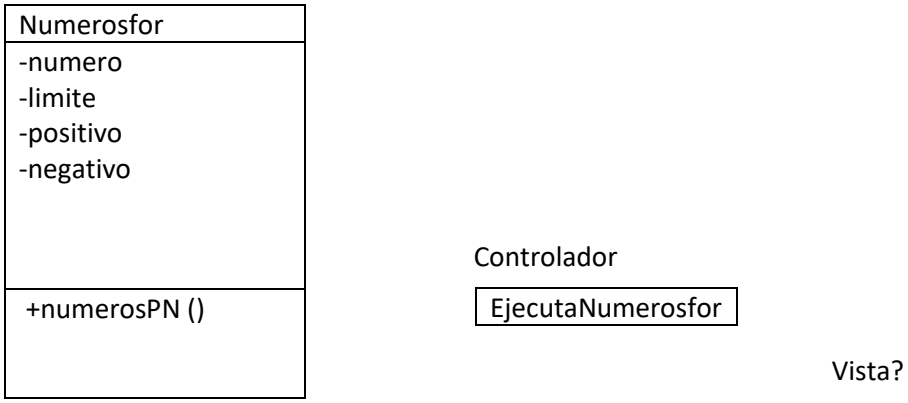
1. DIAGRAMA DE CLASES

   Modelo



2. Analicis del Problema

   Detectar objetos -> numero
   Limite

   Procesos ->

```
if (limite == 1) {

if (numero > positivo)
   positivo = numero;



if (numero < negativo)
   negativo = numero;
```

   Salidas->

   Presentar: positivo
   Negativo

3. Diseño del Programa

   1.Declarar Datos

```
Int numero=0;
Int limite =0;
Int positivo =0;
Int negativo =0;
```

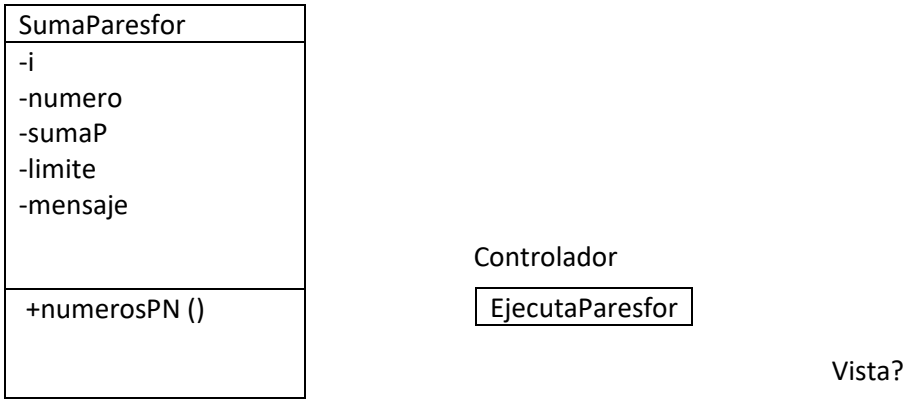
   2.Metodo numerosPN()

```
negativo=numero;

if (limite == 1)

   if (numero > positivo) {
```

```
            positivo = numero;
                    if (numero < negativo)
                      negativo = numero;

            end if
 Fin Metodo numerosPN
Fin clase  Numerosfor
  Clase EjecutaNumerosfor
1.Metodo Principal()
Declarar ,crear e inciar objeto
Numerosfor numero = new Numerosfor();
Solicitar el ingreso de limite de números
Leer limite
For(numero.limite[i], i<= limite ,i++){
Solicitar el ingreso de el número
Leer numero[]
Establecer  números.numerosPn()
Imprimir  números.Positivos()
             Números.Negativos
Fin Metodo principal
Fin Clase EjecutaNumerosfor
```

1. DIAGRAMA DE CLASES

   Modelo



2. Analicis del Problema

   Detectar objetos -> numero
   Limite

   Procesos -> for (i=0;i <limite;i++) {

   if (numero % 2 == 0)
   mensaje=("La suma se realizo corectamente");
   sumaP = sumaP+ numero;

   else

   mensaje=("___|Error|___");

   Salidas->

   Presentar: mensaje
   sumaP

3. Diseño del Programa

   1.Declarar Datos

   Int i=0;
   Int numero=0;
   Int sumaP =0;
   Int limite=0;
   String mensaje;

   2.Metodo sumaP()

```
for (i=0;i <limite;i++) {

    if (numero % 2 == 0) {

      mensaje=("La suma se realizo corectamente");
```

sumaP = sumaP+ numero;

else

mensaje=("___|Error|___");

Fin Metodo SumaP

Fin clase SumaParesfor

Clase EjecutaParesfor

1.Metodo Principal()

