CAPÍTULO I: CONCEPTOS BÁSICOS

- Programa: Es un conjunto de instrucciones, ejecutables sobre una computadora, que permite cumplir una función específica.
- Dato: Es una representación de un objeto del mundo real mediante la cual se pueden modelizar aspectos de un problema.
- Abstracción: Es la interpretación de los aspectos esenciales de un problema del mundo real, y la expresión de los mismos en términos precisos.
- Precondición: Es una información que se conoce como verdadera antes de iniciar el programa.
- Poscondición: Es una información que debiera ser verdadera al concluir el programa, si se cumple con lo pedido.
- Especificación: Es el objetivo que se desea que el programa realice.
- Lenguaje de Programación: Es el conjunto de instrucciones permitidas y definidas por sus reglas sintácticas y su valor semántico, para la expresión de soluciones de problemas.
- Algoritmo: Es la especificación rigurosa de la secuencia de instrucciones a realizar sobre una computadora, para alcanzar el resultado deseado en un tiempo finito.

人名斯特人名 电影性电影 化氯化 化二氯化二氯化二氯化氯化氯

CAPÍTULO II: ALGORITMOS

1) ESTRUCTURAS DE CONTROL

Son aquellas estructuras que controlan el flujo de control de un programa.

gag tegresse i terretre de la cita elementa, edecario a la filla de la sestitus a vivadando e

Se encuentran divididas en 5 categorías:

- Secuencia: Estructura compuesta por una sucesión de operaciones, en la que el orden de ejecución coincide con el orden físico de aparición de las instrucciones.
- Decisión: Estructura que permite elegir entre dos alternativas (bifurcación) dependiendo de condición (expresión booleana: True o False) definida en ella.

Pueden ser bifurcaciones simples, en cascada o anidadas:

```
If (A < B) then
Si (Condición) entonces
  <<Acciones por condición verdadera>>
                                                     Min := A
                                                   Else
                                                     Min := B:
  <<Acciones por condición falsa>>
Fin Si
Si (Condición1) y (Condición2) entonces
                                                   If (A < B) and
                                                                  (A < C)
                                                    Min := A
  <<Acciones por condiciones verdaderas>>, ....
                                                   Else
                                                     If (B < A)
  Si (Condición1) y (Condición3) entonces
                                                                   and (B < C)
    <<Acciones por condiciones verdaderas>>
                                                   then
                                                       Min := B
    <<Acciones por al menos una condición falsa>>
                                                     Else
                                                       Min := C;
  Fin Si
Fin Si
                                                   If (A < B) then
Si (Condición1) entonces
  Si (Condición2) entonces
                                                     Begin
                                                       If (A < 5) then
    <<Acciones por condición verdadera>>
                                                         Min5 := A
    <<Acciones por condición falsa>>
                                                       Else
                                                         Min5 := -1;
  Fin Si
Fin Si
```

 Selección: Estructura que permite elegir entre dos o más alternativas dependiendo de la variable ordinal que interviene.

Iteración: Estructura que se ejecuta un número desconocido de veces dependiendo de la condición definida en ella.

La iteración puede ser pre-condicional (se evalúa la condición primero) o pos-condicional (se evalúa la condición al final). En el primer caso, si la condición es verdadera, se ejecutan todas las instrucciones inmediatas, haciendo que el bloque itere 0, 1 ó más veces. En el segundo caso, el bloque deja de ejecutarse cuando la condición es verdadera, por lo tanto, se itera al menos una vez todas las instrucciones inmediatas.

| Mientras (Condición) hacer | Readln (A); |
|--|-------------------|
| < <acciones condición="" por="" verdadera="">></acciones> | While (A <= 0) do |

| derition modern | uše komite ali vilozmornova i kojemnime | gamment setmini egReadln.(A); egmgft serett 🧸 | |
|---|---|---|---|
| Repetir | | Repeat the period of the control of | |
| < <acci< th=""><th>ones por Condición falsa></th><th></th><th>ŀ</th></acci<> | ones por Condición falsa> | | ŀ |
| Hasta (Co | ndición) | Until (A > 0); | |

Repetición: Estructura que se ejecuta un número fijo de veces, que puede ser conocido de antemano (ValorMáximo - ValorMínimo + 1), hasta que Índice supere ValorMáximo (o ValorMínimo). Indice se incrementa (o decrementa) al siguiente valor de acuerdo al tipo de variable ordinal asociada; en algunos lenguajes el incremento (o decremento) es automático y en otros, se puede establecer.

El valor de Índice solo se utiliza dentro de está estructura, por lo que, en Pascal, no es posible modificarla y el valor de la misma se pierde al terminar la repetición.

| Para Índice := ValorMínimo a ValorMáximo < <acciones>></acciones> | hacer | For X:= 1 to 10 do Writeln (X); |
|--|------------|---|
| Para Índice := ValorMáximo a ValorMínimo < <acciones>></acciones> | hager, ,,, | For X; := 10, downto 1, do; () Writeln (X); |

On algorithm from thosperencedy, permits you so insertainiente see mis cipide y seachle, rescins a que so

2) ESTRUCTURA ESQUEMÁTICA DE UN PROGRAM<mark>A EN PAS</mark>CAL

Program Identificador Programa;

Uses Nombre Unidades;

When the second continues to the second continues the continues of the second continues to the

Const

{Declaraciones de constantes globales}

Type

"a" {Declaraciones de tipo<mark>s de</mark> dato<mark>s globales definidos por el programador}</mark>

Procedure Identificador Procedimiento (Parametros)

-2992 com (Instrucciones ejecutables) non charactor obseno (apenda torbits e continues a)

End;

Tunction Identificador Funcion (Parámetros): Valor Retornado; (2006) (1906) (1906) Begin

{Instrucciones ejecutables}

End:

318C ,

Var

exelect of {Declaraciones ide variables | globales} may relifican removable receive any about a receive of the control of the

Begin

{Instrucciones ejecutables}
End.

MARKETER PROPERTY OF MADE DEPOSITE STREET (A

gent at ear were real and amore as an I

区CETALNESS, 红皮皮肤及

- e a**o**nte<mark>Program: Indica el inicio del programa</mark>losa la suna**ç**amente se adeixano, venolt-qu'é una excluire anticament mos construcción de comente con entrapolar anne, ha A. su**p**enpera circo consideración ou consideración como esta po
 - Nombre Unidades: Identificador Procedimiento, Identificador Funcion y Nombre Unidades: Identifican a los procedimientos y funciones, así como a la aplicación y a las unidades. No puede repetirse ningún identificador cualquiera sea su pertenencia.

