ϣÀ

UNIVERSIDAD NACIONAL DE LA MATANZA

Departamento de Ingeniería e Investigaciones Tecnológicas

**Análisis de Software**

**Documentación Técnica**

**Números A Texto**

Grupo Nº 5

**Integrantes:**

|  |  |  |
| --- | --- | --- |
| **Apellido** | **Nombre** | **DNI** |
| Baca | Iván | 36.536.685 |
| Devoto | Federico | 37.038.974 |
| Osorio | Maximiliano | 34.306.104 |
| Sheila | Soria | 35.537.624 |

**Docentes:**

* + - **Lic. Marcelo D. Vinjoy**
    - **Ing. Roberto Landaburu**
    - **Lic. Gustavo Agustín**
    - **Lic. Enzo Del Ben**

Índice

* Descripción Funcional………………………………………. pág. nº 3
* Modelo de Calidad…………………………………………… pág. nº 4
* Métodos de Caja Blanca…………………………………… pág. nº 12
* Métodos de Caja Negra……………………………………. pág. nº 18
* Pruebas del Sistema Aplicadas…………………………. pág. nº 20
* Manual de Usuario…………………………………………… pág. nº 23
* Cubierta de CD…………………………………………………. pág. nº 30
* Encuesta de Satisfacción de Usuario con comparación sobre el Modelo de Calidad…………………………………………… pág. nº 32

Descripción Funcional del Producto

**Propósito**

El propósito del software es proveer a los usuarios una herramienta fácil e intuitiva para transformar números a texto.

**Beneficios**

Permite a los usuarios obtener rápidamente el valor de un número en formato de texto simplemente ingresando el mismo en formato numérico.

**Características**

El producto brinda una interfaz de comportamiento intuitivo, para que el manejo de la misma por parte de los usuarios sea cómodo.

La información de entrada debe ser un valor de tipo numérico, entero y mayor a cero. Además admite un valor máximo de 20.000.000.

**MODELO DE CALIDAD**

**ALGORITMO PARA OBTENCIÓN DEL NIVEL DE CALIDAD ALCANZADO**

Características y Subcaraterísticas Utilizadas:

1. Funcionabilidad

a. Seguridad de Acceso

b. Exactitud de los resultados

2. Eficiencia

a. Utilización de recursos

b. Comportamiento frente al tiempo

3. Fiabilidad

a. Tolerancia a fallos

b. Capacidad de recuperación de errores

4. Mantenibilidad

a. Capacidad del código de ser analizado

b. Capacidad del código de ser cambiado

c. Estabilidad

5. Usabilidad

a. Capacidad de ser entendido

b. Capacidad de ser operado

c. Capacidad de ser atractivo para el usuario

6. Portabilidad

a. Adaptabilidad

b. Instalabilidad

Niveles de Puntuación

El algoritmo de calidad consiste en evaluar cada subcaracterística como mala, regular o buena, dependiendo de los criterios que se mencionarán más adelante. Las combinaciones de los resultados de la evaluación de las subcaracterísticas determinarán el puntaje que se obtendrá de la característica. El paso siguiente es promediar los puntajes anteriores para obtener un valor general de la característica. Cada una de ellas podría tener un peso que se utilizaría en un promedio general ponderado de las 6 características del modelo de calidad.

El promedio general obtenido determinaría el nivel de calidad obtenido que podría ser ACEPTABLE o NO ACEPTABLE.

Criterios de Evaluación de Métricas

1. Funcionabilidad

**Seguridad de acceso**

Descripción:

Es la capacidad del producto software para asegurar la integridad de los datos y la confidencialidad de estos.

Características a medir:

Encriptación de datos

Inicio de sesión de usuarios

Control de perfil de usuarios

Evaluación:

Mala [0] No cumple con ninguna característica.

Regular [1] Cumple con 1 y/o 2 características.

Buena [3] Cumple con 3 características.

**Exactitud de resultados**itud de los Resultados

Descripción:

Es la capacidad del producto software para proporcionar los resultados con el grado necesario de precisión.

Evaluación:

Mala [>=10-3] Los resultados tienen un error del orden de 10-3 o superior.

Regular [10-4;10-6] Los resultados tienen un error del orden entre10-4y 10-6.

Buena [<=10-7] Los resultados tienen un error del orden de10-7 o inferior

2. Eficiencia

**Utilización de recursos**ecursos

Descripción:

Se evaluará la eficiencia del producto software de acuerdo al porcentaje de uso de procesador que realice.

Evaluación:

Mala [55; 100] 55% o más de uso de procesador.

Regular [20; 54] 20% a 54% de uso de procesador.

Buena [0; 19] 10% o menos de uso de procesador.

**Comportamiento en el tiempo**omportamiento frente al Tiempo

Descripción:

Se evaluará el tiempo que está el producto software sin informarle al usuario del estado en que se encuentra la solicitud que realizó.

Evaluación:

Mala [>=6] El producto está 5 o más segundos sin informar al usuario del estado de la solicitud.

Regular [3; 6] El producto está entre 2 y 4 segundos sin informar al usuario del estado de la solicitud.

