

**Portafolio de Evidencias Testing Fundamentals**

**Fundamentos de Software Testing**

Dalyn Yenisey Lara Cortés

06 de marzo de 2022

# Fundamentos de Software Testing

**A continuación, se te presentarán dos ejercicios. Escoge una opción “a” o “b” y lee detenidamente el requerimiento. Una vez escogido el ejercicio anota 3 posibles defectos o ambigüedades que encuentres en el requerimiento. Explica brevemente cómo podrías mejorarlo.**

* Cuenta de ahorro. Imagina que acabas de abrir una nueva cuenta de ahorros que te ofrece el 4% de interés al año. Estos ahorros debido a intereses, que no se cobran hasta finales de año, se te añaden al balance final de tu cuenta de ahorros. Escribir un programa que comience leyendo la cantidad de dinero depositada en la cuenta de ahorros, introducida por el usuario. Después el programa debe calcular y mostrar por pantalla la cantidad de ahorros tras el primer, segundo y tercer años. Redondear cada cantidad a dos decimales.
* Panadería. Una panadería vende barras de pan a $34 pesos cada una. El pan que no es el día tiene un descuento del 60%. Escribir un programa que comience leyendo el número de barras vendidas que no son del día. Después el programa debe mostrar el precio habitual de una barra de pan, el descuento que se le hace por no ser fresca y el coste final total.

***Ejercicio elegido: b.***

|  |  |
| --- | --- |
| Ambigüedades o defectos en el requerimiento | Cómo se puede mejorar |
| No indica si se debe considerar el número de barras de pan del día. | Definir si se utilizara el mismo programa para hacer la cuenta total o serán independientes |
| La última parte no indica si se debe mostrar el precio, descuento y coste final del total de barras de pan o únicamente de una. | Indicar el precio regular y el precio con descuento desde el inicio para mostrar el total de todas las barras al final. |
| El enunciado no indica si el precio final debe contemplar el precio de las barras sin descuento y con descuento. | Indicar el precio de cada tipo y al final el coste final. |

**Del requerimiento que escogiste anteriormente:**

1. Utilizando la herramienta Blockly implementa el programa.
2. Comenta qué defectos fuiste capaz de encontrar al terminar de implementar tu programa por tu propia cuenta.
3. Graba un video de máximo 1 minuto demostrando su funcionamiento.

|  |
| --- |
| Comentario de defectos encontrados por el mismo autor |
| Los tipos de datos no están validados, por lo que los resultados de los costos podrían fallar |
| La información mostrada es confusa y desordenada, por lo que es difícil de leer. |
| En caso de equivocarse al ingresar un dato, es necesario esperar a finalizar el programa para iniciar el proceso nuevamente. |

**Pregunta con los demás integrantes del curso quién hizo el ejercicio diferente al que escogiste y solicita una captura de pantalla y aplica lo siguiente:**

1. Implementa el programa en Blockly tal como te pasaron la solución de forma distinta a como lo realizaste.
2. Anota todos los defectos que fuiste capaz de encontrar.
3. Copia y pega imágenes de los defectos que fuiste capaz de encontrar.

|  |
| --- |
| Comentario de defectos encontrados por el mismo autor |
| Al momento de mostrar el descuento se agregan muchos decimales a las cantidades. |
| Permite ingresar caracteres siguiendo el procedimiento dando NaN como resultado. |
| Permite añadir un número decimal de panes. |

# Software testing y su relación en el ciclo de vida de desarrollo del software (SDLC)

Dado el siguiente requerimiento realiza los siguientes puntos:

1. Crea el programa en la herramienta online JDOODLE utilizando Java.
2. Identifica los defectos o ambigüedades que puedas encontrar en el requerimiento y llena la tabla mostrada más abajo.
3. Copia y pega el código.
4. Graba un video de 1 minuto demostrando su funcionalidad.

La pizzería Bella Napoli ofrece pizzas vegetarianas y no vegetarianas a sus clientes. Los ingredientes para cada tipo de pizza aparecen a continuación:

* Ingredientes vegetarianos: Pimiento, Hongos y Espinaca.
* Ingredientes no vegetarianos: Pepperoni, Jamón y Salchicha.

