

**INSTITUTO SUPERIOR TECNOLÓGICO PRIVADO CIBERTEC**

**CARRERAS PROFESIONALES**

**SISTEMA DE VENTA DE CERÁMICOS**

**INTRODUCCIÓN A LA ALGORITMIA**

**DOCENTE: ALVA ROSA RIVERA BENITO**

**PRIMER CICLO**

SECCIÓN T1GC – T1GL

SEMESTRE 2022 – 02

**Coordinador:**

Diego Alessandro Rios Villegas

**Integrantes:**

Mauricio Alfonso Menacho Apolitano

Jose David Yep Arce

Mayra Merli Cuadros Huallanca

José Alfonso Torres La Rosa

**CARRERAS PROFESIONALES**

**CIBERTEC**

**MIRAFLORES, 2022**

#### Índice

|  |  |
| --- | --- |
| [Introducción](#_Introducción) |  |
| [Justificación](#_Justificación_1) |  |
| [Objetivos](#_Objetivos_1) |  |
| [Definición](#_Definición_1) |  |
| [Entregables](#_Entregables) |  |
| [Conclusiones](#_Conclusiones) |  |
| [Recomendaciones](#_Recomendaciones) |  |
| [Bibliografía](#_Bibliografía) |  |

#### Introducción

El presente trabajo es un informe técnico de un Sistema de venta de cerámicos. Este diseño es una solución a los requerimientos de los clientes que estén interesados en consultar y comprar en los distintos tipos de cerámicos que ofrece nuestro programa. El trabajo ha sido dividido por partes, entre los 5 integrantes de este grupo, implementando y programando cada uno de los menús y secciones del programa, siendo estos (Archivo, Mantenimiento, Ventas, Configuración y Ayuda). Asimismo, se logró que cada parte tenga su correcto funcionamiento, y con las condiciones más optimas posibles siguiendo las pautas indicadas por la maestra y rúbrica del proyecto.

Aun así, el proyecto consta de ciertas limitaciones, como, por ejemplo:

**Limitaciones de tiempo:** nuestros integrantes de equipo trabajan o tienen carreras aparte de la que llevamos, es por ello que quizás tuvimos algunos inconvenientes en el proceso, a la hora de organizarnos, pero al final todo se pudo culminar ya que fuimos responsables y perseverantes.

**Limitaciones de ortografía:** Muchas veces el eclipse no aceptaba palabras que llevasen tilde y la letra con el acento era cambiaba por un código raro que al final teníamos que cambiarlo, para que así los textos fueran impresos de manera correcta.

#### Justificación

**Implementar el programa de “Ventas de Cerámicos”:**

1. Promover la integración o el uso de tecnologías a empresas y que los clientes acudan a aquella de forma predilecta por lo eficaz del despacho.
2. Aumentar la eficiencia en la atención al cliente, proveyéndole a los vendedores soluciones rápidas con las funciones de cálculo incluidas en el programa.
3. Automático registro del historial de productos vendidos, el sistema podrá generar un reporte de las ventas proveyendo de información útil (inventario) a la empresa.
4. Sistema intuitivo, por lo que facilita el trabajo a los empleados(vendedores).

**Los beneficiarios directos de este proyecto son:**

* La empresa, los programadores, jefe de inventarios, vendedores.

**Los beneficiaros indirectos de este proyecto son:**

* Clientes de la empresa.

#### Objetivos

**S:** El programa creado ayude en la venta de cerámicos.

**M:** Que cualquier persona use el programa fácilmente.

**A:** Crear un programa ayude en la venta de cerámicos, objetivamente en organizar, simplificar cálculos y registros en un lapso de 3 meses y que sea de fácil manejo.

**R:** Programado para organizar, simplificar procesos de cálculo y registro.

**T:** En 3 meses.

#### Definición

|  |  |
| --- | --- |
| 1. Al iniciar el programa aparece un mensaje de bienvenida (Diálogo flotante)   Interfaz de usuario gráfica, Aplicación, Word  Descripción generada automáticamente   1. En la ventana principal podemos observar 5 menús principales. (Archivo, Mantenimiento, Ventas, Configuración y Ayuda). Cada uno tiene los siguientes submenús:  * Archivo -> Salir * Mantenimiento -> Consultar cerámico, Modificar cerámico y Listar cerámicos. * Ventas -> Vender y Generar reportes. * Configuración -> Configurar descuentos, Configurar obsequios, Configurar cantidad óptima y Configurar cuota diaria. * Ayuda -> Acerca de Tienda.   Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico  Descripción generada automáticamente   1. La primera interfaz, **Consultar cerámico,** muestra, con la edición restringida, el precio (S/.), ancho(cm), largo(cm), espesor(mm) y contenido de cada modelo de caja de cerámicos. En este caso vemos los detalles del modelo “Cinza Plus”.   Interfaz de usuario gráfica, Tabla  Descripción generada automáticamente con confianza media   1. La segunda interfaz, **Modificar cerámico**, muestra los detalles y permite la edición de cada característica del modelo seleccionado de cerámicos. Sólo admite números. Con el botón grabar se guardan los nuevos datos ingresados en el sistema y pueden ser consultados luego.   Interfaz de usuario gráfica  Descripción generada automáticamente   1. La tercera interfaz, **Listar cerámicos**, al hacer clic en el botón Listar, muestra los detalles de cada caja de cerámicos (modelo, precio, ancho, largo, espesor y contenido).   Interfaz de usuario gráfica, Texto  Descripción generada automáticamente   1. La cuarta interfaz, **Vender,** se encuentraen el menú Ventas. Permite seleccionar un modelo y cantidad (sólo acepta números enteros) a vender. Al hacer clic al botón procesar, muestra la boleta de venta con el Modelo vendido, Precio unitario, Cantidad adquirida, Importe de compra, Importe de descuento, Importe a pagar, Tipo de obsequio y Unidades obsequiadas.   Interfaz de usuario gráfica  Descripción generada automáticamente   1. La quinta interfaz, **Generar reportes,** muestra 4 tipos de reporte: (Ventas por modelo, Comparación de precios con el precio promedio, Comparación de cajas vendidas con la cantidad óptima y Estadística sobre el precio). 2. El primer reporte, **Ventas por modelo**, muestra por cada tipo de cerámicos la cantidad de ventas, cantidad de unidades o cajas vendidas, importe total vendido y el aporte a la cuota diaria.   Interfaz de usuario gráfica, Texto  Descripción generada automáticamente   1. El segundo reporte, **Comparación de precios con el precio promedio**, muestra por cada modelo, el precio, precio promedio y la comparación entre estos.   Interfaz de usuario gráfica, Texto, Aplicación  Descripción generada automáticamente   1. El tercer reporte, **Comparación de cajas vendidas con la cantidad óptima**, muestra por cada modelo, la cantidad de cajas vendidas, la cantidad óptima y la comparación entre estas.   Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico  Descripción generada automáticamente   1. Interfaz de usuario gráfica, Texto, Aplicación     Descripción generada automáticamenteEl cuarto reporte, **Estadística sobre el precio**, hace una comparación entre los precios de cada modelo de cerámicos y determina el precio promedio, el precio mayor y el precio menor. 2. La sexta interfaz, **Configurar porcentajes de descuento,** se encuentra en el menú Configuración. Permite establecer un porcentaje de descuento por la compra de 1 a 5 unidades, 6 a 10, 11 a 15 y más de 15. No permite el ingreso de letras.   Interfaz de usuario gráfica  Descripción generada automáticamente   1. La séptima interfaz, **Configurar obsequios**, permite modificar el tipo de obsequio, y cuántas unidades se regalarán por la compra de 1 a 5 unidades, 6 a 10 y más de 11. No permite el ingreso de números en el tipo de obsequio, ni de letras en los 3 campos inferiores.   Interfaz de usuario gráfica  Descripción generada automáticamente   1. La décima interfaz, **Configurar cantidad óptima,** permite editar la cantidad óptima de unidades vendidas. Sólo acepta números enteros.   Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamente   1. La undécima interfaz, **Configurar cuota diaria,** permite editar la cuota diaria esperada. Sólo admite números y decimales.   Interfaz de usuario gráfica  Descripción generada automáticamente   1. Finalmente, la última interfaz, **Acerca de Tienda**, se encuentra dentro del menú Ayuda. En esta se presenta a los integrantes del equipo, además de la fecha y el lugar en el que fue desarrollado este programa.   Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamente |  |