- Uses: Permite utilizar los procedimientos, funciones, variables, constantes y/o tipos de datos definidos que pertenezcan a otras unidades.
- Const: Permite definir datos que no variaran a lo largo de la aplicación.
- Type: Permite definir tipos de datos propios utilizando los definidos por el lenguaje.
- Var: Permite definir las variables que se utilizarán, indicando el tipo de dato a la que pertenecen.
- Procedure y Function: Módulos que contienen algoritmos. Su utilización facilità el desarrollo de los programas y agrega legibilidad al mismo.
- Begin y End: Establecen el inicio y fin de los bloques de instrucciones. Cuando aparece un ";" después del End se indica el fin de un bloque común, pero cuando se encuentra un "." Se indica el fin de la aplicación.

3) IMPORTANCIA DE LA DOCUMENTACIÓN DE UN ALGO<mark>RITMO</mark>

Un algoritmo bien documentado, permite que su mantenimiento sea más rápido y sencillo, gracias a que su legibilidad está beneficiada por los comentarios lógicos y por las nomenclaturas de los identificadores:

Comentarios Lógicos: Aportan una gran legibilidad al código, dado que su único propósito es ese, especificando, de la manera más completa posible, cual es el problema a resolver. Siendo su lectura lo suficientemente clara como para entender cuales son las acciones que llevan a cabo los herederos de estos.

Los comentarios son una parte importante del programa por lo tanto, su uso es altamente recomendado.

• Nomenclaturas: Aportan legibilidad al código cuando estas son autoexplicativas (deben estar relacionadas con el propósito del módulo correspondiente).

4) IMPORTANCIA DE LA VE<mark>RIFIC</mark>AC<mark>IÓN</mark>

Un algoritmo es eficaz (correcto) cuando se cumplen con las metas pautadas. Por lo tanto, se debe tener una especificación completa, precisa y libre de ambigüedades del problema a resolver.

A través de pruebas con datos reales, se puede comprobar si el algoritmo escrito, cumple con los requisitos pedidos.

5) EFICIENCIA DE UN ALGORITMO

Un algoritmo es eficiente cuando utiliza de la mejor manera posible, los recursos a su disposición (tiempo y memorias).

Dado que existen varias soluciones posibles para un problema, la solución más eficiente es aquella que se realiza en el menor tiempo posible, con la menor cantidad de pasos y uso de memorias.

6) DESCOMPOSICIÓN DE PROBLEMAS

MODULARIZACIÓN

También conocida como *Top-Down*, consiste en descomponer el problema principal en sub-problemas o módulos, y su vez, estos sub-problemas en sub-problemas más pequeños. A tal punto de que las resoluciones de los mismos sean independientes y más simples; y la suma de estos, la solución del problema principal.

Por esto, la importancia de la modularización, sobretodo una buena, recae en A conceptos básicos:

estivi ediceled ideadifican a sea measidadaeux y fanciones ad semo a fa salicación y a faa uni-

Firedeededte

TipoE

1.1700

- Conocimiento Humano: Dada la limitada capacidad de retención de información que el ser humano posee, la resolución de un problema resulta más fácil si esta se realiza por partes. To solve at early applicable as two solds, we always a good
- Trabajo en equipo: Permite que un programa complejo se desarrolle más rápido, dado que las diversas tareas se dividen entre los programadores, haciendo que las soluciones de estas se haga en paralelo.

la primarità di comatta con applicare coi cata militara di cata

El programador solo se preocupa por solucionar su problema, y si necesita utilizar los módulos creados por sus pares, lo hace con las instrucciones y limitaciones correspondientes.

Mantenimiento del programa: Tanto para corregir los errores como para modificar el código, es más costoso mantener un programa que no ha sido correctamente modularizado de aquel que sí lo fue.

De esta comerci, el dato esta precebbo de las produces que realibre es médido receptor, ya que no abreca

Reusabilidad del código: Aquellos códigos que fueron correctamente modularizados, pueden ser reutilizados, siempre y cuando sirvan en la resolución de otros problemas, tanto en el mismo programa como en e Arko er etter Kartas 👉 franca aller eller eller eller etter betatte etter kommer eller etter eller etter etter eller etter eller eller eller etter eller etter eller eller etter eller etter eller eller etter eller eller

statyone air stranscriptida isasida sistem se statues n

Cuando un módulo se ejecuta, este toma el control del programa hasta que su liberación (cuando se ejecuta la última sentencia del mismo) retorna el control a la instrucción siguiente a la invocación (en el caso de los procedimientos) o la misma que la invoco (en el caso de las funciones). Tanto las funciones como los procedimientos deben ser declarados antes de ser utilizados. El citor el mindratificados, el mindratificados en estamban produce elegence el company en el company el mentione el company el compan

endidad of tanned consuctang for help within the calculation active and they are in a construction of the all camber

ALCANCE DE LOS DATOS

A State of the sta

- Dato Global: Es aquel que está declarado en la sección Var del programa. Su uso está recomendado cuando se necesitan datos que son utilizados por varios módulos.
- Dato Local: Es aquel que está declarado dentro de un módulo, en la correspondiente sección Var, y su uso esta restringido a dicho módulo. BOHOLES /

Un dato perteneciente a un módulo está oculto, porque no puede ser utilizado por otros y por lo tanto está protegido.

of complete and removal and remove an emplify decision removed

- nariq sero meta cis solitibile coi sottema di sonstantite gene coi seg schero? Less autro de meta l'ischera le ergens, par el montre de la semanar a contribue de l'écaptifications, que un pares destinance. Des la rome L'elle de l'expection de la la semana de la les la terres de la sanctification de la la contribue de l'estate
- a property of the countries of a section such as a

อาเมียงแล้วแบบ แบบ คน ในปฏิกันแทนแบบ แบบ ปฏิกัน แบบ แบ้ ได้ เกิด สุดยังสามารั

- i starger teknik berganat berganasan garanter delam antanjak ana incipaci de konganungan kanankon ind to Changes in the the Armer's in Spanic Carriers in English early in the continue of the Carrier in the Car
- -ka si beli lang ali a layak sar yirah selasa kepatanggan sebih kalabaya ali sebesah de dibendag ayirah dipanag as an egizar for invitable to the second of the sequence of the second function server, comp
- A de variet es, filo opres trabantiços en pue la defere car con escribir de la estado. and the comment and the confidence of the

PARÁMETROS a los la sem neligiour den el neronome de limbologica discumil di abitel e contacte con escele

Son variables que se utilizan para transferir información entre los módulos, por lo tanto, la alternativa al uso de variables globales, est superior alternación entre los módulos, por lo tanto, la alternativa al uso de variables globales, est superior alternación entre los módulos, por lo tanto, la alternativa al uso de variables globales, est superior alternación entre los módulos, por lo tanto, la alternativa al uso de variables globales, est superior alternación entre los módulos, por lo tanto, la alternativa al uso de variables globales.

estage not retilize se teso le haid entra diasse partitiona un es apropiaco.