Escribir un programa que pregunte al usuario si quiere una pizza vegetariana o no, y en función de su respuesta le muestre un menú con los ingredientes disponibles para que elija. Sólo se puede elegir 3 ingredientes además de la mozzarella y el tomate que están en todas las pizzas. Al final se debe mostrar por pantalla si la pizza elegida es vegetariana o no y todos los ingredientes que lleva.

|  |  |
| --- | --- |
| Ambigüedades o defectos en el requerimiento | ¿Cómo se puede mejorar? |
| El requerimiento no especifica si es posible hacer una pizza con menos de 3 ingredientes, sin contar la mozzarela y el tomate | El enunciado podría indicar que el número máximo de ingredientes es 3 y el mínimo es 1. |
| No se especifica si es posible repetir un ingrediente más de una vez. | El enunciado podría indicar que se puede agregar un ingrediente más de una vez o que no es posible. |
| No se indica si es posible comprar más de una pizza por cliente. | El enunciado debería especificar si únicamente se evaluará una pizza por cliente o es posible hacer una evaluación de más de una pizza por cliente. |

Código:

|  |
| --- |
| import java.util.Scanner;  public class MyClass {  public static void main(String args[]) {  int[] ing = new int[3];  String[] vg = {"Pimientos", "Hongos", "Espinaca"};  String[] nvg= {"Pepperoni", "Jamon", "Salchica"};    Scanner sc = new Scanner(System.in);  System.out.println("Bienvenido a la Pizzeria Bella Napoli\n\n Por favor, elija el tipo que desea comprar:\n1.Vegetariana\n2.No Vegetariana");  String pz = sc.nextLine();    if(pz.equals("1")){  System.out.println("\n\nSeleccione los ingredientes que desea agregar a su pizza:\n1.Pimientos\n2.Hongos\n3.Espinaca");  for(int i = 0; i < 3; i++){  System.out.println("Ingrese el número que elige como ingrediente numero "+(i+1)+":");  ing[i]= sc.nextInt()-1;  }  System.out.println("\n\nLa pizza que eligió es Vegetariana");  System.out.println("Los ingredientes que eligio son:");  for(int i = 0; i < 3; i++){  System.out.println(vg[ing[i]]);  }      }else{  System.out.println("Seleccione los ingredientes que desea agregar a su pizza:\n1.Pepperoni\n2.Jamon\n3.Salchica");  for(int i = 0; i < 3; i++){  System.out.println("Ingrese el número que elige como ingrediente numero "+(i+1)+":\n\n");  ing[i]= sc.nextInt()-1;  }  System.out.println("La pizza que eligió No es Vegetariana");  System.out.println("Los ingredientes que eligio son:");  for(int i = 0; i < 3; i++){  System.out.println(nvg[ing[i]]);  }  }    }  } |

Del requerimiento anterior descomponlo en historias de usuario. Para este ejercicio se propone descomponer el requerimiento en al menos 5 historias de usuario con sus criterios de aceptación. Recuerda que una historia de usuario describe una pequeña parte de la funcionalidad de una especificación.

|  |  |
| --- | --- |
| Historia de usuario | Criterios de aceptación |
| Yo como cliente quiero que desde el inicio el programa me pregunte qué tipo de pizza deseo para saber lo que prefiero. | * El programa pregunta qué tipo de pizza desea el cliente. * Se almacena la información proporcionada por el usuario. |
| Yo como cliente quiero que el programa me muestre únicamente los ingredientes que prefiero para poder elegir entre ellos. | * El sistema muestra la información disponible basado en la preferencia del usuario. * Ingredientes vegetarianos: Pimiento, Hongos y Espinaca. * Ingredientes no vegetarianos: Pepperoni, Jamón y Salchicha. |
| Yo como cliente deseo que el programa me especifique si hay ingredientes por defecto en la pizza y cuantos puedo elegir. | El sistema muestra una leyenda indicando que:   * Todas las pizzas llevan mozzarela y tomate. * El cliente puede elegir máximo tres ingredientes para su pizza. |
| Yo como cliente quiero que el programa recuerde los ingredientes que elegí para saber cómo está conformada mi pizza. | * El sistema almacena cada una de las selecciones del usuario, tomando en cuenta el número máximo de ingredientes. |
| Yo como como cliente deseo que el sistema me muestre qué tipo de pizza elegí y los ingredientes que lleva para saber qué recibiré. | * El sistema muestra una leyenda con el tipo de pizza y los ingredientes que lleva sin contar el queso y el tomate. |

**Del ejercicio anterior:**

1. Implementa nuevamente el código utilizando una historia de usuario a la vez (preferentemente una función por historia de usuario).
2. Integra las partes del código.
3. Copia y pega el código.
4. Graba un video de 1 minuto demostrando su funcionamiento.