#### Entregables

**Frm.Principal**  
 **VARIABLE GLOBALES Y METODO PARA IMPRIMIR y LIMITAR DECIMALES**

// Datos minimos del primer producto

**public** **static** String *modelo0* = "Cinza Plus";

**public** **static** **double** *precio0* = 92.56;

**public** **static** **double** *ancho0* = 62.0;

**public** **static** **double** *largo0* = 62.0;

**public** **static** **double** *espesor0* = 8;

**public** **static** **int** *contenido0* = 6;

// Datos minimos del segundo producto

**public** **static** String *modelo1* = "Luxury";

**public** **static** **double** *precio1* = 42.77;

**public** **static** **double** *ancho1* = 60;

**public** **static** **double** *largo1* = 60;

**public** **static** **double** *espesor1* = 8.5;

**public** **static** **int** *contenido1* = 4;

// Datos minimos del tercer producto

**public** **static** String *modelo2* = "Austria";

**public** **static** **double** *precio2* = 52.45;

**public** **static** **double** *ancho2* = 45;

**public** **static** **double** *largo2* = 45;

**public** **static** **double** *espesor2* = 6.5;

**public** **static** **int** *contenido2* = 12;

// Datos minimos del cuarto producto

**public** **static** String *modelo3* = "Yungay Mix";

**public** **static** **double** *precio3* = 55.89;

**public** **static** **double** *ancho3* = 80;

**public** **static** **double** *largo3* = 120;

**public** **static** **double** *espesor3* = 6.8;

**public** **static** **int** *contenido3* = 9;

// Datos minimos del quinto producto

**public** **static** String *modelo4* = "Thalía";

**public** **static** **double** *precio4* = 45;

**public** **static** **double** *ancho4* = 45;

**public** **static** **double** *largo4* = 11.8;

**public** **static** **double** *espesor4* = 7.2;

**public** **static** **int** *contenido4* = 10;

// Porcentajes de descuento

**public** **static** **double** *porcentaje1* = 7.5;

**public** **static** **double** *porcentaje2* = 10.0;

**public** **static** **double** *porcentaje3* = 12.5;

**public** **static** **double** *porcentaje4* = 15.0;

// Obsequio

**public** **static** String *tipoObsequio* = "Lapicero";

**public** **static** **int** *obsequioCantidad1* = 2;

**public** **static** **int** *obsequioCantidad2* = 3;

**public** **static** **int** *obsequioCantidad3* = 4;

// Cantidad optima de unidades vendidas por dï¿½a

**public** **static** **int** *cantidadOptima* = 10;

// Cuota diaria

**public** **static** **double** *cuotaDiaria* = 30000;

// Ventas por modelo

// Monto total y unidades vendidas del primer producto

**public** **static** **double** *montoTotal0* = 0;

**public** **static** **int** *cantVentas0* = 0;

**public** **static** **int** *unidades0* = 0;

// Monto total y unidades vendidas del segundo producto

**public** **static** **double** *montoTotal1* = 0;

**public** **static** **int** *cantVentas1* = 0;

**public** **static** **int** *unidades1* = 0;

// Monto total y unidades vendidas del tercer producto

**public** **static** **double** *montoTotal2* = 0;

**public** **static** **int** *cantVentas2* = 0;

**public** **static** **int** *unidades2* = 0;

// Monto total y unidades vendidas del cuarto producto

**public** **static** **double** *montoTotal3* = 0;

**public** **static** **int** *cantVentas3* = 0;

**public** **static** **int** *unidades3* = 0;

// Monto total y unidades vendidas del quinto producto

**public** **static** **double** *montoTotal4* = 0;

**public** **static** **int** *cantVentas4* = 0;

**public** **static** **int** *unidades4* = 0;

**private** JLabel lblTitulo;

**private** JLabel lblDescrip;

// Metodo - Imprimir texto en un JTextArea

**public** **static** **void** imprimir(String texto, JTextArea txtS) {

txtS.append(texto + "\n");

}

// Método - Mostrar sólo 2 decimales de un número

**public** **static** **double** redondear(**double** num) {

**return** Math.*round*(Math.*abs*(num) \* 100) / 100D;

}

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == mntmGenerarReportes) {

actionPerformedMntmGenerarReportes(e);

}

**if** (e.getSource() == mntmConfigObs) {

actionPerformedMntmConfigObs(e);

}

**if** (e.getSource() == mntmConfigDsctos) {

actionPerformedMntmConfigDsctos(e);

}

**if** (e.getSource() == mntmCantOpt) {

actionPerformedMntmCantOpt(e);

}

**if** (e.getSource() == mntmCuotaDiaria) {

actionPerformedMntmCuotaDiaria(e);

}

**if** (e.getSource() == mntmAcercaTienda) {

actionPerformedMntmAcercaTienda(e);

}

**if** (e.getSource() == mntmListarC) {

actionPerformedMntmListarC(e);

}

**if** (e.getSource() == mntmModificarC) {

actionPerformedMntmModificarC(e);

}

**if** (e.getSource() == mntmConsultarC) {

actionPerformedMntmConsultarC(e);

}

**if** (e.getSource() == mntmSalir) {

actionPerformedMntmSalir(e);

}

**if** (e.getSource() == mntmVender) {

actionPerformedMntmVenta(e);

}

}

**void** mensajeBienvenida() {

String texto, titulo;

texto = "Bienvenido al sistema de la tienda.";

titulo = "Tienda de cerámicos";

JOptionPane.*showMessageDialog*(**this**, texto, titulo, JOptionPane.***PLAIN\_MESSAGE***);

}

**protected** **void** actionPerformedMntmSalir(ActionEvent e) {

System.*exit*(0);

}

// Menu - Mantenimiento

**protected** **void** actionPerformedMntmConsultarC(ActionEvent e) {

DlgConsultarCeramico ventConsult = **new** DlgConsultarCeramico();

ventConsult.setVisible(**true**);

ventConsult.setLocationRelativeTo(**this**);

}

**protected** **void** actionPerformedMntmModificarC(ActionEvent e) {

DlgModificarCeramico ventConsult = **new** DlgModificarCeramico();

ventConsult.setVisible(**true**);

ventConsult.setLocationRelativeTo(**this**);

}

**protected** **void** actionPerformedMntmListarC(ActionEvent e) {

DlgListar ventListar = **new** DlgListar();

ventListar.setVisible(**true**);

ventListar.setLocationRelativeTo(**this**);