Distance as against valve sit sometroides are supplied that is a distance of value and the same troit.

e Parámetro por Valor: Son aquellos que se utilizan cuando se quiere mandar una copia del dato.

no energina terri o marcilo as cinas vas defenicación de secultario de la laborativa de la la cabración de la c

De esta manera, el dato está protegido de las modificaciones que realice el módulo receptor, ya que no afecta a la variable pasada (parámetro actual), sino a la copia de la misma.

También son conocidos como parámetros de Entrada o de "Solo lectura", porque envían información al módulo.

• Parámetro por Referencia: Son aquellos se utilizan cuando se quiere modificar el valor de la variable pasada o cuando se quiere obtener información de un módulo.

En estos casos, no se envía una copia sino la dirección de memoria de la misma. Por lo tanto, los cambios efectuados en ella se ven reflejados al finalizar el módulo.

El parámetro por referencia, también conocido como parámetro de Salida, de Entrada/Salida, de "Solo Escritura" o de "Lectura y Escritura", se declara anteponiendo, al identificador del parámetro formal, la palabra Var.

El uso de parametros versus variables globales está fundado en 4 conceptos:

- Integridad de los Datos: Se puede identificar cuales son las variables globales que usan y/o comparten entre sí los módulos, así como también el uso que hacen de ellas, tanto para lectura como para escritura, o lectura y escritura.
- Protección de los Datos: Se evita que se modifique por accidente el valor de una variable, si esta es pasada como un parametro en lugar de actuar directamente sobre ella.
- Utilidad del uso de Parámetros: Permite que los programadores desarrollen los módulos sin tener que preocuparse por el nombre de los parámetros actuales, o identificadores, que sus pares declararon. Por lo tanto facilita el desarrollo de programas modularizados y la corrección de los posibles errores en ellos.
- Reutilización de códigos: Permite que un módulo pueda ser reutilizado varias veces (usando distintas listas de parámetros actuales) dado que los parámetros declarados en él, separan el nombre del dato del dato en sí.

Los parámetros se declaran en la cabecera de los módulos, indicando nombre y tipo de dato.

Existen dos formas de correspondencia entre parámetros:

- Por posición: En Pascal, los parámetros actuales deben coincidir con los tipos de los parámetros formales. Correspondiendo el primer parámetro actual con el primero formal, el segundo actual con el segundo formal, y así sucesivamente.
 - También deben coincidir en número; la cantidad de parámetros actuales debe ser igual a la cantidad de parámetros formales, de lo contrario, en Pascal, se abortará el programa.
- Por nombre: En otros lenguajes, se puede indicar en la invocación del módulo, que parámetro formal corresponderá a que parámetro actual.

ODIOOLOTZ O III OHE

contrait converse he would

CAPITULO III: TIPOS DE DATOS SIMPLES

1) TIPOS DE DATOS

Es una clase de objetos ligados a un conjunto de operaciones para crearlos y manipularlos, habito de operaciones para crearlos y manipularlos de operaciones para crearlos de operaciones

the analysis of the garden search and the contract statement between the contract of the charge and the contract of

Poseen un rango de valores posibles, un conjunto de operaciones realizables y representación interna, todos ellos, predefinidos y acotados por el lenguaje de programación. Cuando se declara uno, se está indicando cual es la clase de valores que poseerá y cuales son las operaciones que se pueden hacer sobre él.

Los tipos de datos simples se clasificament: Palet (Engle) militaria a (Boote alliangation and Allian Alliangation and Allian

TIPO DE DATO NUMÉRICO

Es un conjunto de valores numéricos que se pueden representar de dos formas:

Enteros: Es el tipo de dato numérico más simples y posee un rango de representación limitado por el número de bytes disponibles: en los enteros con signo, el rango es de -maxint <= n <= maxint, siendo n el número que se quiere representar. En los enteros sin signo, el rango es de 0 <= n <= maxint.

completed and so remedences seeks of refulf

In medical despendencies para para inscripcion describiros en

Reales: Es el tipo de dato numérico que permiten representar números fraccionarios bajo la norma IEEE 754.
 El rango de los reales depende de la disponibilidad de bytes que se tengan, por lo tanto, su representación es limitada y aproximada.

Las operaciones válidas que se realizan sobre este tipo de dato son:

- Asignación: Mediante := se puede asignar un valor o el contenido de una variable numérica a otra.
- Aritméticas: Suma (+), Resta (-), División entera (Div), Módulo (Mod) y División real (/).

Tanto la suma, la resta y la multiplicación aceptan operandos enteros como reales y, por lo tanto, su resultado puede ser entero o real. Sin embargo, la división entera restringe a los operandos, debiendo ser estos enteros. Se utiliza Div (que devuelve el resultado entero) y Mod (que retorna el resto), y el resultado es un entero. En el caso de la división real, no existe ninguna limitación con los operandos. Se utiliza / (que devuelve el resultado con parte fraccionaria) y el resultado puede ser entero o real.

reproductive and the metric to intentition paper () whether and the

isoacentrani elevis e dicateresad la 1702 è chimie i sto 1906, i sici 300 1706 esc

Common wagang bag makini bak operang onto againful

Commence of the Commence of th

Ministry of the second of the

Cuando se realizan estas operaciones, el resultado de las mismas puede estar fuera del rango de representación. Si se supera el máximo valor binario permitido, se produce un *Overflow*. Si el número está por debajo del mínimo valor binario permitido, se produce un *Underflow*. Ambos, son desbordamientos que algunos sistemas los detectan como errores y otros los ignoran convirtiéndolos en un resultado incorrecto dentro del rango del tipo.