Código ver 2

|  |
| --- |
| import java.util.Scanner;  public class MyClass {  static int max =3;  static Scanner sc = new Scanner(System.in);    public static boolean val\_opt(String x, int max\_opt){  for(int i = 1; i <= max\_opt; i++){  if(x.equals(Integer.toString(i))){  return true;  }  }  System.out.println("\nHa ingresado una opción no válida. Intentelo de nuevo\n");  return false;    }    public static String pizza\_typ(){  String pz="";  do{  System.out.println("Por favor, elija el tipo que desea comprar:\n\n1.Vegetariana\n\n2.No Vegetariana");  pz = sc.nextLine();  }while(!val\_opt(pz,2));    if(pz.equals("1")){  return "Vegetariana";  }else{  return "No Vegetariana";  }    }  public static String[] pizza\_ing(String opt){  String[] vg = {"Pimientos", "Hongos", "Espinaca"};  String[] nvg = {"Pepperoni", "Jamon", "Salchica"};  if(opt.equals("Vegetariana")){  return vg;  }else{  return nvg;  }    }    public static void show\_ing(int n, String[] ing){  System.out.println("\n\nEstos son los ingredientes disponibles para su pizza:");  for(int i = 0; i<n; i++){  System.out.println((i+1)+". "+ ing[i]);  }  System.out.println();  }    public static int[] select\_ing(int n){  int[] ing = new int[n];  for(int i = 0; i < n; i++){  String aux = "";  do{  System.out.println("Ingrese el número del ingrediente que elige como número "+(i+1)+":");  aux = sc.nextLine();  }while(!val\_opt(aux,3));    ing[i]= Integer.parseInt(aux);    }  return ing;  }    public static void ing\_inf(int n){  System.out.println("Todas las pizzas contienen mozzarela y tomate. Usted tiene derecho a elegir "+ n + " ingredientes además de los ya mencionados.\n");  }    public static void pizza\_inf(String[] ing, int[] sel, int n, String type){  System.out.println("\n\nLa pizza que eligió es: "+type);    System.out.println("Los ingredientes que eligio son:");  for(int i = 0; i < n; i++){  System.out.println(ing[sel[i]-1]);  }  }  public static void main(String args[]) {    System.out.println("Bienvenido a la Pizzeria Bella Napoli\n\n");  String type = pizza\_typ();  String[] ing = pizza\_ing(type);  show\_ing(max,ing);  ing\_inf(max);  int[] sel = select\_ing(max);  pizza\_inf(ing,sel,max,type);  }  } |

Llena la siguiente tabla para responder las siguientes preguntas.

|  |  |
| --- | --- |
| Pregunta | Respuesta |
| ¿En cuál de los ejercicios fue más fácil identificar ambigüedades en las especificaciones? | En el segundo ejercicio. En el ejercicio uno identifiqué varios errores pero se notaron con más facilidad en el segundo. |
| ¿En cuál de los primeros dos ejercicios tuviste más claro cómo elaborar tu código y qué actividades desempeñar? ¿Y por qué? | En el segundo, porque las actividades a desarrollar estaban ya definidas. |
| ¿Por qué concluir que la comunicación es esencial en las actividades de testing y desarrollo? | Porque de esa forma se tiene claro lo que debe hacerse y se comprueba que el resultado es el correcto. |
| ¿Qué etapas del ciclo de vida del desarrollo de software identificas durante esta actividad? | Análisis: se realizó el análisis de cómo se debía abordar el problema a resolver.  Diseño: realizó el diseño lógico de la aplicación.  Implementación: realización del código.  Pruebas: Por cada función agregada se realizaron pruebas. |
| ¿Dónde piensas que se involucraría el proceso de testing de acuerdo a la actividad que se realizó? | En todas las etapas porque es necesario confirmar que se está haciendo bien el trabajo desde el inicio. En este caso específico, comenzando asegurar que se entendieron los requerimientos, el programa da el resultado esperado y cada parte funciona de forma adecuada. |

# Técnicas de creación de pruebas

**Ejercicio 1**

Teniendo el siguiente escenario de negocio, elige entre una de las opciones dependiendo ¿cuál sería la mejor sugerencia?:

"Equipo, tenemos que hacer unas pruebas que nos solicite el cliente, al parecer la aplicación que estamos desarrollando debe funcionar en 4 navegadores web diferentes y en 2 sistemas operativos y en redes con distinta capacidad de velocidad de conexión, además, debemos garantizar una cobertura de código del 85 % e indicar cuáles combinaciones no fueron cubiertas."