Buena [1; 2] El producto está menos de 1 segundo sin informar al usuario del estado de la solicitud.

3. Fiabilidad

**Tolerancia a fallos**lerancia a Fallos

Descripción:

Es la capacidad del producto software de mantener la integridad de los datos cuando se producen fallas del sistema.

Características a medir:

Cuando sucede un error se protegen los datos procesados.

Se realiza un log de actividades que el sistema estaba haciendo.

Se realiza un log de la/s falla/s ocurridas.

Evaluación:

Mala [0] No cumple con ninguna característica.

Regular [1, 2] Cumple con 1 y/o 2 características.

Buena [3] Cumple con 3 características.

Sub

**Capacidad de recuperación de errores** Capacidad de Recuperación de Errores

Descripción:

Es la capacidad del sistema de reanudar sus actividades cuando se producen errores críticos.

Características a medir:

El sistema reanuda las actividades si se produce una falla crítica.

Reanuda sus actividades y vuelve al estado en que estaba.

El sistema reanuda las actividades y todas sus funcionalidades se ejecutan sin errores.

Evaluación:

Mala [0] No cumple con ninguna característica.

Regular [1, 2] Cumple con 1 y/o 2 características.

Buena [3] Cumple con 3 características.

4. Mantenibilidad

**Capacidad del código para ser analizado** Analizado

Descripción:

Para evaluar la capacidad que tiene el código para ser analizado se tiene en cuenta el porcentaje de comentarios que posee el mismo por cada método y en general.

Evaluación:

Mala [0; 20] 20% o menos del código comentado.

Regular [21; 49] Entre 21% y 49% del código comentado.

Buena [>=50] 50% o más del código comentado

**Capacidad del código para ser cambiado** Capacidad del Código para biado

Descripción:

Para evaluar la capacidad que tiene el código para ser cambiado se tomarán cuenta la complejidad ciclomática del método.

Evaluación:

Mala [21] La complejidad ciclomática es mayor o igual a 21.

Regular [11; 20] La complejidad ciclomática es entre 11 y 20.

Buena [1; 10] La complejidad ciclomática es menor o igual a 10.

**Estabilidad**

Descripción:

Para determinar la estabilidad del software se evalúa el promedio de fallas que presenta el producto por prueba.

Evaluación:

Mala [5] El software presenta un promedio 5 o más errores por prueba.

Regular [2; 4] El software presenta un promedio entre 2 y 4 errores por prueba.

Buena [0; 1] El software presenta un promedio entre 0 y 1 error por prueba.

5. Usabilidad

**Capacidad de ser entendido**racterística Capacidad de ser Entendido

Descripción:

Capacidad que posee el software, para ayudar a los usuarios ante una determinada situación donde se necesite asistencia.

Características a medir:

Posee ayuda contextual sobre menús y botones de acción.

Manual de usuario incorporado al sistema como un menú dedicado.

Evaluación

Mala [0] No cumple con ninguna característica.

Regular [1] Cumple con 1 característica.

Buena [2] Cumple con 2 características.

S

**Capacidad para ser operado**acidad de ser Operado

Descripción:

Es la capacidad del producto software de ser utilizado sin asistencia adicional. Se evalúa qué requiere el usuario para operar correctamente el producto.

Evaluación:

Mala [1] El usuario requiere consultar a personal especializado para operar el producto software.

Regular [2] El usuario requiere ayuda contextual y manual de uso para operar el producto software.

Buena [3] El usuario opera el producto software sin asistencia.

**Capacidad de ser atractivo para el usuario** Capacidad de ser Atractivo para el Descripción:

Es la agrupación correcta de funcionalidad del producto software en su interfaz gráfica, desde su agrupación lógica hasta el número promedio de pasos para alcanzar una función o contenido específico.

Evaluación:

Mala [5] 5 o más pasos promedio sin organización de categoría.

Regular [3; 4] Entre 3 y 4 pasos promedio y distribuidos en categorías.

Buena [1; 2] 1 o 2 pasos promedio y distribuidos en categorías.

6. Portabilidad

**Adaptabilidad**ptabilidad

Descripción:

Es la capacidad del producto software de adaptarse a diferentes sistemas operativos sin cambiar su estructura interna o sin que implique reacciones negativas ante el cambio.

Evaluación:

Mala [1] Compatible con 1 sistema operativo.

Regular [2] Compatible con 2 sistemas operativos.

Buena [>=3] Compatible con 3 o más sistemas operativos.

**Instalabilidad**

Descripción:

El producto software debe poder ser instalado en una cantidad mínima de pasos.

Evaluación:

Mala [>8] El producto se instala en 8 o más pasos.

Regular [5; 7] El producto se instala entre 5 y 7 pasos.