}

// Menu - Acerca de la Tienda

**protected** **void** actionPerformedMntmAcercaTienda(ActionEvent e) {

DlgAcercaDeTienda ventAyuda = **new** DlgAcercaDeTienda();

ventAyuda.setVisible(**true**);

ventAyuda.setLocationRelativeTo(**this**);

}

// Menu - Configuracion

**protected** **void** actionPerformedMntmCantOpt(ActionEvent e) {

DlgConfigurarCantidadOptima ventConfigCantOpt = **new** DlgConfigurarCantidadOptima();

ventConfigCantOpt.setVisible(**true**);

ventConfigCantOpt.setLocationRelativeTo(**this**);

}

**protected** **void** actionPerformedMntmCuotaDiaria(ActionEvent e) {

DlgConfigurarCuotaDiaria ventCuotaDiaria = **new** DlgConfigurarCuotaDiaria();

ventCuotaDiaria.setVisible(**true**);

ventCuotaDiaria.setLocationRelativeTo(**this**);

}

**protected** **void** actionPerformedMntmConfigDsctos(ActionEvent e) {

DlgConfigurarDescuentos ventConfigDesc = **new** DlgConfigurarDescuentos();

ventConfigDesc.setVisible(**true**);

ventConfigDesc.setLocationRelativeTo(**this**);

}

**protected** **void** actionPerformedMntmConfigObs(ActionEvent e) {

DlgConfigurarObsequios ventConfigObs = **new** DlgConfigurarObsequios();

ventConfigObs.setVisible(**true**);

ventConfigObs.setLocationRelativeTo(**this**);

}

// Menu - Ventas

**protected** **void** actionPerformedMntmVenta(ActionEvent e) {

DlgVender ventVender = **new** DlgVender();

ventVender.setVisible(**true**);

ventVender.setLocationRelativeTo(**this**);

}

**protected** **void** actionPerformedMntmGenerarReportes(ActionEvent e) {

DlgGenerarReporteVentas ventGenerarReportes = **new** DlgGenerarReporteVentas();

ventGenerarReportes.setVisible(**true**);

ventGenerarReportes.setLocationRelativeTo(**this**);

}

}

***DlgAcercaDeTienda***

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == btnCerrar) {

actionPerformedBtnCerrar(e);

}

}

**protected** **void** actionPerformedBtnCerrar(ActionEvent e) {

dispose();

}

}

***DlgConfigurarCantidadOptima***

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == btnAceptar) {

actionPerformedBtnAceptar(e);

}

**if** (e.getSource() == btnCancelar) {

actionPerformedBtnCancelar(e);

}

}

**protected** **void** actionPerformedBtnAceptar(ActionEvent e) {

**if** (*validarNumeros*(txtCantOpt.getText().trim())) {

frmPrincipal.*cantidadOptima* = Integer.*parseInt*(txtCantOpt.getText());

dispose();

} **else** {

String mensaje, titulo;

titulo = "Datos Incorrectos";

mensaje = "¡Solo se permite el ingreso de números enteros!";

JOptionPane.*showMessageDialog*(**this**, mensaje, titulo, JOptionPane.***WARNING\_MESSAGE***);

txtCantOpt.requestFocus();

txtCantOpt.setText("");

}

}

**private** **static** **boolean** validarNumeros(String datos) {

**return** datos.length() > 0 && datos.matches("[0-9]\*");

}

**protected** **void** actionPerformedBtnCancelar(ActionEvent e) {

dispose();

}

}

***DlgConfigurarCuotaDiaria***

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == btnCancelar) {

actionPerformedBtnCancelar(e);

}

**if** (e.getSource() == btnAceptar) {

actionPerformedBtnAceptar(e);

}

}

**protected** **void** actionPerformedBtnAceptar(ActionEvent e) {

**if** (*validarNumeros*(txtCuotaDiaria.getText().trim())) {

frmPrincipal.*cuotaDiaria* = Double.*parseDouble*(txtCuotaDiaria.getText());

dispose();

} **else** {

String mensaje, titulo;

titulo = "Datos Incorrectos";

mensaje = "Solo se permite el ingreso de números";

JOptionPane.*showMessageDialog*(**this**, mensaje, titulo, JOptionPane.***WARNING\_MESSAGE***);

txtCuotaDiaria.requestFocus();

txtCuotaDiaria.setText("");

}

}

**private** **static** **boolean** validarNumeros(String datos) {

**return** datos.length() > 0 && datos.matches("[0-9]\*\\.?[0-9]\*");

}

**protected** **void** actionPerformedBtnCancelar(ActionEvent e) {

dispose();

}

}

***DlgConfiguraDescuentos***

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == btnCancelar) {

actionPerformedBtnCancelar(e);

}

**if** (e.getSource() == btnAceptar) {

actionPerformedBtnAceptar(e);

}

}

**protected** **void** actionPerformedBtnAceptar(ActionEvent e) {

// Validacion de campo porcentaje

**if** (!validarNumeros(txtPorcentaje1.getText().trim()) || !validarNumeros(txtPorcentaje2.getText().trim())

|| !validarNumeros(txtPorcentaje3.getText().trim())

|| !validarNumeros(txtPorcentaje4.getText().trim())) {

mostrarAlerta("Sólo se puede ingresar números", "Datos inválidos", JOptionPane.***WARNING\_MESSAGE***);

**return**;

}

frmPrincipal.*porcentaje1* = asignarPorcentaje(txtPorcentaje1.getText());

frmPrincipal.*porcentaje2* = asignarPorcentaje(txtPorcentaje2.getText());

frmPrincipal.*porcentaje3* = asignarPorcentaje(txtPorcentaje3.getText());

frmPrincipal.*porcentaje4* = asignarPorcentaje(txtPorcentaje4.getText());

dispose();

}

**protected** **void** actionPerformedBtnCancelar(ActionEvent e) {

dispose();

}

// Metodo - Asignar porcentaje

**double** asignarPorcentaje(String porcentaje) {

**return** Double.*parseDouble*(porcentaje);

}

// Metodo - Mostrar alerta

**void** mostrarAlerta(String mensaje, String titulo, **int** icono) {

JOptionPane.*showMessageDialog*(**this**, mensaje, titulo, icono);

}

// Metodo - Validar numeros

**boolean** validarNumeros(String datos) {

**return** datos.length() > 0 && datos.matches("[0-9]\*\\.?[0-9]\*");

}

}

***DlgConfigurarObsequios***

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == btnCancelar) {

actionPerformedBtnCancelar(e);

}

**if** (e.getSource() == btnAceptar) {

actionPerformedBtnAceptar(e);

}

}

**protected** **void** actionPerformedBtnAceptar(ActionEvent e) {

frmPrincipal.*tipoObsequio* = txtTipoObsequio.getText();

frmPrincipal.*obsequioCantidad1* = Integer.*parseInt*(txtObsequioCantidad1.getText());

frmPrincipal.*obsequioCantidad2* = Integer.*parseInt*(txtObsequioCantidad2.getText());

frmPrincipal.*obsequioCantidad3* = Integer.*parseInt*(txtObsequioCantidad3.getText());

dispose();

}

**protected** **void** actionPerformedBtnCancelar(ActionEvent e) {

dispose();

}

**public** **void** keyTyped(KeyEvent e) {

**if** (e.getSource() == txtObsequioCantidad3) {

keyTypedTxtObsequioCantidad3(e);