1. Como tenemos conocimiento suficiente, podríamos iniciar una prueba de cobertura de sentencia, y cumplir lo solicitado.
2. Aplicar unas pruebas exploratorias, sería suficiente para cumplir con el porcentaje solicitado.
3. Probar solo en un navegador y suponer que, si funciona bien, en todos estará igual.
4. Crear condiciones y combinaciones en una tabla de decisión con los navegadores y sistemas operativos, e iniciar investigación del código fuente para aplicar otras técnicas de caja blanca.

**Actividad 2**

Elaborar la técnica de prueba que requiera conveniente para el siguiente enunciado:

"El cliente requiere probar los siguientes requerimientos del aplicativo electoral"

* Ciudadanos aptos para votar teniendo en cuenta el criterio de la edad
* Ciudadanos aptos para votar solo si tiene la cédula registrada en la ciudad o tiene un certificado de desplazamiento forzado

***Tabla de Decisiones***

Aceptado:

Rechazado:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Criterios | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Tiene edad para votar | V | V | V | V | F | F | F | F |
| Cédula Registrada | V | V | F | F | V | V | F | F |
| Certificado de desplazamiento forzado | V | F | V | F | V | F | V | F |
| Resultados |  |  |  |  |  |  |  |  |

***Valores Límite:***

* Ciudadanos aptos para votar teniendo en cuenta el criterio de la edad.
  + Números inferiores a -1. (x<0, [-1,-2,-n]
  + Números superiores a 150. El número máximo debe ser definido por el cliente, debe consultarse esta cantidad por ser una ambigüedad. (x>150, [151,152,151+n])
  + Números decimales. ([1.5, 2.5,…, n.n])
  + Letras. [a,b,c]
  + Caracteres. especiale. [¡,”,#]
* Ciudadanos que tienen cédula registrada en la ciudad.
  + Cualquier entrada distinta a la respuesta solicitada. Este criterio debe ser definido.
* Ciudadanos que tienen cédula registrada en la ciudad.
  + Cualquier entrada distinta a la respuesta solicitada. Este criterio debe ser definido.

**Ejercicio 3**

Caso de uso: Para una solicitud aprobada de tarjeta de crédito, se debe asignar una categoría, dependiendo del ingreso reportado por el cliente.

Condiciones:

Si el cliente tiene un ingreso de $800 USD a $2000 USD, se le asignar una TDC DORADA

Si el cliente tiene un ingreso de $2001 USD a $3500 USD, se le asignará una TDC PLATINO

Si el cliente tiene un ingreso de $3001 USD ... se le asignará una TDC NEGRA.

Particiones de equivalencia:

* Válidas
  + TDC DORADA: números entre 800-2,000. *800≥x≤2000.* [800,801,…,2000]
  + TDC PLATINO: números entre 2001-3,000 y 3002-3500. *2001≥x≤300 and 3000≥x≤3500.*
  + TDC NEGRA: *x = 3001.* [3001]
* Inválidas
  + Números inferiores a 800. x<800. [799,798,…,-n].
  + Números superiores a 3500.x>3500. [3501,3502,…,3501+n]

Valores límite:

* Números negativos, letras, caracteres especiales.

**Ejercicio 4**

Un algoritmo valida los siguientes campos numéricos:

* Valores menores que 40 son rechazados.
* Valores entre 42 y 71 son aceptados.
* Valores mayores o iguales a 92 son rechazados.

Particiones de equivalencia:

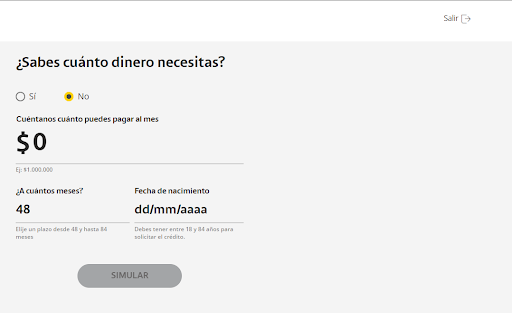
* Válidas
  + Aceptados: números en el rango entre 42 a 71. *42≥x≤71.* [42, 43, …, 71].
* Inválidas
  + Números inferiores a 40. x<40. [39, 38, …, -n].
  + Números superiores a 92. x>92. [93, 94, …, 93+n]
* Indefinidos
  + 41.
  + Números en el rango en el rango entre 72 a 91. *72≥x≤91*

Valores límite:

* Números negativos, letras, caracteres especiales.