El orden de precedencia, cuando existen varios operandos en la expresión, está regido por las mismas reglas que en las Matemáticas:

- Los paréntesis (), que alteran el orden en las operaciones.
 - Operadores *, /, Div y Mod
 - Operadores +, -.
 - De Comparación: Igualdad (=), Desigualdad (<>), Menor que (<), Menor o Igual que (<=), Mayor que (>) y Mayor o Igual que (>=).

El resultado de estas operaciones es un booleano.

TIPO DE DATO LÓGICO

Es un tipo de dato que puede tomar uno de los dos valores posibles, Verdadero (True) o Falso (False), pero no ambos al mismo tiempo. EO FA GRA POPET ()

Se utiliza en casos en donde-se representan dos alternativas a una condición, igno que a conseguir en casos en donde-se representan dos alternativas a una condición, igno que a conseguir en casos en donde-se representan dos alternativas a una condición, igno que a conseguir en casos en donde-se representan dos alternativas a una condición, igno que a conseguir en casos en donde-se representan dos alternativas a una condición, igno que a conseguir en casos en donde-se representan dos alternativas a una condición, igno que a conseguir en casos en donde-se representan dos alternativas a una condición, igno que a conseguir en casos en donde-se representan dos alternativas a una condición, igno que a conseguir en caso en conseguir en caso en conseguir en conseguir en caso en caso en conseguir en caso en c

Las operaciones válidas que se realizan sobre este tipo de dato son: productions of accordes to the property of the property according to decream and, so are tradequated and as tail about

- Asignación: Mediante := se puede asignar un valor o el contenido de una variable booleana a otra.
- Lógicas o Booleanas: Negación (Not), Conjunción (And), Disyunción Incluyente (Or) y Disyunción Excluyente (XOx).

1

El resultado de estas operaciones es un booleano:

had well the region of

actions as the trees

| | Þ | - 2 | P And Q |
|---------------------|-----------|---------|---------|
| acarana is was one | True True | 7. True | na True |
| to endurance of the | True | False | False |
| | False | True | False |
| | False | False | False |

| | P | Q | P Or Q | |
|------|----------|--------|--|--|
| c⊊{e | maTrue a | True : | eqTrue \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ | |
| . (| a True 👵 | False | Ada True by and ob- | |
| 24 3 | False | True | Fig. True | |
| | False | False | False | |

False

True

rendrači v ob obradansko ago je vša<mark>vad</mark> se odgo se brodana konota i obradajeve sa kla

An it properties the decrease and many perfect the entire series in an area by a second to the collection of the collection in the collect etz akiroalarizetekka k

| Pro- | Q | P XOr Q |
|-------|-------|---------|
| True | True | False |
| True | False | True |
| False | True | True |
| False | False | False 🛶 |

by pip occurrence trades and that i de como polición é estadi de como está

men in makkaman shiking mana shi abkazare a kata ada ina misa nen yilishiya. in propublicate vicinates per la

False

El orden de precedencia para los operadores lógicos es:

a dae haar damma (4). Restect of theisida artern (Daw) didinh (Most). Birisian real (7).

Los paréntesis (), que alteran el orden en las operaciones.

est us conced Operador Not energiously colors ago marques normaliques no cure si concede de anno el anno el come s entena parako <mark>operador And</mark>erak signistra mare anto en do entendenda da internaciona de entendenda antendenda esta on so charle at Operadores of y xorrup) box / (casus charlesce i), a remain supe valle mide se lecrado. est and a case de la drivina casa no event alla la como de la como como la como la como la como como como como reals a el continuo con parta il accionación y la confinio puede ser conera o real.

• De Comparación: Igualdad (=), Desigualdad (<>), Menor que (<), Menor o Igual que (<=), Mayor que -sugar al(>) y Mayor o Igual que (>≌), sacrear au car abbilicar la l'acciónicació subse continús es abbandi. spineson in se separa el máximo cabo becara paramida, se produce un Oscallan. El resultado de estas operaciones es un booleano. corporation organizations controlled

TIPO DE DATO CARACTER

ili *araba de precedencia.* espado existen a dios operándos en la experioro, la

Es un tipo de dato que contiene solo un elemento como su valor, establecido y normatizado por los estándares ASCCI, EBCDIC, ANSI, etc.; siendo ASCII el más común a escala internacional.

Tanto los elementos como el orden de precedencia de los mismos, está establecido por el estándar que se utilice. Sin * (haradares British Marketine embargo, los caracteres se dividen en 4 grupos comunes:

a Consideras

Letras Minúsculas: 'a', 'b', 'c',... 'x', 'y', 'z'.

The Letras Mayusculas: A, B, C, A, Y, Y, Y, (2). (3) bublingised (5) bublings the analysis of the Sund's come income or reversely 7 / 1/2

Dígitos: '0', '1', '2',... '8', '9'

connectiond as a communication expect the challenger CI

Las operaciones válidas sobre este tipo de dato son:

- Asignación: Mediante: = se puede asignar un valor o el contenido de una variable caracter a otra.
 - De Comparación: Igualdad (=), Desigualdad (<>), Menor que (<), Menor o Igual que (<=), Mayor que (>) y Mayor o Igual que (>=).

and state of the compact of the comp

Las comparaciones entre caracteres se realizan por el valor ordinal de los mismos, siendo el resultado de estas un booleano.

2) CONSTANTES Y VARIABLES

 Variable: Es una posición de memoria que almacena información que puede variar a lo largo de la ejecución del programa.

La declaración, en Pascal, se logra anteponiendo al identificador de esta, la palabra reservada Var. Y colocando detrás de este, los: más el tipo de dato (Integer, Real, Boolean, Char...).

oro la emaya albante a maralla la **var** Nombre Va<mark>riable: Tipobato</mark>;

La asignación de valores, en Pascal, se hace de la manera:

NombreVariable := Expresión;

Donde Nombre Variable se encuentra a la izquierda e indicando cual es la variable que recibirá a Expresión, que se encuentra del lado derecho. Ambos, deben pertenecer al mismo tipo de dato.

• Constante: Es una posición de memoria que almacena información que no varia a lo largo de la ejecución del programa.

La declaración, en Pascal, se logra anteponiendo al identificador de esta, la palabra reservada Const. Y colocando detrás de este, el signo = más Expresión:

Const NombreConstante = Expresión.

the profit and according to the profit of th

Construit become actific mentioners of

El tipo de dato (Integer, Real, Boolean, Char...) no necesita ser declarado, aunque queda implícito.