}

**if** (e.getSource() == txtObsequioCantidad2) {

keyTypedTxtObsequioCantidad2(e);

}

**if** (e.getSource() == txtObsequioCantidad1) {

keyTypedTxtObsequioCantidad1(e);

}

**if** (e.getSource() == txtTipoObsequio) {

keyTypedTxtTipoObsequio(e);

}

}

**public** **void** keyTypedTxtTipoObsequio(KeyEvent e) {

**char** validar = e.getKeyChar();

**if** (Character.*isDigit*(validar)) {

getToolkit().beep();

e.consume();

JOptionPane.*showMessageDialog*(**this**, "Por favor, ingresar solo letras ¡Gracias!", "Datos inválidos",

JOptionPane.***WARNING\_MESSAGE***);

}

}

**private** **void** keyTypedTxtObsequioCantidad1(KeyEvent e) {

**int** key = e.getKeyChar();

**boolean** numeros = key >= 48 && key <= 57;

**if** (!numeros)

{

mostrarAlerta();

e.consume();

}

}

**private** **void** keyTypedTxtObsequioCantidad2(KeyEvent e) {

**int** key = e.getKeyChar();

**boolean** numeros = key >= 48 && key <= 57;

**if** (!numeros)

{

mostrarAlerta();

e.consume();

}

}

**protected** **void** keyTypedTxtObsequioCantidad3(KeyEvent e) {

**int** key = e.getKeyChar();

**boolean** numeros = key >= 48 && key <= 57;

**if** (!numeros)

{

mostrarAlerta();

e.consume();

}

}

**void** mostrarAlerta() {

JOptionPane.*showMessageDialog*(**this**, "Por favor, ingresar solo números ¡Gracias!", "Datos inválidos",

JOptionPane.***WARNING\_MESSAGE***);

}

**public** **void** keyPressed(KeyEvent e) {

}

**public** **void** keyReleased(KeyEvent e) {

}

}

***DlgConsultarCeramico***

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == btnCerrar) {

actionPerformedBtnCerrar(e);

}

**if** (e.getSource() == cboModelo) {

actionPerformedcboModelo(e);

}

}

**protected** **void** actionPerformedcboModelo(ActionEvent e) {

obtenerModelo();

**switch** (modelo) {

**case** 0:

mostrarCinza();

**break**;

**case** 1:

mostrarLuxury();

**break**;

**case** 2:

mostrarAustria();

**break**;

**case** 3:

mostrarYungay();

**break**;

**default**:

mostrarThalia();

**break**;

}

}

// declaracion de variables

**int** modelo;

// Entrada de datos

**void** obtenerModelo() {

modelo = cboModelo.getSelectedIndex();

}

// mostrar datos de acuerdo al modelo

**void** mostrarCinza() {

txtPrecio.setText("" + frmPrincipal.*precio0*);

txtAncho.setText("" + frmPrincipal.*ancho0*);

txtLargo.setText("" + frmPrincipal.*largo0*);

txtEspesor.setText("" + frmPrincipal.*espesor0*);

txtContenido.setText("" + frmPrincipal.*contenido0*);

}

**void** mostrarLuxury() {

txtPrecio.setText("" + frmPrincipal.*precio1*);

txtAncho.setText("" + frmPrincipal.*ancho1*);

txtLargo.setText("" + frmPrincipal.*largo1*);

txtEspesor.setText("" + frmPrincipal.*espesor1*);

txtContenido.setText("" + frmPrincipal.*contenido1*);

}

**void** mostrarAustria() {

txtPrecio.setText("" + frmPrincipal.*precio2*);

txtAncho.setText("" + frmPrincipal.*ancho2*);

txtLargo.setText("" + frmPrincipal.*largo2*);

txtEspesor.setText("" + frmPrincipal.*espesor2*);

txtContenido.setText("" + frmPrincipal.*contenido2*);

}

**void** mostrarYungay() {

txtPrecio.setText("" + frmPrincipal.*precio3*);

txtAncho.setText("" + frmPrincipal.*ancho3*);

txtLargo.setText("" + frmPrincipal.*largo3*);

txtEspesor.setText("" + frmPrincipal.*espesor3*);

txtContenido.setText("" + frmPrincipal.*contenido3*);

}

**void** mostrarThalia() {

txtPrecio.setText("" + frmPrincipal.*precio4*);

txtAncho.setText("" + frmPrincipal.*ancho4*);

txtLargo.setText("" + frmPrincipal.*largo4*);

txtEspesor.setText("" + frmPrincipal.*espesor4*);

txtContenido.setText("" + frmPrincipal.*contenido4*);

}

**protected** **void** actionPerformedBtnCerrar(ActionEvent e) {

dispose();

}

}

***DlgGenerarReporteVentas***

**public** DlgGenerarReporteVentas() {

setTitle("Generar reportes");

setBounds(100, 100, 666, 422);

getContentPane().setLayout(**null**);

lblTipoReporte = **new** JLabel("Tipo de reporte");

lblTipoReporte.setFont(**new** Font("Tahoma", Font.***PLAIN***, 14));

lblTipoReporte.setBounds(22, 20, 103, 22);

getContentPane().add(lblTipoReporte);

cboTipoReporte = **new** JComboBox<String>();

cboTipoReporte.setFont(**new** Font("Tahoma", Font.***PLAIN***, 14));

cboTipoReporte.setBackground(**new** Color(255, 255, 204));

cboTipoReporte.addActionListener(**this**);

cboTipoReporte.setModel(**new** DefaultComboBoxModel<String>(

**new** String[] { "Ventas por modelo", "Comparación de precios con el precio promedio",

"Comparación de cajas vendidas con la cantidad óptima", "Estadística sobre el precio" }));

cboTipoReporte.setBounds(132, 20, 381, 22);

getContentPane().add(cboTipoReporte);

btnCerrar = **new** JButton("Cerrar");

btnCerrar.setFont(**new** Font("Tahoma", Font.***PLAIN***, 14));

btnCerrar.setBackground(**new** Color(255, 153, 153));

btnCerrar.addActionListener(**this**);

btnCerrar.setBounds(541, 20, 89, 23);

getContentPane().add(btnCerrar);

scp = **new** JScrollPane();

scp.setBounds(20, 56, 610, 269);

getContentPane().add(scp);

txtS = **new** JTextArea();

txtS.setEditable(**false**);

txtS.setFont(**new** Font("Monospaced", Font.***PLAIN***, 16));

scp.setViewportView(txtS);

generarReporteModelo();

}

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == btnCerrar) {

actionPerformedBtnCerrar(e);

}

**if** (e.getSource() == cboTipoReporte) {

actionPerformedCboTipoReporte(e);

}

}

**protected** **void** actionPerformedBtnCerrar(ActionEvent e) {

dispose();

}

**protected** **void** actionPerformedCboTipoReporte(ActionEvent e) {

**int** tipoReporte;

tipoReporte = cboTipoReporte.getSelectedIndex();

// Mostrar reportes

**switch** (tipoReporte) {

**case** 0:

generarReporteModelo();

**break**;

**case** 1:

mostrarComparacionPromedio();

**break**;

**case** 2:

mostrarComparacionCantOpt();

**break**;

**default**:

mostrarEstPrecio();

**break**;

}

}

// Metodo - Calcular y mostrar reporte de ventas por modelo

**void** generarReporteModelo() {

// Mostrando ventas por modelo

txtS.setText("VENTAS POR MODELO \n\n");