**Ejercicio 5**

Describe los casos de uso que probaría, si le pidieran una prueba exploratoria de la siguiente pantalla.



Campo 1: ¿Cuánto puedes pagar al mes?

Campo 2: ¿Cuánto puedes pagar al mes?

Campo 3: ¿A cuántos meses?

Campo 4: Fecha de nacimiento.

**Casos de uso para prueba:**

* Seleccionar sí en el campo 1.
  + Precondición: El usuario entra en la página para solicitar su préstamo.
  + Input:
    - Sí
  + Output: Al usuario se le deberían solicitar campos distintos a si seleccionara no.
  + Flow:

|  |
| --- |
| 1. Seleccionar sí en el campo 1.    1. El usuario ingresa en el link del sitio.    2. El usuario selecciona la opción sí en el campo 1.    3. El usuario visualiza un formulario distinto al mostrado en la imagen. |

* El usuario ingresa valores inválidos campo 2.
  + Precondicion: El usuario entra en la página para solicitar su préstamo.
  + Actor: usuario.
  + Input:
    - No.
    - Abc
    - 48
    - 01/01/1997
  + Output: “El valor del campo dos es incorrecto. Ingrese un valor válido”.
  + Flow:

|  |
| --- |
| 1. El usuario ingresa valores en el campo 2.    1. El usuario ingresa en el link del sitio.    2. El usuario selecciona la opción no en el campo 1.    3. El usuario ingresa letras u caracteres especiales en el campo dos.    4. El usuario ingresa 48 en el campo 3.    5. El usuario selecciona la fecha 01/01/1997.    6. El usuario obtiene el mensaje “El valor del campo dos es incorrecto. Ingrese un valor válido” |

* El usuario ingresa valores inválidos campo 3.
  + Precondicion: El usuario entra en la página para solicitar su préstamo.
  + Actor: usuario.
  + Input:
    - No.
    - 1000
    - Abc
    - 01/01/1997
  + Output: “El valor del campo tres es incorrecto. Ingrese un valor válido”.
  + Flow:

|  |
| --- |
| 1. El usuario ingresa valores en el campo 3.    1. El usuario ingresa en el link del sitio.    2. El usuario selecciona la opción no en el campo 1.    3. El usuario ingresa 1000 en el campo 3.    4. El usuario ingresa letras u caracteres especiales en el campo dos.    5. El usuario selecciona la fecha 01/01/1997.    6. El usuario obtiene el mensaje “El valor del campo tres es incorrecto. Ingrese un valor válido” |

* El usuario ingresa valores inválidos campo 4.
  + Precondicion: El usuario entra en la página para solicitar su préstamo.
  + Actor: usuario.
  + Input:
    - No.
    - 1000
    - 48
    - 01/01/1996
  + Output: “El valor del campo tres es incorrecto. Ingrese un valor válido”.
  + Flow:

|  |
| --- |
| 1. El usuario ingresa valores en el campo 4.    1. El usuario ingresa en el link del sitio.    2. El usuario selecciona la opción no en el campo 1.    3. El usuario ingresa 1000 en el campo 3.    4. El usuario ingresa letras u caracteres especiales en el campo dos.    5. El usuario selecciona la fecha 01/01/1996.    6. El usuario obtiene el mensaje “El valor del campo cuatro es incorrecto. Ingrese un valor válido” |

* El usuario ingresa valores válidos en todos los campos.
  + Precondicion: El usuario entra en la página para solicitar su préstamo.
  + Actor: usuario.
  + Input:
    - No.
    - 1000
    - 48
    - 01/01/1997
  + Output: Resultado esperado.
  + Flow:

|  |
| --- |
| 1. El usuario ingresa valores en el campo 3.    1. El usuario ingresa en el link del sitio.    2. El usuario selecciona la opción no en el campo 1.    3. El usuario ingresa 1000 en el campo 3.    4. El usuario ingresa letras u caracteres especiales en el campo dos.    5. El usuario selecciona la fecha 01/01/1997.    6. El usuario obtiene el mensaje “El valor del campo tres es incorrecto. Ingrese un valor válido” |