Tanto las variables como las constantes, utilizan *Identificadores*, que las referencian en lugar de la posición real de memoria y su valor, y deben ser declarados antes de poder ser utilizados. Los identificadores están formados por letras, dígitos y algunos caracteres especiales, con la restricción que el primer carácter debe ser una letra o un "_".

3) TIPOS ORDINALES

Un tipo de dato es ordinal si para cada elemento, existe uno anterior y otro posterior (siempre y cuando no se trate de los extremos del rango), que pertenezcan al mismo tipo de dato.

Por lo tanto, los enteros, los caracteres y los booleanos son tipos de datos ordinales, y pueden ser utilizados como variables de control en las estructuras de control. En cambio, los reales quedan excluidos porque no es posible saber el antecesor o el sucesor de ellos, ya que estos son infinitos.

4) TIPOS DE DATOS DEFINIDOS POR EL USUARIO

Son aquellos tipos de datos que son definidos por el usuario, a través de la palabra reservada Type, y que no existen en la definición del lenguaje. there exists the north edges are down and this to we produce with the

Por lo tanto, es el usuario el que se encargará de establecer el conjunto de valores permitidos y las operaciones válidas sobre él.

Cuando un lenguaje permite definir tipos de datos propios, y permite usarlos y validarlos del mismo modo que los tipos de datos estándares, se tiene un lenguaje con:

- Mejor posibilidad de abstracción de datos: Permitiendo una mayor riqueza expresiva del lenguaje, ya que agrega claridad a la lectura del programa.
- Mayor seguridad en las operaciones: Permitiendo un mayor número de validaciones sobre el tipo del dato.
- Limitaciones de los valores: Prestableciendo los valores posibles que pueden tomar las variables pertenecientes al tipo de dato propio.
- Mayor Flexibilidad: Si se desea modificar la representación de la información contenida en un tipo de dato propio, basta con cambiar la declaración del tipo, en lugar de alterar una serie de declaraciones de variables.
 - Mayor Seguridad: Reduciendo los errores de correspondencia entre el valor que se pretende asignar y el previamente declarado.

Extension of the result of the American

La declaración de tipos es de la forma:

Type NombreTipo = TipoBase.

Donde NombreTipo es el identificador con el se conocerá al tipo en el programa, y TipoBase puede ser un tipo de dato predefinido o estándar. Este identificador puede ser usado en declaraciones de tipos, variables y módulos posteriores.

Considered I is that position of considerate procedures introdución que en contrata de trace o de la vicación

TIPO DE DATO ENUMERATIVO

Es un tipo de dato escalar formado por valores constantes strings simbólicos. angle to little a object of larger of any

obsidym obwę oppowa z brodych we absolbe on Cireci

Este tipo de dato es utilizado cuando se quiere brindar claridad y legibilidad a los programas. Dado que a veces, existen objetos que son más fáciles de comprender si poseen sus propios nombres.

La declaración de los enumerativos, en Pascal, se logra encerrando entre paréntesis los valores a enumerar: throback mast property and should the

```
Type NombreTipo = (Valor1, Valor2, ... ValorN);
Tamo las cariables como las constantes, atriban bientifications, car les relicances en mon de la nucleida real
```

Donde Valor1, ValorN, son constantes simbólicas que, en Pascal, no pueden repetirse en las defimiciones de otros enumerativos. (1) o pap no contest de mos establedes establementes actuales y autists samultoq

Las *operaciones* válidas sobre este tipo son:

MITAL ORDINAL EX

าร์ง สมรักษาประกาณ

altigates our light

- Asignación: Mediante: = se puede asignar uno de los valores constantes correspondientes a los definidos en el tipo perteneciente. La la la son cale a comana com atenza comanado mbra sand se bantha e a cinta da cent ni oldi shorin ozukim le massarotiva bini Johan Johan lob zomratka kol
- Comparación: Igualdad (=) y Desigualdad (<>). adenta al Laborino di alam arrizo antire derineo ad addentira edinfun nos votas cep na lentis ob escona to o viscostra lo

El resultado de estas operaciones es un booleano.

Al publisero numini el ob-

tales its libraria i globo 🖟 ar like was and be obsolish.

No se pueden realizar, en Pascal, operaciones de E/S. edita de a abayezarent de a abia f

TIPO DE DATO SUB-RANGO

Es un tipo de dato ordinal que está formado por una secuencia contigua de valores ascendentes de algún tipo ordinal (tipo base del sub-rango).

flame in access come to interpretation of the fifther our recorder our configure. Manages par necessaria

graph and the sea earned in year. Este tipo de dato es utilizado cuando no se requiere trabajar con todos los valores que un estándar ordinal provee.

La declaración se determina mediante el límite inferior y el superior de la secuencia, separados por dos puntos:

Type NombreTipo = LimiteInferior..LimiteSuperior

Var NombreVariable: LimiteInferior..LimiteSuperior.

Siendo LimiteInferior menor que LimiteSuperior, de lo contrario la declaración es ilegal.

I visifia Albaz alegari bari, Area salba gila zarena atesta burannear ezkilaba da da benegazanlari betarra beneg

Posgo no rampio aindo por la creacha) de caracteres que la varmida confene, siendo esti sa loscima. La esticina Las operaciones válidas para este tipo de dato son las mismas que las de su tipo base.

TIPO DE DATO CONJUNTO, estarrollor testa calicada in capacita con incluido por la compansa de la conferencia del conferencia de la conferencia de la conferencia del conferencia de la conferencia de la conferencia de la conferencia de la conferencia del conferencia del conferencia de la conferencia de la conferencia del confere

Es una colección de datos simples, sin repeticiones (en caso de haberlas se ignoran) ni relación de orden, pertenecientes al mismo tipo ordinal (enteros, caracteres o enumerativos). Siendo su limite, en Pascal, de 256 elementos.

La declaración se logra mediante la forma:

Type NombreTipo = Set of TipoOrdinal

Var NombreVariable: Set of TipoOrdinal

Donde TipoOrdinal es el tipo al que pertenecen los elementos del conjunto.

La construcción se establece definiendo los elementos individuales consecutivamente, encerrados entre corchetes [] y separados por comas:

```
Conjunto := [Elemento1, Elemento2, ... ElementoN].
```

Los elementos del conjunto, pueden ser también variables que pertenezcan al mismo tipo base de este. Además, se puede definir varios elementos consecutivos mediante ..:

```
Conjunto := [ElementoK..ElementoK+I].
```

lights along this experience comes, the first comment to the solutions of

Un conjunto está vacío si fue declarado de las siguientes maneras: opy a az azáran so zanduden e grunden egyípínest pel

- Conjunto := []; {Sin contenido}
- distribute an exercist. O reviewdo de paras ere enciones es es Conjunto := ['Z'..'A']; {Con orden inverso}

Las operaciones válidas sobre este tipo de dato son:

Asignación: Mediante := se puede asignar un valor o el contenido de un conjunto a otro.

