6/06/23

SAMBIDO ADRIAN A. Leg: 4091/1.

TEMA 1 REGIS ARCHIVOS, SECUENCIALES ARBOLES TURNO TARDE

PROGRAM PARCIAL;

T1	A
T2	D
T3	-

```
CONST
  VALOR ALTO = 9999;

TYPE
  PRODUCTO = RECORD
               CODIGO : INTEGER;
               NOMBRE : STRING;
               DESCRIPCION: STRING;
               PCOMPRA: REAL;
               PVENTA : REAL;
               UBICACION : STRING;
             END;

ARCHIVO = FILE OF PRODUCTO;

PROCEDURE LEER (VAR maestro : ARCHIVO; VAR reg : PRODUCTO);
BEGIN
  IF (NOT EOF(maestro)) THEN { ES UTILIZADA EN EXISTE PRODUCTO }
    READ (maestro, reg);
  ELSE
    reg.codigo := VALORALTO;
END.

PROCEDURE LEERDATOS (VAR reg : PRODUCTO);
BEGIN
  WRITELN ('INGRESE CODIGO'); READLN (reg.codigo);
  WRITELN ('INGRESE NOMBRE'); READLN (reg.nombre);
  WRITELN ('INGRESE DESCRIPCION'); READLN (reg.descripcion);
  WRITELN ('INGRESE PRECIO COMPRA'); READLN (reg.pcompra);
  WRITELN ('INGRESE PRECIO VENTA'); READLN (reg.pventa);
  WRITELN ('INGRESE UBICACION'); READLN (reg.ubicacion);
END;

FUNCTION EXISTE PRODUCTO (VAR maestro : ARCHIVO; codigo : INTEGER);
```

SIMBIDO ADRIAN A. LEG 40911
TEMA 1 RESOLUCIÓN ARCHIVOS SECUENCIALES 6/06/23 TURNO TARDE

```
PROCEDURE AgregarProducto (VAR maestro: ARCHIVO);
VAR
  Aux, Reg: Producto;
Begin
  LeerDatos(Reg);
  IF (ExisteProducto(maestro, reg.codigo)) then
    WriteLn('El Producto ingresado existe');
  ELSE Begin
    Reset(maestro);
    Read(maestro, Aux);
    IF (Aux.Codigo <> 0) then Begin
      Seek(maestro, (Aux.Codigo * -1));
      Read(maestro, Aux);
      Seek(maestro, FilePos(maestro) - 1);
      Write(maestro, Reg);
      Seek(maestro, 0);
      Write(maestro, Aux);
    End;
    ELSE
      Seek(maestro, FileSize);   (Es una función que recibe un archivo)
      Write(maestro, Reg);
    Close(maestro);
End;
```

RENDI: ARCHIVOS SECUENCIALES Y ÁRBOLES

TEMA 1

SAMBUCETTI ADRIANA LEG 40911/1

6/06/23 TURNO: TARDE

```
PROCEDURE quitarProducto (VAR archivo: ARCHIVO);
VAR
POS, cod: integer;
reg: PRODUCTO;
  Begin

    WriteLn('INGRESE EL CODIGO DE PRODUCTO A ELIMINAR');
    Readln(cod);
    IF (existeProducto(archivo, cod)) then  Begin
      Reset(archivo);
      Read(archivo, reg);
      POS := Reg.codigo;
      While (reg.codigo <> cod) do  Begin
        Read(archivo, reg);
      end

      SEEK(archivo, FILEPOS(archivo) - 1);
      reg.codigo := POS;
      POS := FILEPOS(archivo) * -1;
      Write(archivo, reg);
      SEEK(archivo, 0);
      Reg.codigo := POS;
      WRITE(archivo, Reg);
      CLOSE(archivo);
  end;

    Else
      WriteLn('EL PRODUCTO QUE DESEA ELIMINAR NO EXISTE')
end;
```

Hoja 4/4

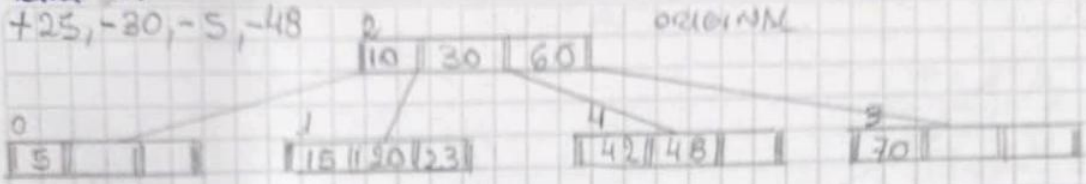
Recu: Archivos secuenciales, índices 6/06/23

