” | (vacío) | Válido | El programa indica que se introduzca un elemento válido, puesto que solo acepta números enteros. |
| App1.10N | numIntroducido == máximo valor (999999999) | 999999998, 999999999, 1000000000 | Válido | El programa indica que tanto el 999999998 como el 999999999, no son primos. Al introducir el 1000000000, no nos deja puesto que supera el número máximo de caracteres que se pueden introducir en el textbox. |
| App1.11N | numIntroducido == mínimo valor | -99999998,  -99999999,  -100000000 | No válido | El programa indica que tanto el -99999998 como el -99999999 son primos, cuando no lo son. Al introducir el -100000000, no nos deja puesto que supera el número máximo de caracteres que se pueden introducir en el textbox. |

Vemos que con todas estas pruebas hemos cubierto la totalidad del código de nuestro método, con lo que ya no sería necesario realizar más pruebas.

Sin embargo, realizaremos las pruebas de caja blanca, cubriendo las distintas partes de las que se compone nuestro método. En total hay 3 bucles que se enumeran en rojo de la forma que vemos en la siguiente imagen:



App1.1B

App1.2B

App1.4B

App1.3B

Para cubrir todos los bucles, será necesario realizar únicamente las siguientes pruebas:

|  |  |  |
| --- | --- | --- |
| **Prueba** | **Valor de entrada** | **Resultado** |
| App1.1B | 0 | No válido |
| App1.2B | 1 | No válido |
| App1.3B | 3 | Válido |
| App1.4B | 4 | Válido |

A continuación, indicaremos las pruebas de caja negra que son cubiertas en cada prueba de caja blanca:

- App1.1B: Cubre las pruebas App1.4N, App1.5N y App1.11N.

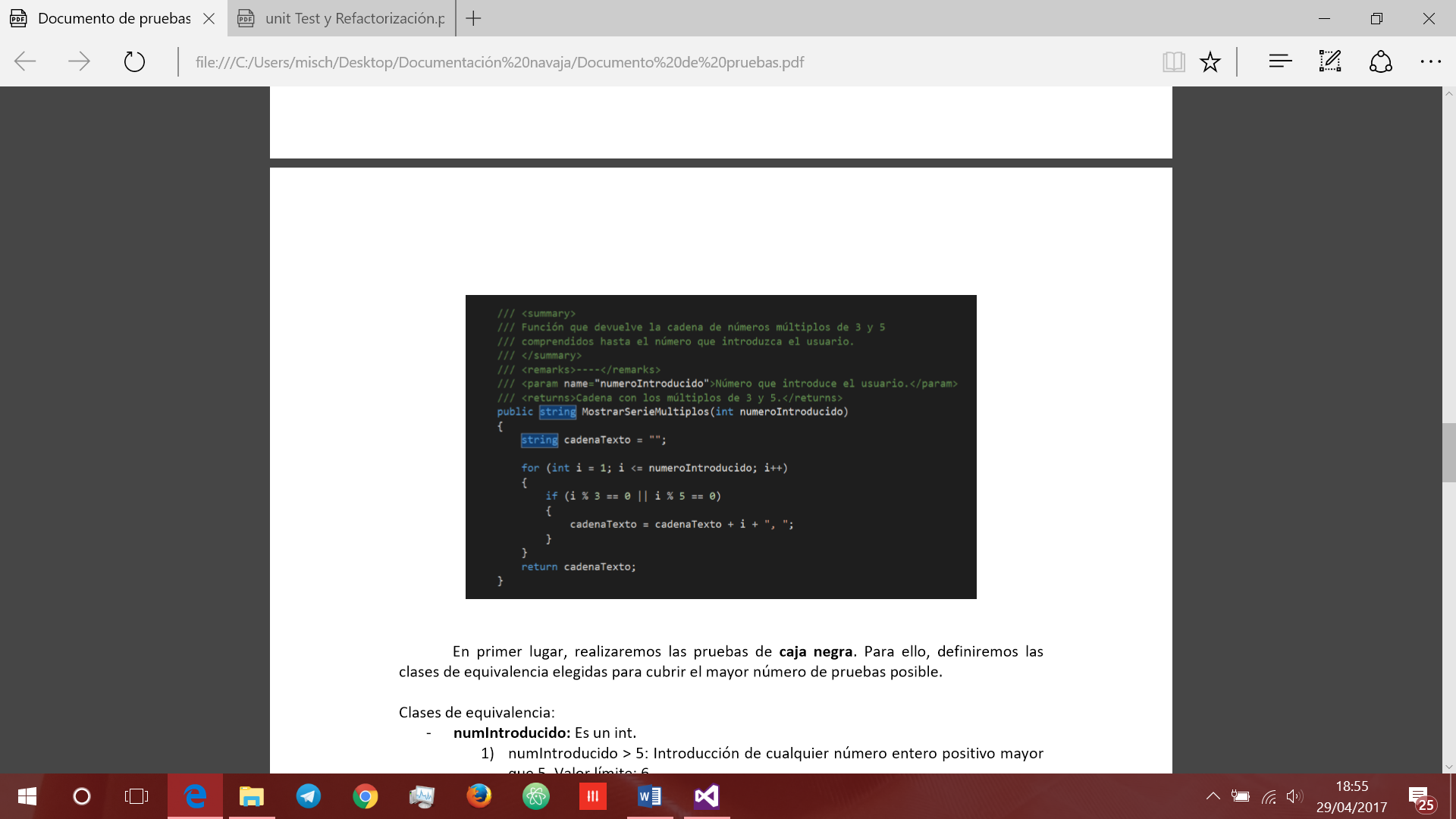
- App1.2B: Cubre la prueba App1.3N.

