Universidad de San Carlos

De Guatemala.

Introducción a la Programación 1.

Sección D.

Manual Técnico.

Nombre: Nelson Daniel Cruz.

Carne: 200915606.

Índice

1. ……Índice.

2. ……Introducción.

3. ……..Objetivos.

4. ……..Análisis.

5. ……Listado de clases.

6. ……Listado de métodos.

7. ……..……Resultados.

8. ……..……Resultados.

9. ……..……Resultados.

10. ……..……...….Resultados.

11. ……...……Resultados.

12. ……...Resultados.

13. ……...Resultados.

14. ……..…… Diagrama de Clase.

15. …..…… Explicación Diagrama de clases.

16-25……..….Clases del software.

26-27-28……..glosario, Conclusión y Bibliografía.

Introducción

En el mundo de la informática día con día, las necesidades que se presentan son más exigentes, con lo cual deben surgir profesionales preparados de manera eficiente que se dediquen al desarrollo y la implementación de software.

Tal es el caso expuesto en el presente; nace con la necesidad de la implementación de una hoja de cálculo que pueda realizar operaciones básicas, dicho software deberá ser utilizado en determina empresa, para que los usuarios, que hagan uso del software, tengan una herramienta útil, versátil y fácil de usar.

Que el usuario se manejen en un ambiente agradable, que el software tenga una imagen cálida, y que su uso sea más eficiente que programas utilizados para dichas operaciones.

Objetivos

- Como objetivo primordial, la realización de un software que implemente lo que requiere el usuario, que este sea lo más versátil y eficiente para que el usuario encuentre solución a la problemática buscada.
- Como objetivo personal, la utilización de lo aprendido en el lenguaje java, las aplicaciones que se puedan implementar para la realización del software, aplicando lo aprendido en el transcurso del curso.

Contenido.

Análisis:

El usuario requiere de un programa donde se pueda utilizar una hoja de cálculo en la cual las personas que utilizaran el software puedan realizar operaciones básicas entre celdas, tanto matemáticas como operaciones de texto.

El software requiere que el programa cuente con una barra de menús, de fácil uso, que tenga el menú editar para que las personas que sean las encardas de usarlo pueda cambiar el color de letra, el color de las celdas y que despliegue en pantalla una ventana donde puedan ellos elegir el color que ellos prefieren, El menú nuevo que servirá para abrir una nueva hoja de cálculo, pero que al cerrar la hoja de cálculo pregunte al usuario si desea guardar la hoja de cálculo en la que se estaba trabajando o no, y el menú ayuda donde este contara con los manuales de uso del software,

tanto con el manual técnico, como el manual de usuario y la información del programa.

Una solución inicial para poder realizar el software requerido por el usuario, es la realización de un listado concreto de las clases que conforman el programa, para poder realizar un diagrama UML de clases.

De igual forma los métodos o funciones a utilizar para poder realizar el software.

Listado de Clases:

- Class Exl (Main).
- Class Celdas.
- Class Operaciones.

Clases Atómicas: Estas clases dependen de otras clases.

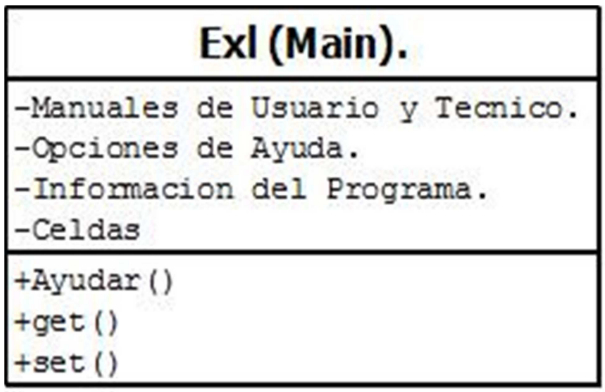
- Class Operaciones Matemáticas.
    - Class Suma.
    - Class Resta.
    - Class Multiplicación.
    - Class Exponen ciar.