Buena [1; 4] El producto se instala en 4 o menos pasos.

|  |  |
| --- | --- |
| NIVELES DE ACEPTACION | PUNTAJE DE ACEPTACION |
| SATISFACTORIO | >50 |
| MINIMAMENTE SATISFACTORIO | Entre 35 y 50 |
| NO SATISFACTORIO | <35 |

|  |  |  |  |
| --- | --- | --- | --- |
| Característica | Promedio puntuación | Ponderación | Puntaje general |
| Portabilidad | 1.48 | 7 | 10.4 |
| Eficiencia | 2 | 6 | 12 |
| Usabilidad | 1.3 | 5 | 6.66 |
| Mantenibilidad | 1.3 | 4 | 5.2 |
| Fiabilidad | 57 | 3 | 1.7 |
| Funcionabilidad | 1 | 2 | 2 |

**Total = 37.96**

***Métricas de Complejidad***

***Complejidad Ciclomática de Mc Cabe***

Se realiza el procedimiento brindando por el método de Mc Cabe, para calcular la complejidad ciclomática del método “decmiles” perteneciente al programa “NumerosATexto”.

A continuación se detalla el código de dicha funcionalidad:

private String decmiles(int numero){

if (numero == 10000)

num\_letradm = "diez mil";

if (numero > 10000 && numero <20000){

flag=1;

num\_letradm = decena(numero/1000).concat("mil ").concat(centena(numero%1000)); }

if (numero >= 20000 && numero <100000){

flag=1;

num\_letradm = decena(numero/1000).concat(" mil ").concat(miles(numero%1000)); }

if (numero < 10000)

num\_letradm = miles(numero);

return num\_letradm;

}

Usamos la condición de los Nodos predicados, ya que permite calcular el número ciclomático sin la necesidad de realizar el grafo. Para ello, identificamos los **IF.**

1. if (numero > 10000 && numero <20000) esto se puede reescribir como 2 IF separados, uno dentro de otro. Aquí tenemos **2 nodos predicados.**
2. if (numero == 10000). Aquí tenemos **1 nodo predicado.**
3. if (numero < 10000). Aquí tenemos **1 nodo predicado.**
4. if (numero >= 20000 && numero <100000) esto se puede reescribir como 2 IF separados, uno dentro de otro. Aquí tenemos **2 nodos predicados.**

En total obtuvimos 4 nodos predicados. Según Mc Cabe, el número ciclomático es igual a los Nodos predicados (de aquí en adelante se los llamará NP) más 1.

Por lo tanto tenemos: NP + 1 = NC (número ciclomático)

6 + 1 = 7

En conclusión, el **NC = 7.**

Para poder corroborar esto, realizaremos el grafo de la función

En primer lugar, detallamos los Nodos a utilizar en el grafo, asignándole un número de referencia.

* if (numero == 10000) **Nodo1**
* num\_letradm = "diez mil" **Nodo2**
* if (numero > 10000 && numero <20000)
  + if(numero > 10000) **Nodo3**
  + if(numero <20000) **Nodo4**
* flag=1;

num\_letradm = decena(numero/1000).concat("mil ").concat(centena(numero%1000)); **Nodo5**

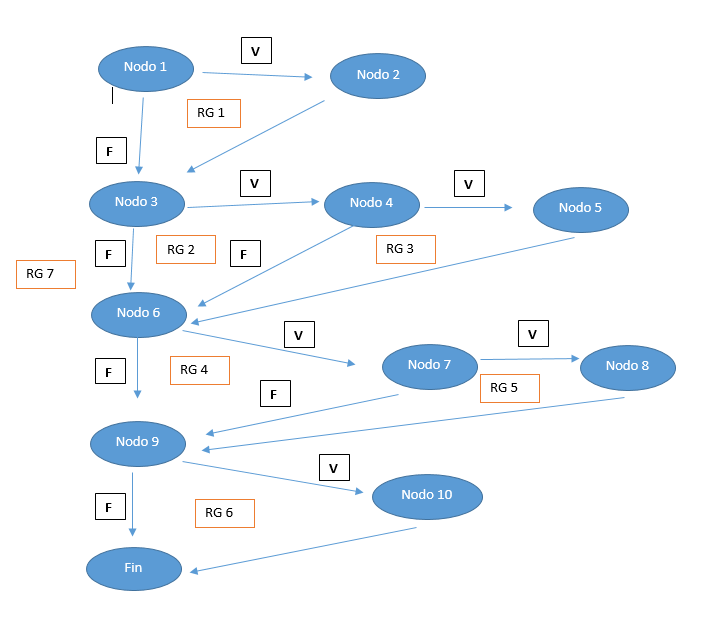
* if (numero >= 20000 && numero <100000)
  + if(numero >= 20000) **Nodo6**
  + if(numero <100000) **Nodo7**
* flag=1;

num\_letradm = decena(numero/1000).concat(" mil ").concat(miles(numero%1000));

**Nodo8**

* if (numero < 10000) **Nodo9**
* num\_letradm = miles(numero); **Nodo10**
* return num\_letradm; Fin de programa **FP**

Luego dibujamos el grafo.