SAMUDIO ADRIAN Leg 40911/1

turno: tarde

Tema 1

+25, -30, -5, -48



orden N

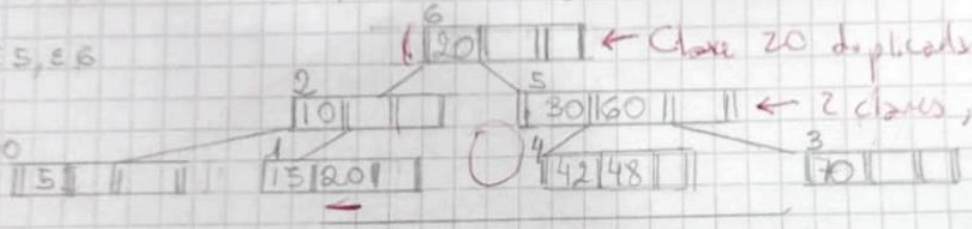
+25 va al nodo "1" y se produce overflow. Se separa

L2, L1 Se produce overflow en nodo "2"

15 20 23,25 10, 20 30 60,

Faltan L/E Promoción incorrecta

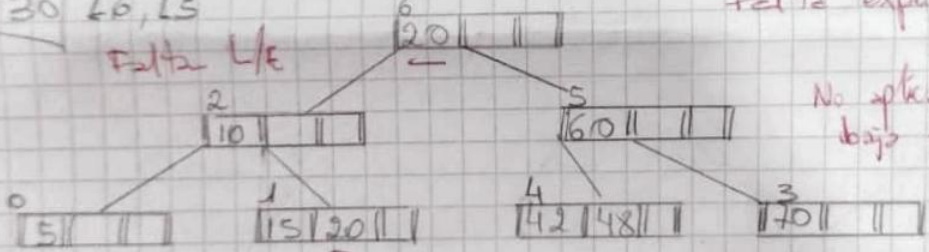
E5, E6



-30 L6, L5

Falta explicación

Falta L/E



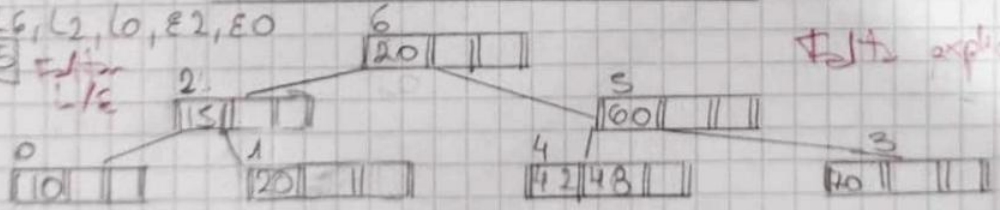
No aplica algoritmo de baja de nodo interno

Admite error

L6, L2, L0, E2, E0

-5 Falta L/E

Falta explicación

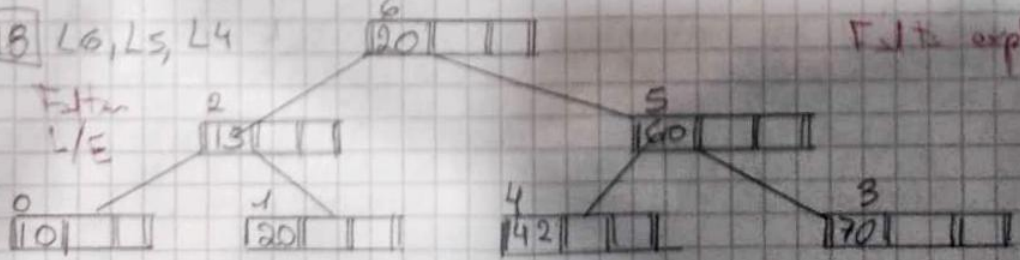


Admite error

-48 L6, L5, L4

Falta L/E

Falta explicación



Admite error