- App1.3B: Cubre la prueba App1.1N.

- App1.4B: Cubre las pruebas App1.2N y App1.10N.

**Pruebas para la aplicación 2:**

Como hemos indicado anteriormente, la aplicación 2 consiste en mostrar los múltiplos de 3 y 5 encontrados hasta un número introducido, estando este comprendido entre el 1 y el 100. El método con el cual haremos las pruebas se llama **MostrarSerieMultiplos** y su código es el siguiente:



En primer lugar, realizaremos las pruebas de caja negra. Para ello, definiremos las clases de equivalencia elegidas para cubrir el mayor número de pruebas posible. Al igual que en el caso anterior, usaremos la misma nomenclatura para dar un identificador a cada prueba.

Clases de equivalencia:

**- numIntroducido**: Es un int.

App2.1N) 5 > numIntroducido >= 100: Introducción de cualquier número entero positivo mayor que 5 y menor o igual a 100. Valores límite: 6, 101.

App2.2N) 3 <= numIntroducido <= 5: Introducción de valores enteros positivos entre el 3 y el 5 incluidos. Valores límite: 3, 4, 5.

App2.3N) numIntroducido < 3: Introducción de valores positivos hasta el 2. Valores límite: 1, 2.

App2.4N) numIntroducido == 0: Es un valor especial, ya no entra en el rango de valores permitidos.

App2.5N) numIntroducido < 0: Introducción de valores negativos. Valores límite: -1, -3, -5.

App2.6N) numIntroducido == decimal: Introducción de un número decimal, ya sea positivo o negativo.

App2.7N) numIntroducido == letra: Introducción de letras.

App2.8N) numIntroducido == símbolo: Introducción de caracteres no alfanuméricos.

App2.9N) NumIntroducido == “”: No introducir ningún carácter.