Aritméticas: Unión (+), Intersección (*), Diferencia (+) y Pertenencia (In). 4654 fragmente de 107

Tanto la unión, como la intersección y la diferencia retornan otro conjunto. Mientras que pertenencia devuelve un booleano.

inmiso De Comparación: Igualdad (=) y Desigualdad (<>): As una requirement usano instrucción (or mandas lab estadadad anti-

El resultado de estas operaciones es un booleano.

Sobre los conjuntos no es posible realizar operaciones de lectura y de escritura.

TIPO DE DATO STRING

Es una secesión de caracteres que se almacenan en un área contigua de la memoria principal, que puede ser leída y/o escrita.

Posee un tamaño dado por la cantidad de caracteres que la variable contiene, siendo está su longitud. La máxima capacidad, en Pascal, es de 255 caracteres. En memoria, ocupa Longitud+1 caracteres, dado que la posición cero de la misma contiene la Longitud.

En la declaración de un string, se puede especificar el tamaño del mismo, pudiendo ser menor, el contenido real de este al establecido:

Donde Longitud es el número máximo de caracteres que puede contener. Si no se establece una longitud, por defecto, el tamaño del string es la máxima capacidad.

Las operaciones válidas sobre este tipo de dato son: contenta est accompany out la est

- Asignación: Mediante: = se puede asignar un valor o el contenido de un string a otro, truncando los caracteres sobrantes si se excede la capacidad del string receptor.
 - Concatenación: Permite adosar un string a continuación de otro:
 - Encolar: Str1 := Str1 + Str2;

 **Apilar: Str1 := Str2 + Str1;

 **Apilar: Str1 := Str2 + Str1;

 **Apilar: Str1 := Str2 + Str1;

 **The str1 is a str1 in the str2 in
 - De Comparac<mark>ión: Igualdad (=), D</mark>esigualdad (<>), Menor que (<), Menor o Igual que (<=), Mayor que (>) y Mayor o Igual que (>=).

La Igualdad retorna verdadero, si ambos strings poseen la misma Longitud y contienen los mismos caracteres en el mismo orden, caso contrario devuelve falso. La Desigualdad funciona de la misma manera que la Igualdad. Las demás operaciones, comparan carácter por carácter, según su valor ordinal ASCII, hasta determinar un resultado. El resultado de estas operaciones es un booleano.

tion and maken talked softer asteriors are the son

rato a començo em sis cibiastros to o mino na sacreto diserção en comendado Ariena, espeia e

aqv%

3 75 7 7

CAPITULO IV: PROCEDIMIENTOS, FUNCIONES Y PARÁMETROS

1) MÓDULOS

Conjunto de instrucciones y de definiciones de datos que realiza una tarea lógica.

Pueden realizar las mismas acciones que un programa. La diferencia está en que el módulo se ejecuta cada vez que se lo invoca desde el programa principal o desde otro; dándole el control durante su ejecución y retornándolo al finalizar.

Cada módulo debe tener un identificador único, con el cual será llamado, y puede poseer una serie de parámetros asociados a el, con los que se enviará y/o recibirá información.

El *alcance* de un módulo está regido por el contexto (visibilidad en el mismo nivel) de este y, a su vez, depende de la ubicación dentro de la definición de las rutinas.

Un módulo conoce todo lo que anteriormente fue definido a él y todo lo que está definido en su interior. El programa principal no puede invocar a los módulos que estén dentro de otros módulos.

2) PROCEDIMIENTOS

Es un conjunto de instrucciones que realizan una tarea específica y, como resultado, pueden retornar o no, uno o más valores como resultado.

Su invocación en el programa, u otro módulo, es una simple línea: el nombre del módulo.

```
Procedure Indentificador (Var Parametro1: TipoBase; Parametro2: TipoBase);
Const
    {Constantes Locales}

Type
    {Tipos Locales}

Var
    {Variables Locales}

Begin
    {Instrucciones}
End;
```

Los Procedimientos pueden recibir tanto parámetro por referencia (Var) como por valor.

3) FUNCIONES

Es un conjunto de instrucciones que realizan una tarea específica y, como resultado, retorna siempre una única respuesta.

Su invocación en el programa, u otro módulo, es a través de una asignación y/o comparación. También, el resultado de una función puede ser impreso.

```
Function Indentificador(Parametro1, Parametro2: TipoBase): ValorRetornado;
Const
{Constantes Locales}
```

CONTRACTOR PROPERTY OF A SOURCE PARKS STATEMENT

Type

{Tipos Locales}

D MORELLES

Var

(Variables Locales) poly(representation) of respective places and respective properties of the propert

on **Begin** at the set is obtained to set as a constant in a constant the set is a constant in the set is a constant in the set is a constant to the constant of the constant in the constant of the constant is a constant to the constant in the constant in

Las funciones solamente deben recibir parámetro por valor, aunque pueden recibir por referencia en el caso de los arreglos, para no violar el concepto de función: retornar un único valor. Y su valor debe ser siempre consumido.

til akomer tid met nombe ombre, de por et somerne er sabilidad en er monne needs de este yer somere myende dom Beherde nedere de la de**tir ein de** des monne.

l en enedado como a todo lo cas americamentes for totando e a y exterío que esta dellando en se en anor el programa principal en produciones anódados que estan carro de care a módalos.

MARKATER MARKET

Es es communición mediade entre que tendician una case especidica y como comindo, enciden acción o no, enco o más subjectos como como como en descripción.

componente bisadimenta a agral cirquia and es combota nato a contangora es ao norsado arte se

🌖 arang katang matang katang kabupatèn 🖜

aqVI.

a.a.V

Contract to the Contract of th

Begin

A MAGAINE SECTION AND SECTION

; bad

k os Procedinamenos parakta pedala, camo presentero par el destar (1, 1, 1, 2, 2, 2, 1, 1, par 1, 1, 1, 1

BELL WALL

tis un communicae mismacchones que neellezaciona ana cultipolítea y. Lomo resumsdo, certan sample una ádica rasturesta.

su încomina un el propontor a orce medide, es a traces de una arquidide y le compandida. Lasinon, el compado de una finición paede ser imprese

Sunction each backet sussent on an analysis of the explication of th

Carporation Research Selection

CAPITULO V: ESTRUCTURAS DE DATOS COMPUESTOS

1) CONCEPTOS BÁSICOS

Una estructura de datos es un conjunto de variables, que pueden no ser de mismo tipo, relacionadas entre sí de diversas formas.