// Mostrando reporte del primer modelo

imprimirReporte(frmPrincipal.*modelo0*, frmPrincipal.*cantVentas0*, frmPrincipal.*unidades0*,

frmPrincipal.*montoTotal0*);

// Mostrando reporte del segundo modelo

imprimirReporte(frmPrincipal.*modelo1*, frmPrincipal.*cantVentas1*, frmPrincipal.*unidades1*,

frmPrincipal.*montoTotal1*);

// Mostrando reporte del tercer modelo

imprimirReporte(frmPrincipal.*modelo2*, frmPrincipal.*cantVentas2*, frmPrincipal.*unidades2*,

frmPrincipal.*montoTotal2*);

// Mostrando reporte del cuarto modelo

imprimirReporte(frmPrincipal.*modelo3*, frmPrincipal.*cantVentas3*, frmPrincipal.*unidades3*,

frmPrincipal.*montoTotal3*);

// Mostrando reporte del quinto modelo

imprimirReporte(frmPrincipal.*modelo4*, frmPrincipal.*cantVentas4*, frmPrincipal.*unidades4*,

frmPrincipal.*montoTotal4*);

}

// Metodo - Imprimir reporte de ventas por modelo

**void** imprimirReporte(String modelo, **int** cantVentas, **int** unidades, **double** montoTotal) {

**double** aporteCuota = montoTotal \* 100 / frmPrincipal.*cuotaDiaria*;

frmPrincipal.*imprimir*("Modelo \t\t\t\t: " + modelo, txtS);

frmPrincipal.*imprimir*("Cantidad de ventas \t\t: " + cantVentas, txtS);

frmPrincipal.*imprimir*("Cantidad de cajas vendidas \t: " + unidades, txtS);

frmPrincipal.*imprimir*("Importe total vendido \t\t: S/." + frmPrincipal.*redondear*(montoTotal), txtS);

frmPrincipal.*imprimir*("Aporte a la cuota diaria \t: " + frmPrincipal.*redondear*(aporteCuota) + "%\n", txtS);

}

// Metodo - Calcular y mostrar comparación de precios con el precio promedio

**void** mostrarComparacionPromedio() {

// Calculo del precio promedio

**double** precioProm = (frmPrincipal.*precio0* + frmPrincipal.*precio1* + frmPrincipal.*precio2* + frmPrincipal.*precio3*

+ frmPrincipal.*precio4*) / 5;

// Salida de resultados

txtS.setText("COMPARACIÓN DE PRECIOS CON EL PRECIO PROMEDIO \n\n");

imprimirPrecioPromedio(frmPrincipal.*modelo0*, frmPrincipal.*precio0*, precioProm);

imprimirPrecioPromedio(frmPrincipal.*modelo1*, frmPrincipal.*precio1*, precioProm);

imprimirPrecioPromedio(frmPrincipal.*modelo2*, frmPrincipal.*precio2*, precioProm);

imprimirPrecioPromedio(frmPrincipal.*modelo3*, frmPrincipal.*precio3*, precioProm);

imprimirPrecioPromedio(frmPrincipal.*modelo4*, frmPrincipal.*precio4*, precioProm);

}

// Metodo - Imprimir comparación precio promedio

**void** imprimirPrecioPromedio(String modelo, **double** precio, **double** precioProm) {

String mayorOMenor = " más que el promedio";

**double** diferenciaPre = (precio - precioProm);

**if** (diferenciaPre < 0)

mayorOMenor = " menos que el promedio";

frmPrincipal.*imprimir*("Modelo \t\t: " + modelo, txtS);

frmPrincipal.*imprimir*("Precio \t\t: S/." + precio, txtS);

frmPrincipal.*imprimir*("Precio promedio : S/." + precioProm, txtS);

frmPrincipal.*imprimir*("Comparación \t: S/." + frmPrincipal.*redondear*(diferenciaPre) + mayorOMenor + "\n", txtS);

}

// Metodo - Calcular y mostrar comparación de cajas vendidas con la cantidad

// óptima

**void** mostrarComparacionCantOpt() {

// Lectura de datos

**int** cantOpt = frmPrincipal.*cantidadOptima*;

// Salida de resultados

txtS.setText("COMPARACIÓN DE DE CAJAS VENDIDAS CON LA CANTIDAD ÓPTIMA \n\n");

imprimirCantOpt(frmPrincipal.*modelo0*, frmPrincipal.*unidades0*, cantOpt);

imprimirCantOpt(frmPrincipal.*modelo1*, frmPrincipal.*unidades1*, cantOpt);

imprimirCantOpt(frmPrincipal.*modelo2*, frmPrincipal.*unidades2*, cantOpt);

imprimirCantOpt(frmPrincipal.*modelo3*, frmPrincipal.*unidades3*, cantOpt);

imprimirCantOpt(frmPrincipal.*modelo4*, frmPrincipal.*unidades4*, cantOpt);

}

// Metodo - Imprimir comparación cajas vendidas con la cantidad óptima

**void** imprimirCantOpt(String modelo, **int** cajasVendidas, **int** cantOpt) {

String comparacion = " más que la cantidad óptima";

**int** diferenciaCant = (cajasVendidas - cantOpt);

**if** (diferenciaCant < 0)

comparacion = " menos que la cantidad óptima";

**else** **if** (diferenciaCant == cantOpt) {

comparacion = " igual que la cantidad óptima";

}

frmPrincipal.*imprimir*("Modelo \t\t\t : " + modelo, txtS);

frmPrincipal.*imprimir*("Cantidad de cajas vendidas: " + cajasVendidas, txtS);

frmPrincipal.*imprimir*("Cantidad óptima \t : " + cantOpt, txtS);

frmPrincipal.*imprimir*("Comparación \t\t : " + Math.*abs*(diferenciaCant) + comparacion + "\n", txtS);

}

// Metodo - Calcular y mostrar estadistica sobre el precio

**void** mostrarEstPrecio() {

**double** precioMayor, precioProm, precioMenor;

// Calculando precio promedio, precio mayor y precio menor

precioProm = (frmPrincipal.*precio0* + frmPrincipal.*precio1* + frmPrincipal.*precio2* + frmPrincipal.*precio3*

+ frmPrincipal.*precio4*) / 5;

precioMayor = Math.*max*(frmPrincipal.*precio0*, Math.*max*(frmPrincipal.*precio1*,

Math.*max*(frmPrincipal.*precio2*, Math.*max*(frmPrincipal.*precio3*, frmPrincipal.*precio4*))));

precioMenor = Math.*min*(frmPrincipal.*precio0*, Math.*min*(frmPrincipal.*precio1*,

Math.*min*(frmPrincipal.*precio2*, Math.*min*(frmPrincipal.*precio3*, frmPrincipal.*precio4*))));

// Salida de resultados

txtS.setText("ESTADÍSTICA SOBRE EL PRECIO \n\n");

frmPrincipal.*imprimir*("Precio promedio : S/." + frmPrincipal.*redondear*(precioProm), txtS);

frmPrincipal.*imprimir*("Precio mayor : S/." + frmPrincipal.*redondear*(precioMayor), txtS);

frmPrincipal.*imprimir*("Precio menor : S/." + frmPrincipal.*redondear*(precioMenor), txtS);

}

}

***DlgListar***

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == btnListar) {

actionPerformedBtnListar(e);