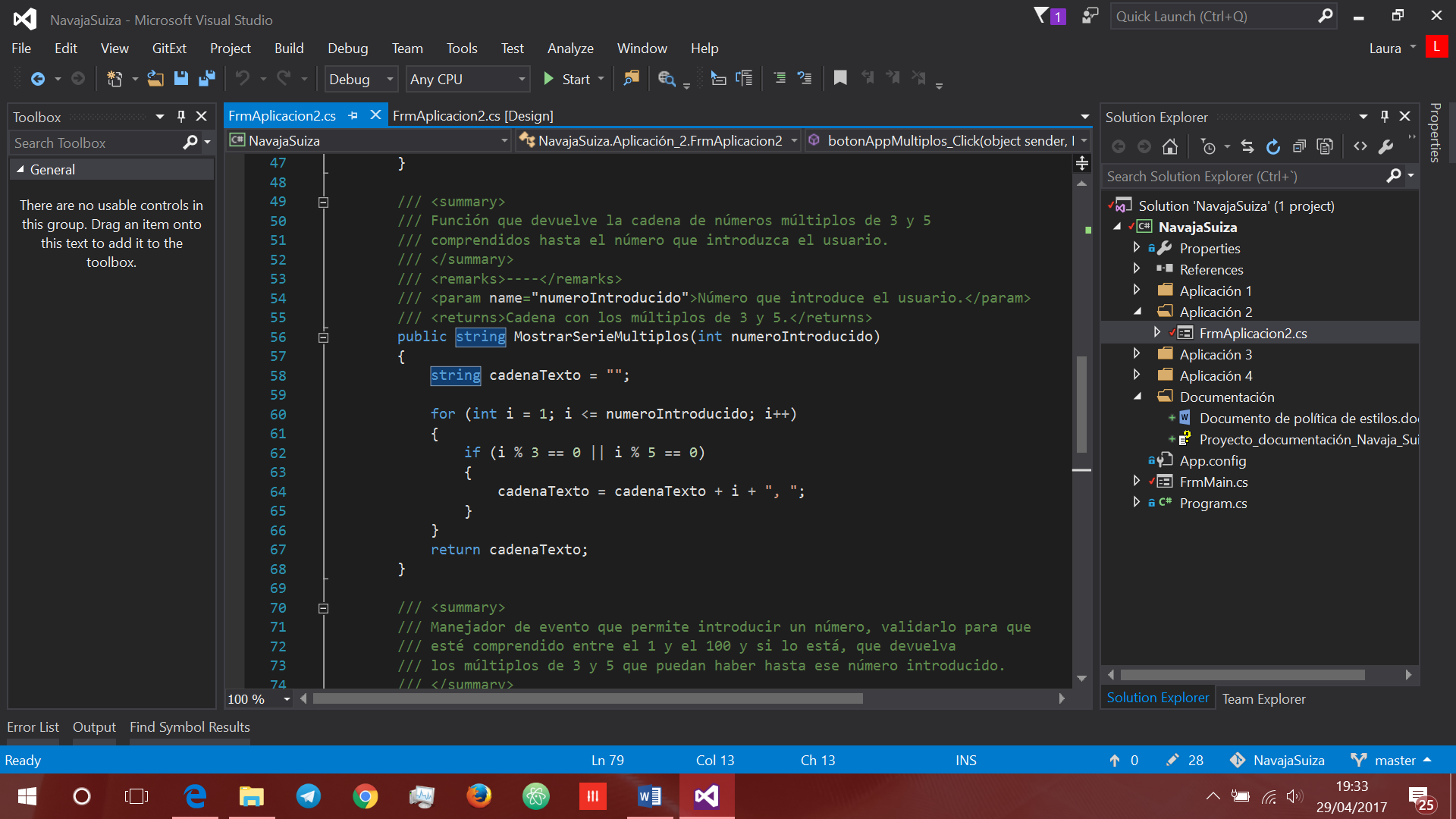
App2.10N) NumIntroducido == Máximo valor (999999999): Introducción del máximo valor que se puede introducir en una caja de texto, controlando también los límites. En nuestro caso, como hemos limitado el número máximo de caracteres que se pueden introducir a 9, el máximo valor que podemos introducir es el 999999999. Valores límite: 999999998, 999999999, 1000000000.