- Class Operaciones de Texto
    - Class Concatenar.
    - Class SubString.
    - Class Sustraer.

Listado de Métodos:

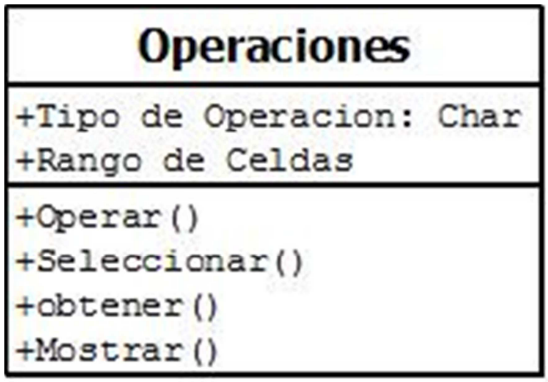
- getText.
- Sustraer.
- multiplicar.
- sumar.
- restar.
- Concatenar.
- setText.
- substring.
- color.
- Demostrar.

Clase Exl: El nombre de la clase principal es Exl donde se muestra los diferentes objetos necesarios para empezar a realizar el software, tal es el caso que mostramos esta clase con sus atributos que son los menús, la ayuda, los manuales del software, y la hoja de cálculo, con métodos como lo son abrir el software, cerrarlo, borrar, ayudar, editar.

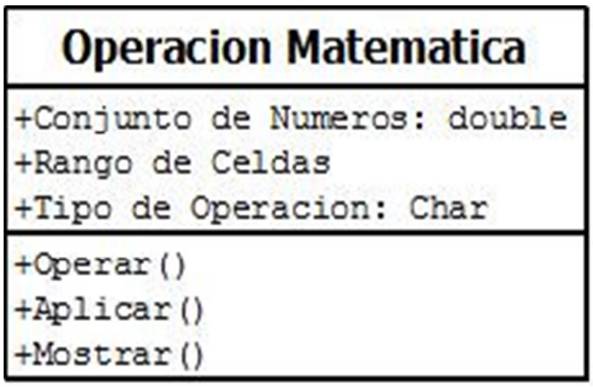


Clase Celdas: La función de esta clase es escribir formulas introducir datos y que atreves de ella se puedan realizar las operaciones requeridas por el usuario.

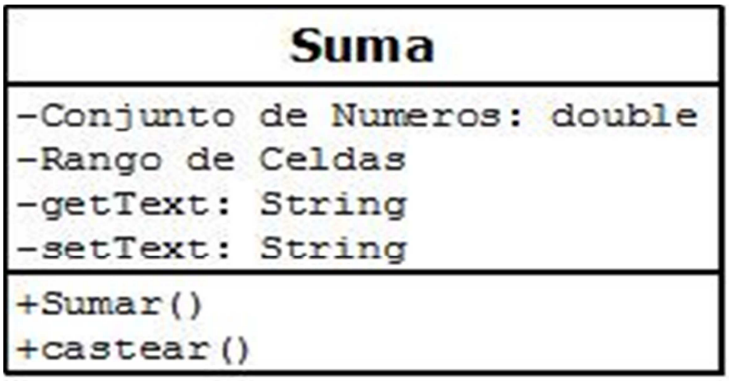
Clase Operaciones: El objetivo de esta clase es que por medio de ella, se agrupen a todos los objetos que van a tener el metodo de realizar todas y cada una de las operaciones que se van a poder realizar entre celdas.



Clase Operaciones Matemáticas: El objetivo de esta clase es agrupar a los objetos encargados de realizar las sumas, resta, multiplicaciones y exponenciaciones.