Declarar ,crear e inciar objeto

SumaParesfor suma = new SumaParesfor();

Solicitar el ingreso de limite de números

Leer limite

For(numero.limite[i], i<= limite ,i++){

Solicitar el ingreso de el número

Leer numero[]

suma.SumaT();

Imprimir suma.Mensaje()

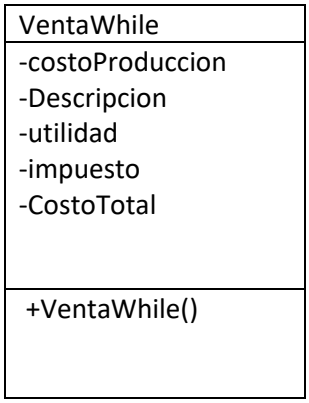
suma.SumaPares

Fin Metodo principal

Fin Clase EjecutaParesfor

1. DIAGRAMA DE CLASES

   Modelo



Controlador

Ejecuta

Vista?

2. Analicis del Problema

   Detectar objetos -> nombre
   producto

   Procesos ->
   utilidad=1.2 * costoProduccion;
   impuesto= 0.15 * (costoProduccion + utilidad);
   Salidas->
   Presentar: descripción
   costoProduccion
   utilidad
   impuesto
   costoTotal

3. Diseño del Programa

   1.Declarar Datos
   double costoProduccion=0;
   double impuesto=0;
   double utilidad =0;
   Int =0;
   String Descripcion;

   2.Metodo getImpuesto()

   impuesto= 0.15 * (costoProduccion + utilidad);

   Fin Metodo Utilidad()

   utilidad=1.2 * costoProduccion;

   Fin clase VentadoWhile

Clase EjecutaVentadoWhile

1.Metodo Principal()

Declarar ,crear e inciar objeto

VentadoWhile ventas = new VentadoWhile(nombre,producto);

Solicitar el ingreso nombre del producto

Leer nombre

Solicitar el ingreso precio del producto

Leer nombre

producto = sc.nextDouble();

Imprimir descripción

costoProduccion

impuesto

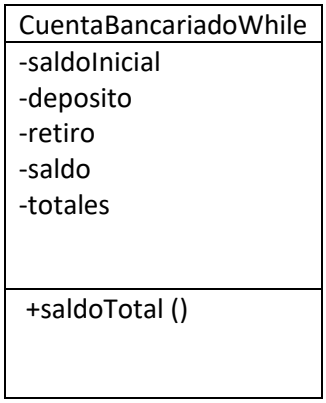
costoTotal

Fin Metodo principal

Fin Clase EjecutaVentadoWhile

1. DIAGRAMA DE CLASES

   Modelo



   Controlador

   EjecutaBancodoWhile

   Vista?

2. Analicis del Problema

   Detectar objetos -> nombre

   Saldo inicial

   Procesos ->

```
saldo = saldoInicial;
      if (desea == 'd')
         saldo = saldo + deposito;

      else
         if (desea == 'r')
            saldo = saldo - retiro;

      sumaD=sumaD+deposito;
      sumaR=sumaR+retiro;
      sumaS=sumaS+saldo;
```

   Salidas->sumaD

   sumaR

   sumaS

   Presentar: usuarios

   saldoIncial

   movimiento

   deposito

   retiro

   SumaD

   sumaR

3. Diseño del Programa

   1.Declarar Datos

```
double saldoInicial=0;
double depositar=0;
```

char desea;
char opcion;
String usuario;

2.Metodo saldoTotal()

saldo = saldoInicial;

if (desea == 'd') {

saldo = saldo + deposito;

else

if (desea == 'r') {

saldo = saldo - retiro;

sumaD=sumaD+deposito;

sumaR=sumaR+retiro;

sumaS=sumaS+saldo;

Fin clase CuentaBancariadoWhile

Clase EjecutaBancodoWhile

1.Metodo Principal()