}

**if** (e.getSource() == btnCerrar) {

actionPerformedBtnCerrar(e);

}

}

**protected** **void** actionPerformedBtnCerrar(ActionEvent e) {

dispose();

}

**protected** **void** actionPerformedBtnListar(ActionEvent e) {

txtS.setText("Listado de Ceramicos \n\n");

frmPrincipal.*imprimir*("Modelo : " + frmPrincipal.*modelo0*, txtS);

frmPrincipal.*imprimir*("Precio : " + frmPrincipal.*precio0*, txtS);

frmPrincipal.*imprimir*("Ancho (cm) : " + frmPrincipal.*ancho0* + " cm", txtS);

frmPrincipal.*imprimir*("Largo (cm) : " + frmPrincipal.*largo0* + " cm", txtS);

frmPrincipal.*imprimir*("Espesor (mm) : " + frmPrincipal.*espesor0* + " mm", txtS);

frmPrincipal.*imprimir*("Contenido : " + frmPrincipal.*contenido0* + " unidades" + "\n", txtS);

frmPrincipal.*imprimir*("Modelo : " + frmPrincipal.*modelo1*, txtS);

frmPrincipal.*imprimir*("Precio : " + frmPrincipal.*precio1*, txtS);

frmPrincipal.*imprimir*("Ancho (cm) : " + frmPrincipal.*ancho1* + " cm", txtS);

frmPrincipal.*imprimir*("Largo (cm) : " + frmPrincipal.*largo1* + " cm", txtS);

frmPrincipal.*imprimir*("Espesor (mm) : " + frmPrincipal.*espesor1* + " mm", txtS);

frmPrincipal.*imprimir*("Contenido : " + frmPrincipal.*contenido1* + " unidades" + "\n", txtS);

frmPrincipal.*imprimir*("Modelo : " + frmPrincipal.*modelo2*, txtS);

frmPrincipal.*imprimir*("Precio : " + frmPrincipal.*precio2*, txtS);

frmPrincipal.*imprimir*("Ancho (cm) : " + frmPrincipal.*ancho2* + " cm", txtS);

frmPrincipal.*imprimir*("Largo (cm) : " + frmPrincipal.*largo2* + " cm", txtS);

frmPrincipal.*imprimir*("Espesor (mm) : " + frmPrincipal.*espesor2* + " mm", txtS);

frmPrincipal.*imprimir*("Contenido : " + frmPrincipal.*contenido2* + " unidades" + "\n", txtS);

frmPrincipal.*imprimir*("Modelo : " + frmPrincipal.*modelo3*, txtS);

frmPrincipal.*imprimir*("Precio : " + frmPrincipal.*precio3*, txtS);

frmPrincipal.*imprimir*("Ancho (cm) : " + frmPrincipal.*ancho3* + " cm", txtS);

frmPrincipal.*imprimir*("Largo (cm) : " + frmPrincipal.*largo3* + " cm", txtS);

frmPrincipal.*imprimir*("Espesor (mm) : " + frmPrincipal.*espesor3* + " mm", txtS);

frmPrincipal.*imprimir*("Contenido : " + frmPrincipal.*contenido3* + " unidades" + "\n", txtS);

frmPrincipal.*imprimir*("Modelo : " + frmPrincipal.*modelo4*, txtS);

frmPrincipal.*imprimir*("Precio : " + frmPrincipal.*precio4*, txtS);

frmPrincipal.*imprimir*("Ancho (cm) : " + frmPrincipal.*ancho4* + " cm", txtS);

frmPrincipal.*imprimir*("Largo (cm) : " + frmPrincipal.*largo4* + " cm", txtS);

frmPrincipal.*imprimir*("Espesor (mm) : " + frmPrincipal.*espesor4* + " mm", txtS);

frmPrincipal.*imprimir*("Contenido : " + frmPrincipal.*contenido4* + " unidades", txtS);

}

}

***DlgModificarCeramico***

txtPrecio.setText("" + frmPrincipal.*precio0*);

txtAncho.setText("" + frmPrincipal.*ancho0*);

txtLargo.setText("" + frmPrincipal.*largo0*);

txtEspesor.setText("" + frmPrincipal.*espesor0*);

txtContenido.setText("" + frmPrincipal.*contenido0*);

}

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == btnGrabar) {

actionPerformedBtnGrabar(e);

}

**if** (e.getSource() == btnCerrar) {

actionPerformedBtnCerrar(e);

}

**if** (e.getSource() == cboModelo) {

actionPerformedcboModelo(e);

}

}

**protected** **void** actionPerformedcboModelo(ActionEvent e) {

obtenerModelo();

**switch** (modelo) {

**case** 0:

mostrarCinza();

**break**;

**case** 1:

mostrarLuxury();

**break**;

**case** 2:

mostrarAustria();

**break**;

**case** 3:

mostrarYungay();

**break**;

**default**:

mostrarThalia();

**break**;

}

}

// declaracion de variables

**int** modelo;

// Entrada de datos

**void** obtenerModelo() {

modelo = cboModelo.getSelectedIndex();

}

// mostrar datos de acuerdo al modelo

**void** mostrarCinza() {

txtPrecio.setText("" + frmPrincipal.*precio0*);

txtAncho.setText("" + frmPrincipal.*ancho0*);

txtLargo.setText("" + frmPrincipal.*largo0*);

txtEspesor.setText("" + frmPrincipal.*espesor0*);

txtContenido.setText("" + frmPrincipal.*contenido0*);

}

**void** mostrarLuxury() {

txtPrecio.setText("" + frmPrincipal.*precio1*);

txtAncho.setText("" + frmPrincipal.*ancho1*);

txtLargo.setText("" + frmPrincipal.*largo1*);

txtEspesor.setText("" + frmPrincipal.*espesor1*);

txtContenido.setText("" + frmPrincipal.*contenido1*);

}

**void** mostrarAustria() {

txtPrecio.setText("" + frmPrincipal.*precio2*);

txtAncho.setText("" + frmPrincipal.*ancho2*);

txtLargo.setText("" + frmPrincipal.*largo2*);

txtEspesor.setText("" + frmPrincipal.*espesor2*);

txtContenido.setText("" + frmPrincipal.*contenido2*);

}

**void** mostrarYungay() {

txtPrecio.setText("" + frmPrincipal.*precio3*);

txtAncho.setText("" + frmPrincipal.*ancho3*);

txtLargo.setText("" + frmPrincipal.*largo3*);

txtEspesor.setText("" + frmPrincipal.*espesor3*);

txtContenido.setText("" + frmPrincipal.*contenido3*);

}

**void** mostrarThalia() {

txtPrecio.setText("" + frmPrincipal.*precio4*);

txtAncho.setText("" + frmPrincipal.*ancho4*);

txtLargo.setText("" + frmPrincipal.*largo4*);

txtEspesor.setText("" + frmPrincipal.*espesor4*);

txtContenido.setText("" + frmPrincipal.*contenido4*);

}

**protected** **void** actionPerformedBtnCerrar(ActionEvent e) {

dispose();

}

**protected** **void** actionPerformedBtnGrabar(ActionEvent e) {

**int** modelo;

modelo = cboModelo.getSelectedIndex();