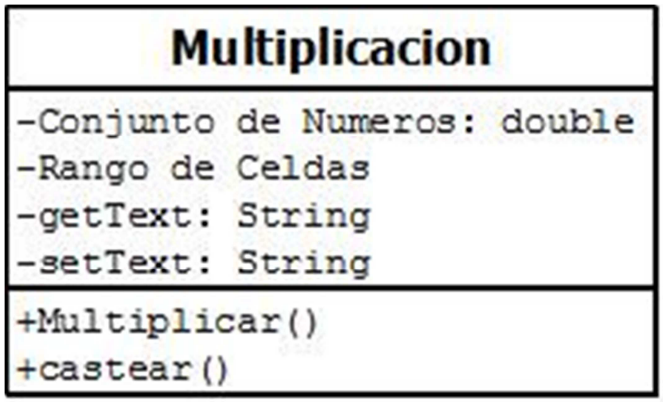


Clase suma: Esta clase contiene a los objetos que se van a encargar de realizar las sumas entre celdas, atributos principales un rango de celdas y conjunto de números para poder realizar su función o método sumar.

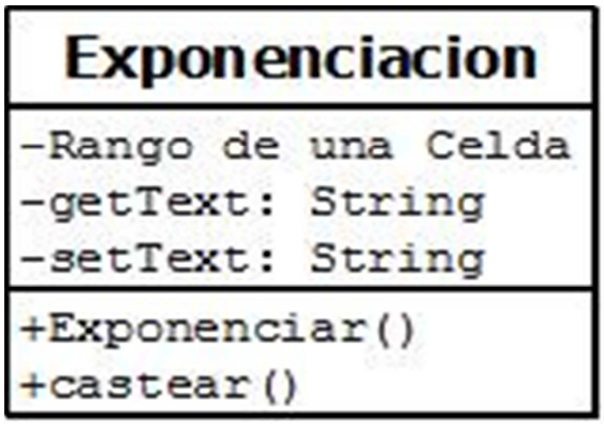


Clase Resta: Esta clase agrupa a los objetos necesarios para realizar las restas entre celdas, atributos necesarios es tener un conjunto de números en un determinado rango de celda, su función o método a realizar es la restar.

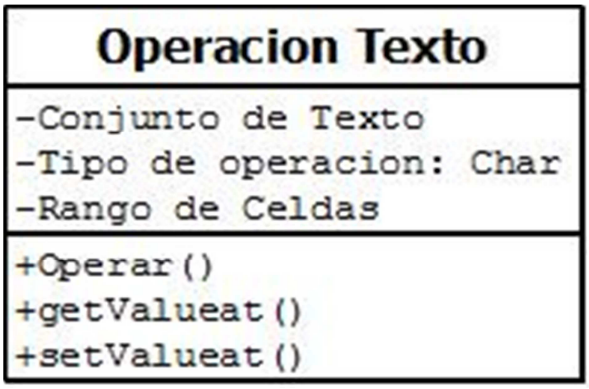
Clase Multiplicación: En esta clase la función que realiza es multiplicar los números que se encuentran en un rango determinado de celdas.



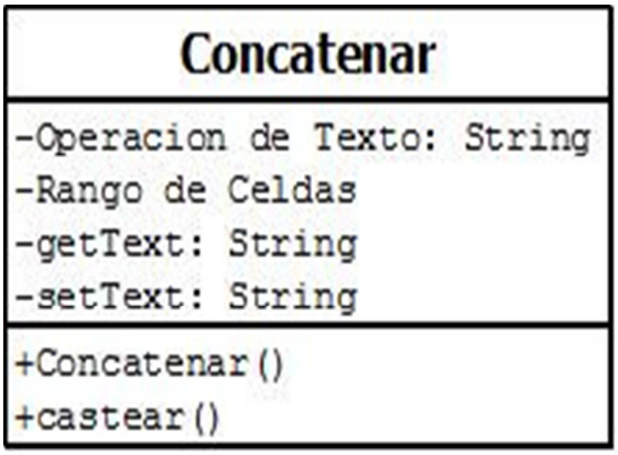
Clase Exponen ciar: La función de esta clase es elevar al cuadrado un número que se encuentre en un rango de celdas.