Ahora corroboramos el número ciclomático calculado al inicio. Para ello tenemos 3 condiciones más:

1. NC = cantidad de caminos = **7**
   1. 1 3 6 9 FP
   2. 1 2 3 6 9 FP
   3. 1 2 3 4 6 9 FP
   4. 1 2 3 4 5 6 9 FP
   5. 1 2 3 4 5 6 7 9 FP
   6. 1 2 3 4 5 6 7 8 9 FP
   7. 1 2 3 4 5 6 7 8 9 10 FP
2. NC = cantidad de regiones = **7**
3. NC = Aristas - Nodos + 2 = **16 – 11 + 2 = 7**

**En conclusión, la complejidad ciclomática de la función es 7. Esto también cumple con la condición que debe ser menor o igual a 10.**

**Casos de Prueba**

A partir del método de Mc Cabe realizado anteriormente, definimos un caso de prueba por cada camino linealmente independiente.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Camino Mc Cabe | Número de Caso | Entrada | Resultado Esperado | Resultado Obtenido |
| **1** | **1** | **No se puede armar Caso de prueba para este camino** |  |  |
| **2** | **2** | número = 10000 | diez mil | diez mil |
| **3** | **3** | **No se puede armar Caso de prueba para este camino** |  |  |
| **4** | **4** | **No se puede armar Caso de prueba para este camino** |  |  |
| **5** | **5** | **No se puede armar Caso de prueba para este camino** |  |  |
| **6** | **6** | **No se puede armar Caso de prueba para este camino** |  |  |
| **7** | **7** | **No se puede armar Caso de prueba para este camino** |  |  |

Vemos que este método no es eficiente para esta función, debido a que su TER es muy bajo.

**TER = 1/7 = 14.28 %**

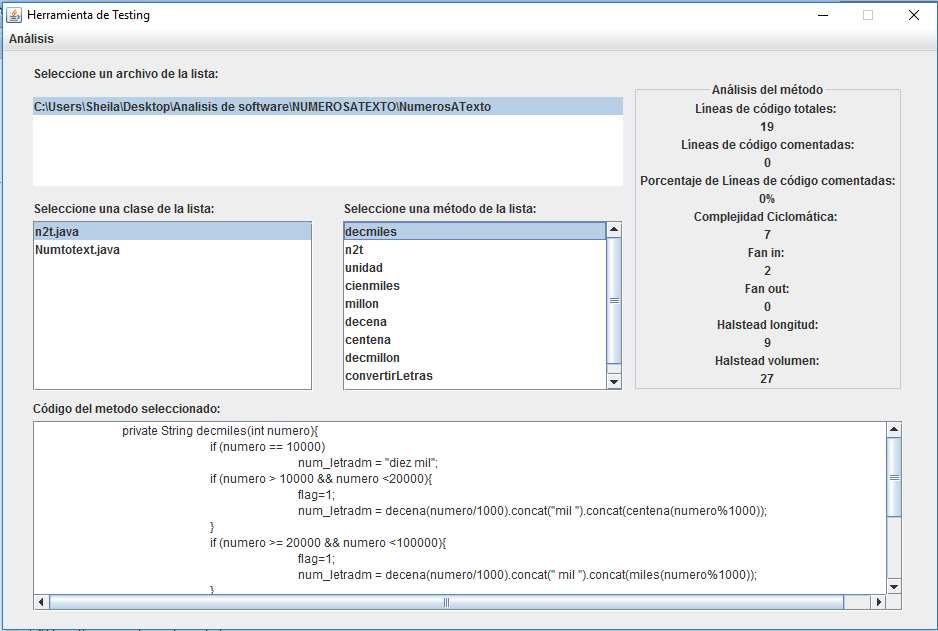
Por esto, decidimos generar otro conjunto de caminos linealmente independientes:

1. 1 3 6 9 FP
2. 1 2 3 6 9 FP
3. 1 3 4 6 9 FP
4. 1 3 6 7 9 FP
5. 1 3 6 9 10 FP
6. 1 3 4 5 6 9 FP
7. 1 3 6 7 8 9 FP

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Camino Mc Cabe | Número de Caso | Entrada | Resultado Esperado | Resultado Obtenido |
| **1** | **1** | **No se puede armar Caso de prueba para este camino** |  |  |
| **2** | **2** | número = 10000 | diez mil | diez mil |
| **3** | **3** | **No se puede armar Caso de prueba para este camino** |  |  |
| **4** | **4** | **No se puede armar Caso de prueba para este camino** |  |  |
| **5** | **5** | número = 500 | quinientos | quinientos |
| **6** | **6** | número = 10500 | diez mil quinientos | diez mil quinientos |
| **7** | **7** | número = 40000 | cuarenta mil | cuarenta mi |

**Aquí el TER es del 57,14%(4 casos probados / 7 casos totales).**

Utilizamos la herramienta de Testing para evaluar el método “decmiles”.



***Pruebas de Caja Negra***

***Particiones o Clases de equivalencia***

Se realiza el procedimiento brindado por la técnica de particiones o clases de equivalencia para dividir el campo de entrada del método “decmiles” perteneciente al programa “NumerosATexto” en clases de datos de los que se pueden derivar casos de prueba.

**Clases de equivalencia:**

Establecemos los casos de pruebas según las siguientes reglas.