There is a first consequence of the enclosed hardel and the interest

enter the configurations of the region of the contract of the contract of the region o

Construidos a partir de datos simples o estructurados, la rentabilidad de las estructuras compuesta, recae en poder representar elementos del mundo real más complejos que los datos simples.

Estas estructuras pueden clasificarse por dos criterios:

- Datos constructores:
 - Homogéneas: Los datos que componen la estructura son del mismo tipo de la componen la estructura son del mismo tipo de la componen la estructura son del mismo tipo de la componen la estructura son del mismo tipo de la componen la estructura son del mismo tipo de la componen la estructura son del mismo tipo de la componen la co
 - Heterogéneas: Los datos que componen la estructura son de distintos tipo.
- Espacio en memoria:
 - Estáticas: La cantidad de elementos es fija, por lo tanto, la cantidad de memoria que se utiliza no varía a lo largo de la ejecución de un programa.

La desventaja de estás estructuras esta en la cantidad fija de memoria, ya que se puede tanto malgastar el espacio reservado, si sobra, así como también, quedarse sin él para más datos. Sin embargo, la ventaja recae en el acceso directo a los datos, ya que las posiciones de los mismos se pueden calcular como desplazamientos de la posición inicial de la estructura, dado que se encuentran fisicamente consecutivos.

tanz ombody vidi vers sobre sobility vynakywyna se i

The second system of the group of the contract of the contract

Para la cantidad de elementos puede variar a lo largo de la ejecución del programa, por lo tanto, la cantidad de memoria también.

La ventaja de estás estructuras esta en la cantidad variable de memoria, ya que se reserva la necesaria para cada elemento a medida que el algoritmo lo requiere. Sin embargo, las desventajas recaen en el acceso secuencial a los datos (debido a que las posiciones de memoria de estos no se encuentran físicamente consecutivas, sino lógicamente) y en el tiempo extra de procesamiento que el lenguaje y el sistema operativo realizan (debido a la reserva y la liberación de memoria).

2) REGISTROS

Son estructuras de datos, estáticas y heterogéneas, que permiten agrupar diferentes tipos de datos con una conexión lógica y en una única estructura. Los datos contenidos, son conocidos como campos y cada uno de ellos posee un identificador único dentro de la misma.

La declaración de los registros es de la forma:

```
Type

NombreTipo = Record

Campo1: TipoCampo1;

Campo2: TipoCampo2;

CampoN: TipoCampoN;

End;
```

```
NombreVariable = Record
                  Campol: TipoCampol;
              Campo2: TipoCampo2;
                                   FOR CHARLES MAKE (SH
              CampoN: TipoCampoN;
```

-man de la sunt enlanguaria como o **Endi**cia ele carteden con establica en como nombre por lo espet. Os contrates su d

Donde Campo1, Campo2... CampoN son los identificadores de los campos y TipoCampo1, TipoCampo2... TipoCampoN los tipos correspondientes. ser ou audificientes el sentamentes o schenic soul sicultate el constitució commenced the commenced of the commenced products and the commenced of the commenced products and the commenced products and the commenced products and the commenced products are commenced to the commenced products and the commenced products are commenced by the commenced products and the commenced products are commenced by the commenced products are commenced by

El acceso a los campos de un registro es a través del nombre del registro, seguido por el punto y el nombre del camworse, not was even their early or equipment on t po:

NombreRegistro.NombreCampo

Esto es lo que se denomina calificar el campo. También se pude acceder a los campos a través de la sentencia With:

```
With NombreRegistro dos is all the anthony an evaluation to be added to be washington
           begin
                Campo1 := Valor1;
                Readln (Campo2);
                Writeln (Campo1 + Campo2);
n time cole in end ky zalo informatic de balantan en adale el carte este esta v<mark>iliza e de babilita</mark> en Colo esta el c
                                                              ware speak the objections sit at the organical
```

El uso de With recae en evitar la tediosa tarea de la calificación de campos, ya que nombrando el registro una sola vez, se puede acceder a ellos como variables ordinarias, por motivos como a ser activo se substativos e como esta activo se substativos e como esta activo se substativos estas esta with the attended a first time to provide the principle of the principle o parameters do la profesiona appeal de la estactoria da la que se caracitam discomerno acipavecament Las operaciones válidas sobre este tipo de dato son:

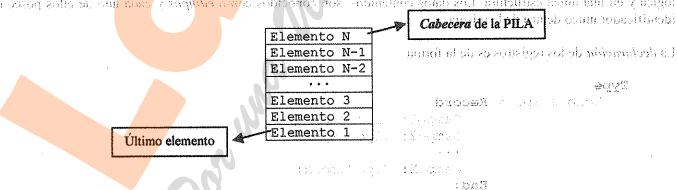
Asignación: Mediante := se puede asignar un valor a un campo o el contenido entero de un registro a otro, siempre y cuando pertenezcan al mismo tipo. ra camidad de premera anseidee.

Las demás operaciones están restringidas a los campos del registro y dependen del tipo de los mismos. Por lo tanto, las comparaciones como la aritmética entre registros completos están prohibidas: 46 n obratirado mano entre residential authorial de contraction of conclosed an one object between the process residential fishers control consecutives, vine ligit amenders, and tiangual ever de procesamente que el longraphy el sintereincopses do miscopidi el y expercisal cobeloby nexibos a discopromi 3) PILAS

Es una colección ordenada, dinámica y homogénea de elementos, que retorna los datos almacenados en el orden inverso en el que fueron ingresados (LIFO: Last In, First Out - Último en Ingresar, Primero en Salir).