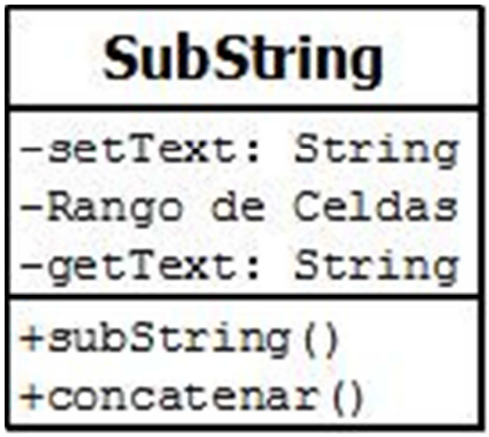
Clase Operaciones de Texto: En esta clase, se agrupara a los objetos que se encargaran de realizar todas las operaciones de texto que el software requiera.



Clase Concatenar: Esta clase tiene como función concatenar como su nombre lo indica, agrupara el texto que se encuentre en un rango de celdas.



Clase SubString: Esta Clase tiene como función, armar nuevas palabras, con palabras escritas en celdas.



Clase Sustraer: Esta clase como su nombre lo indica su función es sustraer; la letra de una palabra en la ubicación que se le pida al software.

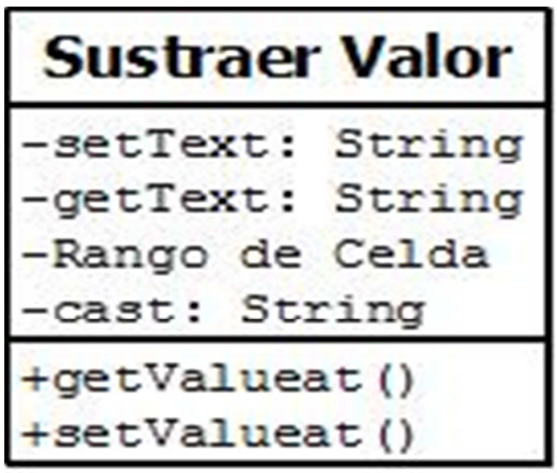
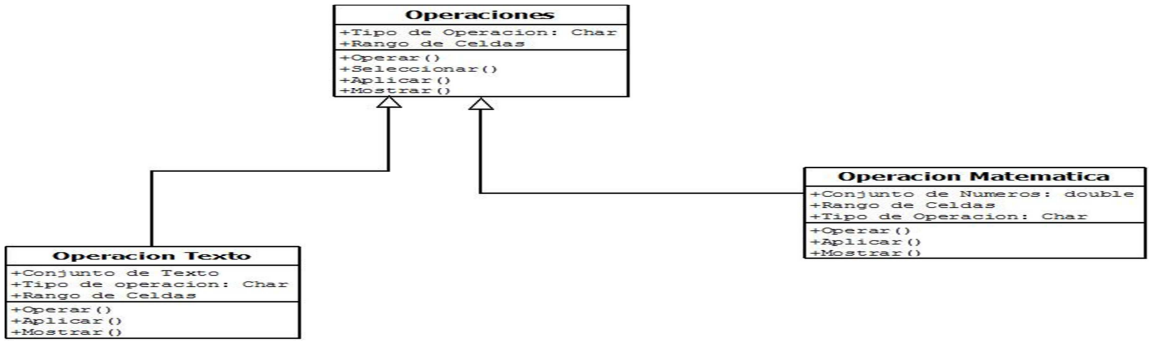
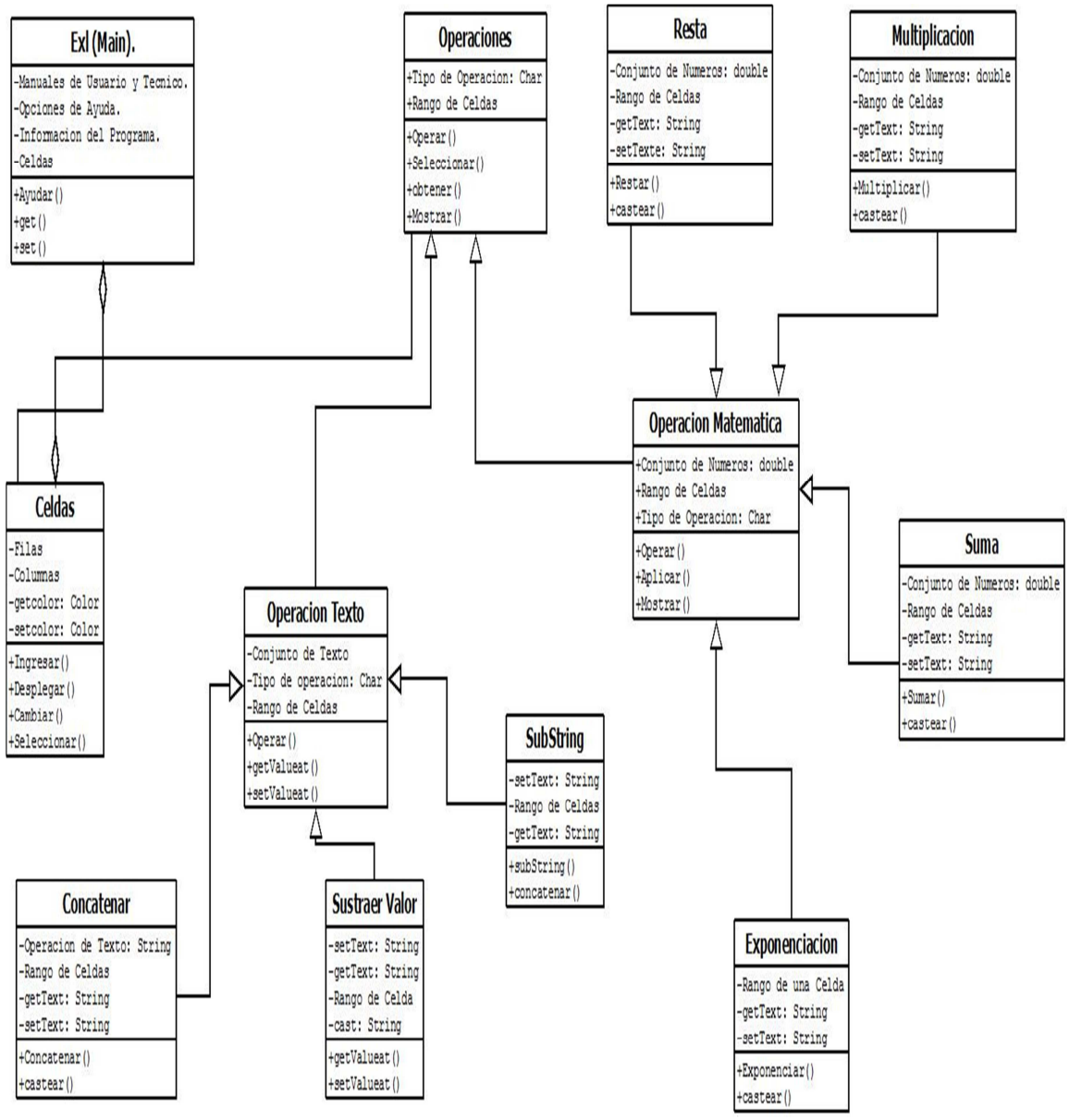


Diagrama de clases donde clases hijas heredan métodos de clases padre, el caso de la clase Operaciones, donde las clases Operaciones Matemáticas y Operaciones de Texto, heredan los métodos de la clase padre.



Diseño: Diagrama de clases:



Explicación Diagrama de Clases:

El diagrama de clases mostrado en la imagen de arriba; parte de 1 clase principal llamada XCL, de la cual dependen dos clases llamadas “celdas” y “operaciones”; dependen por que la relación que tienen, es que, si una clase no existe no pueden existir las otras, o no hay sentido en el diagrama, de estas dos clases heredan los métodos dos clases llamadas “clase Operaciones Matemáticas” y “clase Operaciones de Texto”, clases que tienen relación por asociación por composición con clases llamadas “suma”, “resta”, “multiplicación”, “exponenciación”, “concatenación”, “SubString”, “sustraer”, por que únicamente pueden ser llamadas atreves de las clases “operaciones matemáticas” y “operaciones de texto”.