1. Identificamos las condiciones de entrada del programa, es decir, restricciones de formato o contenido de los datos de entrada.
2. A partir de ellas, se identifican clases de equivalencia que pueden ser

-de datos validos

-de datos no validos

1. Utilizamos distintas reglas que ayudan a identificar clases de equivalencia.

4. Asignamos un número único a cada clase de equivalencia.

|  |  |  |  |
| --- | --- | --- | --- |
| **Condición de entrada** | **Clases validas** | **Clases no validas** | **Reglas** |
| Tipo de dato entero | Número entero (1) | Número no entero (7) | Booleano |
| Valor número | [1 ; 20.000.000] (2) | <1 (8)  > 20.000.000 (9) | Rango |
| Cantidad de dígitos del numero | Entre [1 ; 8] dígitos.(3) | Ningún dígito (10)  Más de 8 dígitos (11) | Cantidad |
| Tipo de dato | Caracteres numéricos (4) | Caracteres no numéricos (12) | Booleano |
| Formato con Punto (.) | Numero entero sin Punto (.) entre sus dígitos (5) | Numero entero Con Punto (.) entre sus dígitos (13) | Booleano |
| Formato con Coma (,) | Numero entero sin Coma (,) entre sus dígitos (6) | Numero entero con Coma (,) entre sus dígitos (14) | Booleano |

**Casos de prueba:**

Establecemos los casos de pruebas según las siguientes reglas.

1. [H](http://www.cropscience.bayer.cl/soluciones/fichaproducto.asp?id=43)asta que todas las clases de equivalencia hayan sido cubiertas por (incorporadas a) casos de prueba, se tratara de escribir un caso de prueba que cubra tantas clases validas no incorporadas como sea posible.
2. [H](http://www.cropscience.bayer.cl/soluciones/fichaproducto.asp?id=43)asta que todas las clases de equivalencia no validas hayan sido cubiertas por casos de prueba, escribir un caso para una única clase valida sin cubrir.

|  |  |  |  |
| --- | --- | --- | --- |
| **Caso n°** | **Valor Entrada** | **Salida** | **Clases cubiertas** |
| 1 | 100 | “Cien” | 1, 2, 3, 4, 5, 6 |
| 2 | 100,4 | “ERROR” | 7 |
|  | 34,9 | “ERROR” | 14 |
| 3 | 0 | “ERROR” | 8 |
| 4 | 200000000 | “ERROR” | 9 |
|  | 465789234 | “ERROR” | 11 |
| 5 | “ ” | “ERROR” | 10 |
| 6 | “A” | “ERROR” | 12 |
| 7 | 100.4 | “ERROR” | 14 |

**Análisis de Valores Límite:**

Complementa a la partición equivalente mediante la selección de uno o más elementos tal que los márgenes se sometan a prueba.

Teniendo en cuenta la siguiente clase de equivalencia establecida previamente generamos los casos teniendo en cuenta que para un rango de valores se deben generar casos para los extremos del rango y casos no validos para situaciones justo mas allá de ellos.

|  |  |  |  |
| --- | --- | --- | --- |
| **Condición de entrada** | **Clases validas** | **Clases no validas** | **Reglas** |
| Valor número | [1 ; 20.000.000 ]  (1) (2) | <1 (3)  > 20.000.000 (4) | Rango |

**Casos de prueba:**

Establecemos los casos de pruebas teniendo en cuenta el dominio de entrada y el espacio de salida.

|  |  |  |  |
| --- | --- | --- | --- |
| **Caso n°** | **Valor Entrada** | **Salida** | **Clases cubiertas** |
| 1 | 1 | “Uno” | 1 |
| 2 | 20000000 | “Veinte millones” | 2 |
| 3 | -3 | “ERROR” | 3 |
| 4 | 21000000 | “ERROR” | 4 |

# PRUEBAS DEL SISTEMA

Las siguientes pruebas tienen como objetivo verificar la funcionalidad del producto, comprobar que las funcionalidades sean las esperadas en función de los requisitos del sistema, las mismas Avalan la calidad del sistema.

## Pruebas de Recuperación

Estas pruebas consisten en forzar el fallo del software y comprobar que la recuperación se lleva a cabo de manera correcta, devolviendo al sistema a un estado coherente.

Las siguientes son las pruebas de recuperación para el software NumeroATexto.

|  |  |  |
| --- | --- | --- |
| **Campo** | **Entrada** | **Salida obtenida** |
| Ingreso\_Numero | 100.4 | Error. Debe ingresar un valor entero positivo distinto de “0” |
| Ingreso\_Numero | 100,4 | Error. Debe ingresar un valor entero positivo distinto de “0” |
| Ingreso\_Numero | 0 | Error. Debe ingresar un valor entero positivo distinto de “0” |
| Ingreso\_Numero | “ “ | Error. Debe ingresar un valor entero positivo distinto de “0” |
| Ingreso\_Numero | -100 | Error. Debe ingresar un valor entero mayor que “0” |
| Ingreso\_Numero | 300000000 | Error. Debe ingresar un valor entre 1 y 20000000 |
| Ingreso\_Numero | A | Error. Debe ingresar un valor entre 1 y 20000000 |

Luego de realizar las pruebas de recuperación, verificamos que el sistema retorna a su estado coherente.