**switch** (modelo) {

**case** 0:

frmPrincipal.*precio0* = Double.*parseDouble*(txtPrecio.getText());

frmPrincipal.*ancho0* = Double.*parseDouble*(txtAncho.getText());

frmPrincipal.*largo0* = Double.*parseDouble*(txtLargo.getText());

frmPrincipal.*espesor0* = Double.*parseDouble*(txtEspesor.getText());

frmPrincipal.*contenido0* = Integer.*parseInt*(txtContenido.getText());

**break**;

**case** 1:

frmPrincipal.*precio1* = Double.*parseDouble*(txtPrecio.getText());

frmPrincipal.*ancho1* = Double.*parseDouble*(txtAncho.getText());

frmPrincipal.*largo1* = Double.*parseDouble*(txtLargo.getText());

frmPrincipal.*espesor1* = Double.*parseDouble*(txtEspesor.getText());

frmPrincipal.*contenido1* = Integer.*parseInt*(txtContenido.getText());

**break**;

**case** 2:

frmPrincipal.*precio2* = Double.*parseDouble*(txtPrecio.getText());

frmPrincipal.*ancho2* = Double.*parseDouble*(txtAncho.getText());

frmPrincipal.*largo2* = Double.*parseDouble*(txtLargo.getText());

frmPrincipal.*espesor2* = Double.*parseDouble*(txtEspesor.getText());

frmPrincipal.*contenido2* = Integer.*parseInt*(txtContenido.getText());

**break**;

**case** 3:

frmPrincipal.*precio3* = Double.*parseDouble*(txtPrecio.getText());

frmPrincipal.*ancho3* = Double.*parseDouble*(txtAncho.getText());

frmPrincipal.*largo3* = Double.*parseDouble*(txtLargo.getText());

frmPrincipal.*espesor3* = Double.*parseDouble*(txtEspesor.getText());

frmPrincipal.*contenido3* = Integer.*parseInt*(txtContenido.getText());

**break**;

**default**:

frmPrincipal.*precio4* = Double.*parseDouble*(txtPrecio.getText());

frmPrincipal.*ancho4* = Double.*parseDouble*(txtAncho.getText());

frmPrincipal.*largo4* = Double.*parseDouble*(txtLargo.getText());

frmPrincipal.*espesor4* = Double.*parseDouble*(txtEspesor.getText());

frmPrincipal.*contenido4* = Integer.*parseInt*(txtContenido.getText());

**break**;

}

dispose();

}

**public** **void** keyPressed(KeyEvent e) {

}

**public** **void** keyReleased(KeyEvent e) {

}

**public** **void** keyTyped(KeyEvent e) {

**if** (e.getSource() == txtContenido) {

keyTypedTxtContenido(e);

}

**if** (e.getSource() == txtEspesor) {

keyTypedTxtEspesor(e);

}

**if** (e.getSource() == txtLargo) {

keyTypedTxtLargo(e);

}

**if** (e.getSource() == txtAncho) {

keyTypedTxtAncho(e);

}

**if** (e.getSource() == txtPrecio) {

keyTypedTxtPrecio(e);

}

}

**void** mostrarAlerta() {

JOptionPane.*showMessageDialog*(**this**, "Por favor, ingresar solo numeros ¡Gracias!", "Datos inválidos",

JOptionPane.***WARNING\_MESSAGE***);

}

**protected** **void** keyTypedTxtPrecio(java.awt.event.KeyEvent evt) {

**char** caracter = evt.getKeyChar();

**if** (((caracter < '0' || caracter > '9')) && (caracter != KeyEvent.***VK\_BACK\_SPACE***)

&& (caracter != '.' || txtPrecio.getText().contains("."))) {

mostrarAlerta();

evt.consume();

}

}

**protected** **void** keyTypedTxtAncho(java.awt.event.KeyEvent evt) {

**char** caracter = evt.getKeyChar();

**if** (((caracter < '0' || caracter > '9')) && (caracter != KeyEvent.***VK\_BACK\_SPACE***)

&& (caracter != '.' || txtAncho.getText().contains("."))) {

mostrarAlerta();

evt.consume();

}

}

**protected** **void** keyTypedTxtLargo(java.awt.event.KeyEvent evt) {

**char** caracter = evt.getKeyChar();

**if** (((caracter < '0' || caracter > '9')) && (caracter != KeyEvent.***VK\_BACK\_SPACE***)

&& (caracter != '.' || txtLargo.getText().contains("."))) {

mostrarAlerta();

evt.consume();

}

}

**protected** **void** keyTypedTxtEspesor(java.awt.event.KeyEvent evt) {

**char** caracter = evt.getKeyChar();

**if** (((caracter < '0' || caracter > '9')) && (caracter != KeyEvent.***VK\_BACK\_SPACE***)

&& (caracter != '.' || txtEspesor.getText().contains("."))) {

mostrarAlerta();

evt.consume();

}

}

**protected** **void** keyTypedTxtContenido(java.awt.event.KeyEvent evt) {

**int** key = evt.getKeyChar();

**boolean** numeros = key > 48 && key <= 57;

**if** (!numeros)

{

mostrarAlerta();

evt.consume();

}

}

}

***DlgVender***

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == btnBorrar) {

actionPerformedBtnBorrar(e);

}

**if** (e.getSource() == btnProcesar) {

actionPerformedBtnProcesar(e);

}

**if** (e.getSource() == cboModelo) {

actionPerformedcbo(e);

}

}

**protected** **void** actionPerformedcbo(ActionEvent e) {

**int** modelo;

modelo = cboModelo.getSelectedIndex();

**switch** (modelo) {

**case** 0:

txtPrecio.setText("" + frmPrincipal.*precio0*);

**break**;

**case** 1:

txtPrecio.setText("" + frmPrincipal.*precio1*);

**break**;

**case** 2:

txtPrecio.setText("" + frmPrincipal.*precio2*);

**break**;

**case** 3:

txtPrecio.setText("" + frmPrincipal.*precio3*);

**break**;

**default**:

txtPrecio.setText("" + frmPrincipal.*precio4*);

**break**;

}

}

**protected** **void** actionPerformedBtnProcesar(ActionEvent e) {

contadorCant++;

// Declaracion de variables

**int** cantidad = 0, canobsequio;

**double** impD, impC, impP, precio;

String modelo, tipo;

// Validacion de campo cantidad

**if** (validarNumeros(txtCantidad.getText().trim())) {

cantidad = Integer.*parseInt*(txtCantidad.getText());

} **else** {

JOptionPane.*showMessageDialog*(**this**, "¡Solo se puede ingresar números!", "Datos inválidos",

JOptionPane.***WARNING\_MESSAGE***);

actionPerformedBtnBorrar(e);

**return**;

}

// Lectura de datos

modelo = "" + cboModelo.getSelectedItem();

precio = Double.*parseDouble*(txtPrecio.getText());

tipo = frmPrincipal.*tipoObsequio*;

// Calculando el importe de compra

impC = calcularImporteCompra(cantidad);

// Calculando el importe de descuento

impD = calcularImporteDescuento(impC, cantidad);

// Calculando el importe a pagar

impP = calcularImporteAPagar(impC, impD);

// Asignando el total de importes

totalImportes += impP;

// Calculando el obsequio

canobsequio = calcularObsequio(cantidad);

// Salida de resultados

mostrarResultados(modelo, precio, cantidad, impC, impD, impP, tipo, canobsequio);

// Mensaje de alerta

mostrarAlerta();

// Sumando al total de ventas por modelo

registrarVentasModelo(impP, cantidad);