Clase xcl: Clase en la que se maneja todo lo relacionado a eventos de todos los componentes.

```
package xcl;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.*;

import java.awt.*;

import java.awt.event.*;

import java.awt.Color;

import java.io.*;

import javax.swing.table.TableColumn;

import java.awt.Desktop;
/**
 * @author nelson
 */
public final class xcl extends JFrame{
//Variables de la clase xcl
```

```
public JMenu
men1,men2,men3,men4,men5,men6,men7,men8,men9,men1
0;

public JMenuBar barmen;

public Color color = Color.LIGHT_GRAY;

public Color color1 = Color.LIGHT_GRAY;

private JToolBar tol1;

private JLabel lab1;

private JTextField fil1,fil2;

public JTable table;

int a,b,c,d,e,f,g,h,i,j,k,l;

TableColumn tcol;

String
dato1,dato2,dato3,dato4,dato5,dato6,dato7,dato8,dato9,dato1
0,dato11,dato12,dato13,dato14,dato15,dato16;

//Constructor de la Clase xcl

public xcl(){

super ("nelson");
```

```
CrearTabla();

componentes();

//Maneja el Evento del Texfiel donde se coloca las formulas

ManejadorCampoTexto manejador = new
ManejadorCampoTexto();

fil2.addActionListener(manejador);

//Dar color a celdas

for (k=1;k<31;++k){

tcol = table.getColumnModel().getColumn(k);

tcol.setCellRenderer(new colorCel());

}

//Tamaño cierre y hacer Visible la tabla

setSize(800,600);

setVisible(true);

setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

}

//Metodo donde se crea la tabla
```

```
public void CrearTabla(){
final tabl dftm = new tabl();
table = new JTable(dftm);
dftm.setValueAt ("Nelson", 0, 1);
for (h = 0; h<31;h++){
for (f = 1; f<31; f++){
System.out.println(""+dftm.getValueAt(h,f));
}
}
//Visible el scrol de abajo y poder selecionar columna
table.setColumnSelectionAllowed(true);
table.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);
table.doLayout();
//Agregarle un ScrollPane a Tabla
JScrollPane barras = new JScrollPane(table);
barras.setBounds(25,60,975,545);
add(barras);
```

```
//encabezados de la tabla

dftm.setColumnIdentifiers(new String[] { "", "A", "B", "C",
"D","E","F","G","H","I","J","K","L","M","N","O","P","Q","R","S","T
","U","V","W","X","Y","Z","AA","AB","AC","AD"});

dftm.setColumnIdentifiers(new String[] { "", "a", "b", "c",
"d","e","f","g","h","i","j","k","l","m","n","o","p","q","r","s","t","u","v"
,"w","x","y","z","aa","ab","ac","ad"});

//se crea la numeracion de todas las filas en la columna no
editable

for(int fil = 0; fil < 60; fil++)

table.setValueAt(fil + 1, fil, 0);

table.getColumnModel().getColumn(0).setCellRenderer(table.
getTableHeader().getDefaultRenderer());

//Visualizar valor de celda con un click

table.addMouseListener(new MouseAdapter()

  {

@Override

public void mouseClicked(MouseEvent e)

{
```

```
int fila = table.rowAtPoint(e.getPoint());
int columna = table.columnAtPoint(e.getPoint());
if ((fila > -1) && (columna > 0))
fil2.setText((String) dftm.getValueAt(fila,columna));      }
   });
```