Permitiendo volver a su estado original y permitiéndonos poder ingresar un nuevo valor numérico a convertir.

## Pruebas de Seguridad

Intentan verificar que los mecanismos de protección incorporados al sistema lo protegerán.

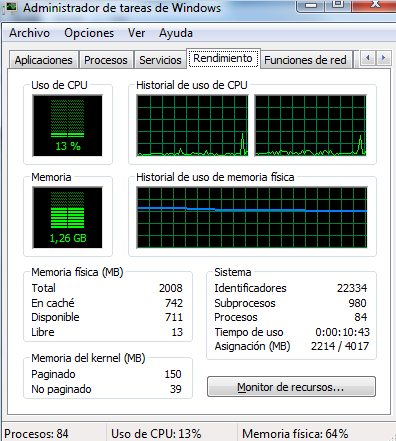
El acceso al sistema contará con una protección mínima mediante el uso de contraseña. Una vez ingresada dicha contraseña el usuario podrá hacer pleno uso del sistema.

## Pruebas de Resistencia

Estas pruebas están diseñadas para que el sistema requiera recursos en cantidad, frecuencia o volumen anormales.

El producto no consume grandes cantidades de recursos y hace poco uso de la CPU. Luego de llevar a cabo la prueba de resistencia se verificó que el manejo de datos por parte del software es realizado de manera óptima, no alterando ninguna de sus funcionalidades.

## 



## Pruebas de Rendimiento

Es inaceptable que el software proporcione las funciones requeridas fuera de las condiciones de rendimiento exigidas.

Luego de haber sometido al software a las anteriores pruebas de recuperación y resistencia, se comprobó que el rendimiento no se vio afectado, respondiendo dentro de las condiciones de rendimiento exigidas.

## Pruebas de Aceptación

Mediante el uso del cuestionario, se pudo verificar que el sistema cumple con todos los requisitos indicados, habiendo obtenido la aprobación de los usuarios del producto.

**FiveSoft ®**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_­­­­­­­­­\_\_\_\_\_\_\_\_**

*Manual de Usuario*

*Primero pasos con* ***Número a Texto***

Índice

1. ¿Qué es **Número a Texto**?......................................................................................24

2. Requisitos.................................................................................................................25

3. ¿Cómo lo instalo?.....................................................................................................26

4. La interfaz **Número a Texto**....................................................................................28

5. Licencia de software.................................................................................................29

1. ¿Qué es **Número a Texto**?

Es es un software que permite facilitar la escritura de números, transformando un número ingresado en formato numérico a su representación en palabras.

2. Requisitos

Para utilizar **Número a Texto** debe contar como mínimo con el siguiente hardware:

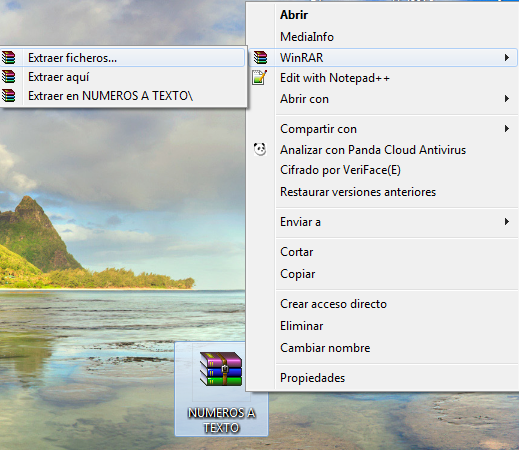
* Procesador 1GHz 32/64
* 1GB RAM
* 2GB disponible en Disco Duro
* Video 32 MB

El software funciona tanto en Windows XP o superior, como en sistemas operativos de la familia UNIX, Ubuntu 12.04 y Debian 8 o superior.

3. ¿Cómo lo instalo?

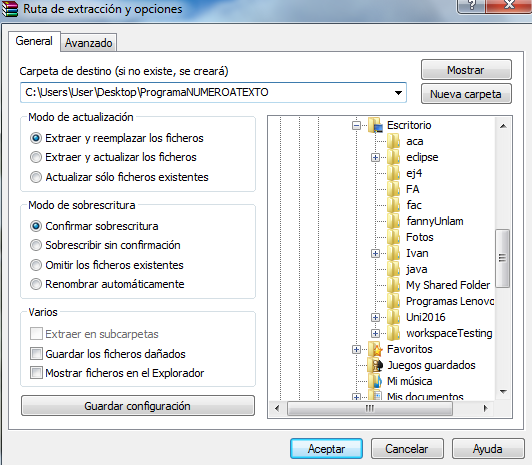
Es muy sencillo, siguiendo los siguientes pasos:

**Paso 1** - Descomprimir el ejecutable.

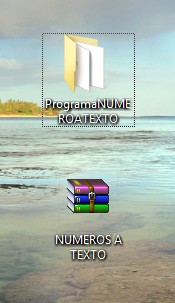


**Paso 2** – Determinar la dirección donde se va a descomprimir el archivo, en este caso de ejemplo la ruta será: **C:\Users\User\Desktop\ProgramaNUMEROATEXTO**

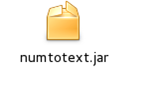
Luego seleccionar “ACEPTAR”.

