EJERCICIOS

TAD FechaHora:

Tipo:

FechaHora = registro

día: Entero

mes: Entero

año: Entero

hora: Entero

minuto: Entero

segundo: Entero

zona: Entero

fin Registro

PRE:{fecha1 >= fecha 2}

proced restarFechas(fecha1, fecha2: FechaHora, ref resta: FechaHora)

si fecha1 > fecha2 entonces

resta.segundo 🡨 fecha1. Minuto- fecha2.minuto

resta.minuto🡨 fecha1.minuto – fecha2.minuto

resta.hora🡨 fecha1.hora – fecha2.hora

resta.dia🡨 fecha1.dia – fecha2.dia

resta.mes 🡨 fecha1. Mes – fecha2.mes

resta.año🡨 fecha1.año – fecha2.año

sino

resta🡨 0

fin si

fin proced

Tipo:

Edificio = registro

nombre: Cadena

direccion: Direccion

pisos: Lista(Piso)

cocheras: Lista(Cochera)

ascensores: Lista(Ascensor)

finRegistro

funcion cantidadPisos (edificio: Edificio): Entero

var loc: cont , x: entero

si no lVacia (edificio.piso) entonces

lPpio (edificio.piso)

lInfo(edificio.piso, x)

cont🡨 cont.. + x

lSig (edificio.piso)

mientras no lFin (edificio.piso) entonces

lInfo(edificio.piso, x)

cont🡨 cont.. + x

lSig (edificio.piso)

fin mientras

devolver (cont)

fin funcion

clase 17/3

contador con TAD:

TAD contador:

Tipo:

Contador: entero (real, lista(entero))

Proced inicializarCont (ref contador: Contador)

contador🡨 0

fin proced

Proced IncrementarCont (ref contador: Contador)

contador🡨 contador+ 1

fin proced

Funcion leerCont ( contador: Contador): entero

Devolver (contador)

Fin fun

Proced reiniciarCont (ref contador: Contador)

InicializarCont (contador)

Fin proced

Proced decrementarCont (ref contador: Contador)

Si contador > 0 entonces

Contador 🡨 contador- 1

Fin si

Fin proced

TAD contador:

Tipo:

Contador: registro

Actual: real

Incremento: real

Fin reg

Proced inicializarCont (ref contador: Contador, incremento: real)

Contador.actual🡨 0

Contador.incremento🡨 incremento

fin proced

Proced IncrementarCont (ref contador: Contador)

Si contador.actual < MAX\_REAL entonces

Contador.actual🡨 contador.actual + contador.incremento

Fin si

fin proced

Funcion leerCont ( contador: Contador): real

Devolver (contador.actual)

Fin fun

Proced reiniciarCont (ref contador: Contador)

InicializarCont (contador, contador.incremento)

Fin proced

Proced decrementarCont (ref contador: Contador)

Si contador.actual > 0 entonces

Contador.acutal🡨 contador.acutal – contador incremento

Fin si

Fin proced