Clase tabl: En esta clase se implementa defaultTableModel, el cual nos sirve para controlar todo lo relacionado a la tabla.

```
package xcl;
import java.util.ArrayList;
import javax.swing.table.DefaultTableModel;
/**
 * @author nelson
 */
//Modelo de datos
public final class tabl extends DefaultTableModel{
private ArrayList<Object[]> data;
```

```
//Constructor de clase tabl
public tabl(){
        super(60,33);
        Iniciar_Tabla();
    }
//Iniciar tabla
    public void Iniciar_Tabla(){
    for (int row=0; row < super.getRowCount(); row++){
    for(int col=0; col<super.getColumnCount(); col++){
    super.setValueAt("", row, col);
    super.getValueAt(row, col);
            }
      }
    }
//Columna no editable
    @Override
public boolean isCellEditable(int fil, int col){
```

```
if (0 == col){
return false;
    }
for (int a=1; a<=33; a++){
if (a == col)
return true;
}
return super.isCellEditable(fil, col);
};

//Hacignar tipo de valor por Columna
    @Override
public Class getColumnClass(int columna)
   {
    int c;
    for (c = 1 ; c<31; c++){
    if (columna == 0) return Integer.class;
```

```
    if (columna == c) return String.class;
        }
return String.class;
}
//corchete final de la clase
}
```

Clase MainPrincipal:

En esta clase se llama el constructor de la clase encargada de crear todos los componentes y los métodos utilizados.

```
package xcl;
/**
 * @author nelson
 */
public class MainprincipalXcl {
public static void main (String [] args){
xcl uno = new xcl();
}
}
```

Glosario:

Software: Software se refiere al equipamiento lógico o soporte lógico de una computadora digital, y comprende el conjunto de los componentes lógicos necesarios para hacer posible la realización de tareas específicas; en contraposición a los componentes físicos del sistema, llamados hardware.

Celda: En las hojas de cálculo, una celda es el lugar donde se pueden introducir los datos. En hojas de cálculo como Microsoft Excel u Open Office.

UML: Un "lenguaje" visual que sirve para modelar código, en concreto programación orientada a objetos.

Funcionalidad: Coherencia entre las necesidades detectadas y los resultados que se obtienen con el uso del material.

Método: En la programación orientada a objetos, un método es una subrutina asociada exclusivamente a una clase (llamados métodos de clase o métodos estáticos) o a un objeto (llamados métodos de instancia).

Objeto: En el paradigma de programación orientada a objetos (POO, o bien OOP en inglés), un objeto se define como la unidad que en tiempo de ejecución realiza las tareas de un programa. También a un nivel más básico se define como la instancia de una clase.

Clase: Las clases son declaraciones o abstracciones de objetos, lo que significa, que una clase es la definición de un objeto. Cuando se programa un objeto y se definen sus características y funcionalidades, realmente se programa una clase.

Operadores: En matemáticas, un operando es una de las entradas (argumentos) de un operador. Por ejemplo, en 3 + 6 = 9, "+" es el operador y "*3*" y "*6*" son los operan dos.

Conclusiones.

- La realización de los diagramas de clase son, una herramienta indicada, y eficaz, para iniciar el desarrollo de cualquier tipo de software, ya que permite trabajar sobre el diseño que el software requiere, de una manera más clara y concreta.
- El análisis previo realizado para la entrega de un software, como lo es el análisis de usuario, es recomendable para tener en cuenta las exigencias que el usuario como comprador del software requiere para la implementación de este, a modo que resuelva la problemática que tiene, y la conformidad de recibirlo.

Bibliografía.

http://www.mitecnologico.com/Main/ElaboracionManualTecnico

http://telematica.ing-soft.com/post/6/

http://www.voltimum.es/search/programa+diseno+manual+tecnico.html

http://www.monografias.com/trabajos13/mapro/mapro.shtml

http://es.wikipedia.org/wiki/Polimorfismo_%28inform%C3%A1tica%29

http://office.microsoft.com/es-es/excel-help/concatenar-HP005209020.aspx

http://en.wikipedia.org/wiki/Substring

http://elvex.ugr.es/decsai/java/pdf/3C-Relaciones.pdf