****

**Paso 3** – Quedara creada la carpeta “ProgramaNUMEROATEXTO”, dar doble click para ingresar.



**Paso 4 –** Tendremos listo el ejecutable llamado de la siguiente forma. Hacer doble click sobre el ejecutable y listo, la interfaz **Número a Texto** estará lista para usar.



**Paso 5 –** Se deberá ingresar un Numero y el sistema devolverá su resultado en texto.



4. La interfaz **Número a Texto**

Llegado a este punto usted verá la interfaz que le permite ingresar un número, una vez ingresado, pulse entrar y el texto que representa el valor ingresado será visualizado por pantalla, el texto puede ser copiado y el sistema queda listo para que usted ingrese otro número.  
  
Una vez que termino de usar el software simplemente cierre desde Archivo->Salir o desde la X en el extremo de la pantalla.

**5. Licencia de software**

No se permite la reproducción total o parcial del material que usted ha recibido, ya sea en formato DVD o partir de la descarga desde el sitio oficial FiveSoft®.

**Número a Texto** es un software de FiveSoft® Argentina, Copyright © todos los derechos reservados.





Encuesta de satisfacción

**Producto**:

**Número a texto**.

**Alcance**:

La encuesta se realizará a 10 personas de diversas edades y ocupaciones.

**Características de la encuesta**:

La encuesta está dividida en 6 características diferentes que el sistema posee con tres preguntas que deben ser respondidas mediante las opciones Bueno/Mala/Regular.

Las mismas se corresponden a:

* Funcionalidad: Seguridad de acceso, cumplimiento de la funcionalidad.
* Fiabilidad: Tolerancia a fallos, capacidad de recuperación.
* Usabilidad: Capacidad para ser entendido, aprendido y operado de manera simple.
* Eficiencia: Rapidez de respuesta, utilización de recursos.
* Mantenibilidad: Estabilidad, capacidad para ser probado.
* Portabilidad: Adaptabilidad a distintos entornos, instabilidad, capacidad para ser reemplazado.

**Procedimiento:**

La encuesta será realizada en persona, dónde se les brindará una hoja para seleccionar la respuesta que consideren más apropiada.

Encuesta

**Funcionabilidad**

1. ¿Considera que la herramienta cumple con las necesidades requeridas por usted?

Mala Buena Regular

1. ¿Cómo evaluaría la seguridad del sistema?

Mala Buena Regular

1. ¿Le parece que los resultados son exactos?

Mala Buena Regular

**Eficiencia**

1. ¿Cómo describiría el tiempo de respuesta del software?

Mala Buena Regular

1. Mientras utiliza el software, ¿Puede utilizar otros sistemas en forma simultánea y correctamente?

Mala Buena Regular

1. ¿Cómo evaluaría el uso del almacenamiento del software?

Mala Buena Regular

**Fiabilidad**

1. Si el software se detiene, ¿Cómo le resulta reiniciarlo?

Mala Buena Regular

1. ¿Puede recuperar los datos luego del que el software se detenga?

Mala Buena Regular

1. ¿Cómo considera en cuanto a si son entendibles las advertencias lanzadas por el programa?

Mala Buena Regular

**Usabilidad**

1. ¿Se siente cómodo utilizando el sistema?

Mala Buena Regular

1. ¿Cómo evaluaría la apariencia del software?

Mala Buena Regular

1. ¿Considera que la distribución de los menús o listas son lógicas o de acceso simple?

Mala Buena Regular

**Portabilidad**

1. ¿Cómo evaluaría el proceso de instalación?

Mala Buena Regular

1. ¿Cómo considera la posibilidad de usar el software en varios sistemas operativos?

Mala Buena Regular

1. ¿Cómo considera la experiencia en la cantidad de pasos para la instalación del programa?

Mala Buena Regular

**Resultados de la encuesta:**

Caso 1

Ocupación: Masajista

Edad: 24

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Funcionalidad | Fiabilidad | Usabilidad | Eficiencia | Mantenibilidad | Portabilidad |
| 1 | Buena | Regular | Buena | Buena | Buena | Buena |
| 2 | Regular | Buena | Buena | Buena | Regular | Regular |
| 3 | Buena | Buena | Regular | Buena | Buena | Buena |

Caso 2

Ocupación: Docente

Edad: 50

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Funcionalidad | Fiabilidad | Usabilidad | Eficiencia | Mantenibilidad | Portabilidad |
| 1 | Buena | Regular | Buena | Buena | Buena | Regular |
| 2 | Buena | Buena | Buena | Buena | Regular | Buena |
| 3 | Regular | Buena | Buena | Buena | Buena | Regular |

Caso 3

Ocupación: Enfermera

Edad: 26

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Funcionalidad | Fiabilidad | Usabilidad | Eficiencia | Mantenibilidad | Portabilidad |
| 1 | Buena | Regular | Buena | Buena | Buena | Buena |
| 2 | Regular | Regular | Mala | Buena | Buena | Buena |
| 3 | Regular | Buena | Mala | Buena | Buena | Regular |