App2.11N) NumIntroducido == Mínimo valor (-99999999): Introducción del mínimo valor que se puede introducir en la caja de texto, controlando también los límites. En nuestro caso, como hemos limitado el número máximo de caracteres que se pueden introducir a 9, el máximo valor que podemos introducir es el -99999999. Valores límite: -99999998, -99999999, -100000000.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Prueba** | **Clase de equivalencia** | **Valores de entrada** | **Resultado** | **Comentarios** |
| App2.1N | 5 > numIntroducido >= 100 | 6, 7, 100, 101 | Válido | Como resultado obtenemos la cadena de múltiplos. Como el 7 no es múltiplo ni de 3 ni de 5, no se añade a la cadena. El resto de valores sí, excepto el 101, que se sale del rango de valores permitidos y el programa nos indica que introduzcamos un número entre el 1 y el 100. |
| App2.2N | 3 <= numIntroducido <= 5 | 3, 4, 5 | Válido | Como resultado obtenemos la cadena de múltiplos. Como el 4 no es múltiplo ni de 3 ni de 5, no se añade a la cadena. |
| App2.3N | numIntroducido < 3 | 1, 2 | Válido | Como el 1 y el 2 no son múltiplos ni de 3 ni de 5, no se añaden a la cadena. |
| App2.4N | numIntroducido == 0 | 0 | Válido | No entra en el rango de valores permitidos. |
| App2.5N | numIntroducido < 0 | -1, -3, -5 | Válido | No entran en el rango de valores permitidos. |
| App2.6N | numIntroducido == decimal | 1.5, -1.5 | Válido | El programa indica que se introduzca un elemento válido, puesto que solo acepta números enteros. |
| App2.7N | numIntroducido == letra | A, a, aa | Válido | El programa indica que se introduzca un elemento válido, puesto que solo acepta números enteros. |
| App2.8N | numIntroducido == símbolo | >, \* | Válido | El programa indica que se introduzca un elemento válido, puesto que solo acepta números enteros. |
| App2.9N | numIntroducido == “” | (vacío) | Válido | El programa indica que se introduzca un elemento válido, puesto que solo acepta números enteros. |
| App2.10N | numIntroducido == máximo valor (999999999) | 999999998, 999999999, 1000000000 | Válido | Como son valores que superan el rango de entre 1 y 100, el programa nos indica que introduzcamos un número que se encuentre en dicho rango. |
| App2.11N | numIntroducido == mínimo valor | -99999998,  -99999999,  -100000000 | Válido | Como son valores que menores al rango de entre 1 y 100, el programa nos indica que introduzcamos un número que se encuentre en dicho rango. |

Vemos que con todas estas pruebas hemos cubierto la totalidad del código de nuestro método, con lo que ya no sería necesario realizar más pruebas.

Sin embargo, realizaremos las pruebas de caja blanca, cubriendo las distintas partes de las que se compone nuestro método. En total hay 2 bucles que se enumeran en rojo de la forma que vemos en la siguiente imagen:



App2.3B

App2.2B

App2.1B

Para cubrir todos los bucles, será necesario realizar únicamente las siguientes pruebas:

|  |  |  |
| --- | --- | --- |
| **Prueba** | **Valor de entrada** | **Resultado** |
| App2.1B | 0 | No válido, no permite entrar al bucle al no ser un valor permitido (no está entre 1 y 100) |
| App2.2B | 1 | Válido |
| App2.3B | 3 | Válido |

A continuación, indicaremos las pruebas de caja negra que son cubiertas en cada prueba de caja blanca:

- App2.1B: Cubre las pruebas App2.4N, App2.5N, App2.11, y App2.10N.

- App2.2B: Cubre la prueba App2.3N.

- App2.3B: Cubre las pruebas App2.2N y App2.1N.