}

**protected** **void** actionPerformedBtnBorrar(ActionEvent e) {

txtCantidad.setText("");

txtS.setText("");

cboModelo.setSelectedIndex(0);

txtCantidad.requestFocus();

}

// Método - Importe de compra

**double** calcularImporteCompra(**int** cantidad) {

**switch** (cboModelo.getSelectedIndex()) {

**case** 0:

**return** cantidad \* frmPrincipal.*precio0*;

**case** 1:

**return** cantidad \* frmPrincipal.*precio1*;

**case** 2:

**return** cantidad \* frmPrincipal.*precio2*;

**case** 3:

**return** cantidad \* frmPrincipal.*precio3*;

**default**:

**return** cantidad \* frmPrincipal.*precio4*;

}

}

// Método - Importe de descuento

**double** calcularImporteDescuento(**double** impC, **int** cantidad) {

**if** (cantidad <= 5)

**return** impC / 100 \* frmPrincipal.*porcentaje1*;

**else** **if** (cantidad <= 10)

**return** impC / 100 \* frmPrincipal.*porcentaje2*;

**else** **if** (cantidad <= 15)

**return** impC / 100 \* frmPrincipal.*porcentaje3*;

**else**

**return** impC / 100 \* frmPrincipal.*porcentaje4*;

}

// Metodo - Importe a pagar

**double** calcularImporteAPagar(**double** impC, **double** impD) {

**return** impC - impD;

}

// Metodo - Obsequio

**int** calcularObsequio(**int** cantidad) {

**if** (cantidad <= 5)

**return** cantidad \* frmPrincipal.*obsequioCantidad1*;

**else** **if** (cantidad <= 10)

**return** cantidad \* frmPrincipal.*obsequioCantidad2*;

**else**

**return** cantidad \* frmPrincipal.*obsequioCantidad3*;

}

// Metodo - Mostrar boleta de venta

**void** mostrarResultados(String modelo, **double** precio, **int** cantidad, **double** impC, **double** impD, **double** impP,

String tipo, **int** canobsequio) {

txtS.setText("Boleta de Venta \n\n");

frmPrincipal.*imprimir*("Modelo\t\t\t: " + modelo, txtS);

frmPrincipal.*imprimir*("Precio\t\t\t: S/." + precio, txtS);

frmPrincipal.*imprimir*("Cantidad Adquirida\t: " + cantidad, txtS);

frmPrincipal.*imprimir*("Importe de Compra\t: S/." + frmPrincipal.*redondear*(impC), txtS);

frmPrincipal.*imprimir*("Importe de Descuento\t: S/." + frmPrincipal.*redondear*(impD), txtS);

frmPrincipal.*imprimir*("Importe a Pagar\t\t: S/." + frmPrincipal.*redondear*(impP), txtS);

frmPrincipal.*imprimir*("Tipo de obsequio\t: " + tipo, txtS);

frmPrincipal.*imprimir*("Unidades obsequiadas\t: " + canobsequio, txtS);

}

// Metodo - Mostrar alerta

**void** mostrarAlerta() {

**if** (contadorCant % 5 == 0) {

String mensaje, titulo;

**double** porcentaje;

porcentaje = totalImportes \* 100 / frmPrincipal.*cuotaDiaria*;

mensaje = "Venta Nro. " + contadorCant + "\n" + "Importe total general acumulado: S/. "

+ frmPrincipal.*redondear*(totalImportes) + "\n" + "Porcentaje de la cuota diaria: "

+ frmPrincipal.*redondear*(porcentaje) + "%";

titulo = "Avance de ventas";

JOptionPane.*showMessageDialog*(**this**, mensaje, titulo, JOptionPane.***INFORMATION\_MESSAGE***);

}

}

// Metodo - Registrando ventas

**void** registrarVentasModelo(**double** impP, **int** cantidad) {

**switch** (cboModelo.getSelectedIndex()) {

**case** 0:

frmPrincipal.*montoTotal0* += impP;

frmPrincipal.*cantVentas0*++;

frmPrincipal.*unidades0* += cantidad;

**break**;

**case** 1:

frmPrincipal.*montoTotal1* += impP;

frmPrincipal.*cantVentas1*++;

frmPrincipal.*unidades1* += cantidad;

**break**;

**case** 2:

frmPrincipal.*montoTotal2* += impP;

frmPrincipal.*cantVentas2*++;

frmPrincipal.*unidades2* += cantidad;

**break**;

**case** 3:

frmPrincipal.*montoTotal3* += impP;

frmPrincipal.*cantVentas3*++;

frmPrincipal.*unidades3* += cantidad;

**break**;

**default**:

frmPrincipal.*montoTotal4* += impP;

frmPrincipal.*cantVentas4*++;

frmPrincipal.*unidades4* += cantidad;

**break**;

}

}

// Método - Validar numeros

**boolean** validarNumeros(String datos) {

**return** datos.length() > 0 && datos.matches("[0-9]\*");

}

}

#### Conclusiones

Al finalizar este proyecto, hemos comprobado que el proceso de creación de un programa de sistema de ventas es complejo, ya que, depende de varios factores, se tiene que seguir una estructura de pasos para que el proyecto marche de forma correcta y con los menores inconvenientes posibles.

* Primero se debe tener una idea del proyecto a trabajar (en este caso Sistema de ventas de Cerámicos).
* Es necesario primero crear la GUI de la página principal donde irían ubicados cada uno de los menús y sus distintas secciones aparte de colocar las distintas variables globales.
* Seguido de ello se crea los JFrame’s con sus distingas GUI’s a cada sección de los menús y se procede a añadir un controlador de evento para que al presionar en los submenús nos habrá las ventanas a configurar.

Teniendo los conocimientos previos adquiridos en las clases de Introducción a la algoritmia y otras fuentes, se procede a empezar a trabajar con el código Java respectivamente. De ser posible el proyecto pasaría por una retroalimentación por parte de un especialista; en este caso la maestra, ya que podremos tomar en cuenta las recomendaciones y así poder mejorar el proyecto de la mejor manera posible, claro está.

Siguiendo estas pautas, el objetivo se pudo cumplir a partir con mucho esfuerzo, responsabilidad y dedicación. Este proyecto incluso consiguió ampliar las perspectivas de cada integrante de nuestro grupo, logrando así desarrollar, al final de este trabajo de 1er ciclo. Un programa de un sistema de ventas de cerámicos con la capacidad de poder ser utilizada por empresas, tiendas u otros, para así poder dar a conocer su variedad de cerámicos que ofrecen estas, en los distintos menús y secciones en las que se encuentren.

#### Recomendaciones

1. Recomendamos que el equipo realice reuniones frecuentes para lograr una mejor organización y distribución de trabajo en el proyecto.
2. Recomendamos utilizar Git y GitHub para colaborar en tiempo real y evitar enviar por mensajes o correos código extenso. Estos softwares facilitan el trabajo en equipo, ahorrando tiempo y brindando una mejor organización. Además, GIT al ser un software de control de versiones brinda seguridad al proyecto y evita que se pierdan partes importantes del código. Pueden aprender de Git y GitHub aquí: https://www.freecodecamp.org/espanol/news/guia-para-principiantes-de-git-y-github/

#### Bibliografía

* Método utilizado para calcular el precio mayor en el programa; <https://www.geeksforgeeks.org/java-math-max-method-examples/>
* Método utilizado para calcular el precio menor en el programa <https://www.geeksforgeeks.org/java-math-min-method-examples/>