Caso 4

Ocupación: Desocupado

Edad: 18

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Funcionalidad | Fiabilidad | Usabilidad | Eficiencia | Mantenibilidad | Portabilidad |
| 1 | Buena | Regular | Buena | Buena | Buena | Buena |
| 2 | Buena | Buena | Regular | Buena | Buena | Buena |
| 3 | Regular | Buena | Regular | Buena | Buena | Buena |

Caso 5

Ocupación: Psicóloga

Edad: 25

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Funcionalidad | Fiabilidad | Usabilidad | Eficiencia | Mantenibilidad | Portabilidad |
| 1 | Buena | Regular | Buena | Buena | Buena | Buena |
| 2 | Buena | Buena | Regular | Buena | Buena | Buena |
| 3 | Buena | Buena | Buena | Buena | - | Buena |

Caso 6

Ocupación: Arquitecta

Edad: 25

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Funcionalidad | Fiabilidad | Usabilidad | Eficiencia | Mantenibilidad | Portabilidad |
| 1 | Buena | Regular | Buena | Buena | Buena | Buena |
| 2 | Buena | Buena | Buena | Buena | Buena | Regular |
| 3 | Buena | Buena | Regular | Buena | Buena | Regular |

Caso 7

Ocupación: Desarrollador Android

Edad: 25

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Funcionalidad | Fiabilidad | Usabilidad | Eficiencia | Mantenibilidad | Portabilidad |
| 1 | Buena | Regular | Buena | Buena | Buena | Regular |
| 2 | Buena | Buena | Regular | Buena | Regular | Regular |
| 3 | Mala | Buena | Regular | Buena | Buena | Buena |

Caso 8

Ocupación: Estudiante

Edad: 15

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Funcionalidad | Fiabilidad | Usabilidad | Eficiencia | Mantenibilidad | Portabilidad |
| 1 | Buena | Regular | Buena | Buena | Buena | Buena |
| 2 | Buena | Regular | Regular | Buena | Buena | Regular |
| 3 | Buena | Buena | Regular | Buena | - | Buena |

Caso 9

Ocupación: Ama de casa

Edad: 21

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Funcionalidad | Fiabilidad | Usabilidad | Eficiencia | Mantenibilidad | Portabilidad |
| 1 | Buena | Mala | Buena | Buena | Buena | Buena |
| 2 | Buena | Buena | Mala | Buena | Buena | Buena |
| 3 | Regular | Buena | Regular | Buena | Buena | Buena |

Caso 10

Ocupación: Desarrollador Java

Edad: 23

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Funcionalidad | Fiabilidad | Usabilidad | Eficiencia | Mantenibilidad | Portabilidad |
| 1 | Buena | Regular | Buena | Buena | Buena | Buena |
| 2 | Buena | Buena | Regular | Buena | Buena | Buena |
| 3 | Buena | Buena | Regular | Buena | - | Mala |

Análisis de los resultados obtenidos:

\*Se toman las mismas condiciones especificadas en el modelo de calidad

Bueno: [2] Regular: [2] Malo: [0]

Sin Contestar: No cuentan para el promedio

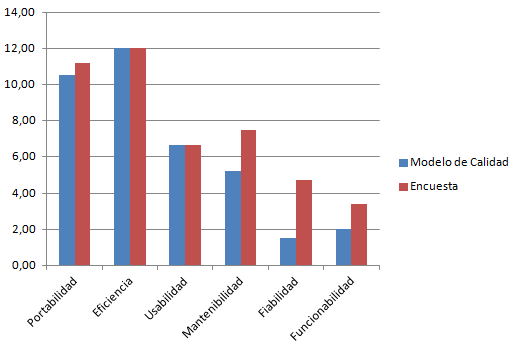
|  |  |  |  |
| --- | --- | --- | --- |
| Característica | Promedio puntuación | Ponderación | Puntaje general |
| Portabilidad | 1.6 | 7 | 11.2 |
| Eficiencia | 2 | 6 | 12 |
| Usabilidad | 1.3 | 5 | 6.66 |
| Mantenibilidad | 1.9 | 4 | 7.5 |
| Fiabilidad | 1.57 | 3 | 4.7 |
| Funcionabilidad | 1.7 | 2 | 3.4 |

**Total = 45.46**

|  |  |
| --- | --- |
| NIVELES DE ACEPTACION | PUNTAJE DE ACEPTACION |
| SATISFACTORIO | >50 |
| MINIMAMENTE ACEPTABLE | Entre 35 y 50 |
| NO SATISFACTORIO | <35 |

El software se considera según encuesta: **Mínimamente Aceptable**

**Gráfico Comparativo:**



-Para las características de Fiabilidad y Funcionabilidad los valores obtenidos a partir de la encuesta difieren notablemente de los obtenidos aplicando el modelo de calidad definido debido a que con este último se realizo una evaluación más completa y detallada de las características y funcionalidades del sistema basándose en un número mayor de pruebas que evalúan en profundidad el funcionamiento del sistema a diferencia de la experiencia que tiene el usuario utilizando el mismo, dando como resultado un puntaje más bajo.