Declarar ,crear e inciar objeto

CuentaBancariadoWhile cuenta = new CuentaBancariadoWhile

Solicitar el ingreso nombre del usuario

Leer nombre

Solicitar el ingreso del saldo Inicial

Leer saldoInicial

Imprimir usuario

saldoInicial

costoTotal

movimiento

deposito

retirar

sumaD

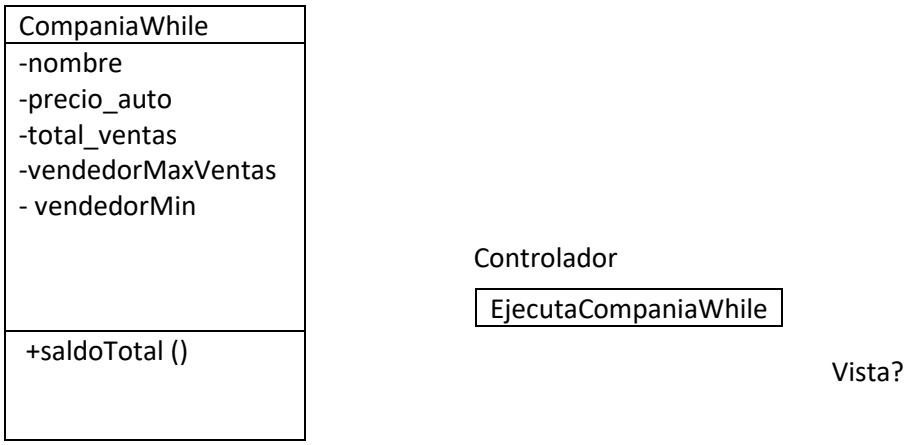
sumaR

Fin Metodo principal

Fin Clase EjecutaBancodoWhile

---

1. DIAGRAMA DE CLASES
   Modelo



2. Analicis del Problema
   Detectar objetos -> nombre
   autoVendido
   totalVentas
   valorAuto
   autoVendido
   Procesos ->
   saldo = saldoInicial;

```
if (cont == 0){
        ventaMin = totalVentas;
        vendedorMinVentas = objCompañia.getNombre();
    }
    if (totalVentas > ventaMax) {
        vendedorMaxVentas = objCompañia.getNombre();
    }
    if(totalVentas < ventaMin){
        vendedorMinVentas = objCompañia.getNombre();
```

   Salidas->vendedorMaxVentas
   vendedorMinVentas

   Presentar
   vendedorMaxVentas
   vendedorMinVentas

3. Diseño del Programa
   1.Declarar Datos

Char nuevo =’s’

String vendedorMaxVentas = ""

String vendedorMinVentas = ""

double ventaMax = 0

double ventaMin = 0

Fin clase CompaniaWhile

Clase EjecutaCompaniaWhile

1.Metodo Principal()

Declarar ,crear e inciar objeto CompaniaWhile objCompañia = new CompaniaWhile(nombre, totalVentas);

String vendedorMaxVentas = "";

String vendedorMinVentas = "";

double ventaMax = 0;

double ventaMin = 0;

Solicitar el ingreso nombre del vendedor

Leer nombre

Solicitar si Desea Ingresar otro auto vendido

Leer saldoInicial

Imprimir vendedorMaxVentas

vendedorMinVentas

Fin Metodo principal

Fin Clase EjecutaCompaniaWhile

1. DIAGRAMA DE CLASES

   Modelo

Controlador

EjecutaResta

Vista?

4. Analicis del Problema

   Detectar objetos -> valor1
   Valor2

   Procesos ->
   resta=0;
   while (valor1>=valor2){
   valor1=valor1-valor2;
   resta++

   Salidas-> resta
   Presentar: Resta

5. Diseño del Programa

   1.Declarar Datos
   Int valor1=0;
   Int valor2 =0;
   Int resta=0;

   2.Metodo Dividir()

resta=0;

while (valor1>=valor2){

valor1=valor1-valor2;

resta++;

Fin Metodo Dividir()

Fin clase RestaRepetitiva

Clase EjecutaResta

1.Metodo Principal()

Declarar ,crear e inciar objeto

RestaRepetitiva resta = new RestaRepetitiva()

Solicitar el ingreso del dividendo

Leer valor1

Solicitar el ingreso del divisor

Leer valor2

Establecer resta.Dividir()

Imprimir resta.Resta()

Fin Metodo principal

Fin Clase EjecutaResta

---

1. DIAGRAMA DE CLASES

   Modelo

MultiplicacionEnteros
-valo1 -valor2 -producto
+Dividir()

Controlador

EjecutaMultiplicacion

Vista?

2. Analicis del Problema

   Detectar objetos -> valor1
   Valor2

   Procesos ->

```
for (int i = 1; i <=valor2; i++) {
        producto = producto +valor1;
```

   Salidas-> producto

   Presentar: producto

3. Diseño del Programa
   1.Declarar Datos
   Int valor1=0;
   Int valor2 =0;
   Int producto=0;

   2.Metodo Multiplicar()

for (int i = 1; i <=valor2; i++) {

   producto = producto +valor1;

Fin Metodo Multiplicar()

Fin clase MultiplicacionEnteros

Clase EjecutaMultiplicacion

1.Metodo Principal()

Declarar ,crear e inciar objeto

   MultiplicacionEnteros enteros = new MultiplicacionEnteros()

Solicitar el ingreso del valor1

Leer valor1

Solicitar el ingreso del valor2

Leer valor2

Establecer enteros.Dividir()

Imprimir resta.Multiplicar()

Fin Metodo principal

Fin Clase EjecutaMultiplicacion