建设料 [金属] 建筑 [1]

A partir de una posición en memoria, los datos se van depositando sucesivamente adelante del primer elemento, convirtiéndose estos en la cabecera o tope de la pila, estamba papa en abgonica de la cabacera o tope de la pila, estamba papa en abgonica de la cabacera o tope de la pila, estamba papa en acceptado de la cabacera o tope de la pila, estamba papa en acceptado de la cabacera o tope de la pila, estamba papa en acceptado de la cabacera o tope de la pila, estamba papa en acceptado de la cabacera o tope de la cabacera o on second solls at somethic gramma constraints now additional new addition of the alless person in the gramma and the person of the second sec Cabecera de la PILATI de comunicación de la



La declaración, en Seudo-Pascal, de una pila es de la forma:

Var NombreVariable: TipoPila; Type NombreTipo = TipoPila;

Kitomina beng opina di

Donde TipoPila es alguno de los siguientes valores posibles:

regionalith eans alghusurgian 🦠 🔑 each tach a group a scripter ang retarrantizargeti. 🚶 🧽

- Stack Integer
- Stack Real constraint in the second of th
- Stack Char
- Stack String (De 255 caracteres)
- Stack Rec TipoRegistro (TipoRegistro es el identificador del tipo de registro deseado).

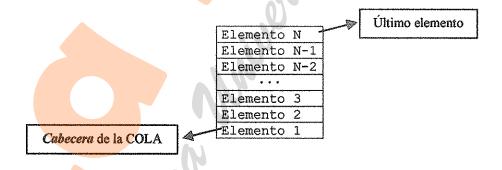
Las operaciones válidas sobre este tipo de datos son:

- St_Create (NombrePila): Procedimiento que activa o crea a NombrePila asignándole una dirección de inicio en la memoria:
- **St_Push (NombrePila) Elemento): Procedimiento que agrega (apila) a Elemento en Nombre
 Pila, poniéndolo en el tope de la misma e incrementando en uno su tamaño actual abasa de la misma en un constanta de la misma en la misma
- St_Pop (NombrePila, Elemento): Procedimiento que quita (desapila) a Elemento de NombrePila, sacándolo del tope de la misma y decrementando en uno su tamaño actual. NombrePila debe contener un elemento como mínimo para que esta operación sea válida.
- St_Empty (Nombre Pila): Función que retorna un valor booleano: True si Nombre Pila está vacía o False en caso contrario.
- Nombre Pila. Función que retorna el número (un entero) de elementos almacenados en Nombre Pila.
- Pila, sin quitarlo de la misma.

4) COLAS

Es una colección ordenada, dinámica y homogénea de elementos, que retorna los datos almacenados en el mismo orden en el que fueron ingresados (FIFO: First In, First Out – Primero en Entrar, Primero en Salir).

A partir de una posición en memoria, los datos se van depositando sucesivamente detrás del último elemento.



La declaración, en Seudo-Pascal, de una cola es de la forma:

Type Nombre Tipo = TipoCola; Var Nombre Variable: TipoCola;

Donde TipoCola es alguno de los siguientes valores posibles:

Queue Integer

estring angles continuencing out absorbed to the first to the first

างสายสายสาย 255 ยร์เรียกการ

- Queue Real
- Queue Char
- Queue String (De 255 caracteres)
- Queue Rec TipoRegistro (TipoRegistro es el identificador del tipo de registro deseado).

Las operaciones válidas sobre este tipo de datos son:

- Q_Create (NombreCola): Procedimiento que activa o crea a NombreCola asignándole una dirección de inicio en la memoria.
- e Q Push (NombreCola, Elemento): Procedimiento que agrega (encola) a Elemento en NombreCola, poniéndolo al final de la misma e incrementando en uno su tamaño actual.
- Q_Pop (NombreCola, Elemento): Procedimiento que quita (desencola) a Elemento de NombreCola, sacándolo del tope de la misma y decrementando en uno su tamaño actual. NombreCola debe contener un elemento como mínimo para que esta operación sea válida.
- Palse en caso contrario. Si dia any managa and a managa ang managa

Peachery on a companional

- NombreCola.: Función que retorna el número (un entero) de elementos almacenados en
- Q Top (Nombre Cola): Función que retorna el contenido del elemento que está en el tope de Nombre Cola, sin quitarlo de la misma.
 - Q_Bottom (NombreCola): Función que retorna el contenido del elemento que está al final de Nombre-Cola, sin quitarlo de la misma.

光表(10)3体

Es des concated catameir destablica comprovada a elementos ara contacios mes mans de acidente con ante contracto en sun composito de la mismo de el que el que el que el presente en sun el composito en el que el presente en sun el composito en el que el presente en sun el composito en el que el presente en sun el composito en el que el q

communicación de entral analysistes an characteristical de entral de arrandor concederante de arrandor. Con arrandor



to declaration of the explosion of the colors of the formation of

The Toler of Miner Segriff

Domestic or and the control of the second state of the second of the sec

and they want to the control of the con-

roughous remover profit rots of already conficts and but of

ger come (typplage), et a glochliege et legelei (eg legelegg) et et legelegge en reg

distinctions are eased the betterning of county

CAPITULO VI: DATOS COMPUESTO INDEXADOS - ARREGLOS

机制组制存

1) ARREGLOS

Es una colección ordenada, indexada, estática y homogénea de elementos, que retorna cualquier dato, indicándole la posición que ocupa el mismo dentro de la estructura.

Los arreglos son de la forma:

Type NombreTipo = Array [LimiteInferior..LimiteSuperior] of TipoDato;

Var NombreVariable: Array [LimiteInferior..LimiteSuperior] of TipoDato;

Donde LimiteInferior debe ser menor o igual que LimiteSuperior, y juntos conforman el Índice, que debe ser ordinal.

Pueden existir varios indices o dimensiones en un arreglo dependiendo del lenguaje y de la memoria que ocupe la estructura.

je Alfongaleri i je krjedi k

into the compliance and control solution of the control control to the control of the control of

VECTORES

Es un tipo de dato arreglo de una dimensión o indice

| | | t a dyindeliin d |
|-------------------------|-------------------|------------------|
| Elem. 1 Elem. 2 Elem. 3 | Elem. N-1 Elem. N | |
| | . 10 777 211 201 | |

Cada celda del vector, ocupa un lugar en forma consecutiva a partir de la dirección inicial de memoria asignada. La forma de acceder a las celdas, es utilizando el nombre de la variable junto con los [] y la posición dentro del vector:

Vector[Posición]

Donde Posición debe ser un valor literal o una variable correspondiente con el tipo de Índice, así como también, el valor literal o variable que se le pudiera asignar a la celda.

La definición de un vector es de la forma:

Type NombreTipo = Array [Indice] of TipoDato;

Var NombreVariable: Array [Índice] of TipoDato;

Donde Indice es el rango de valores que se toman para direccionar a las celdas y, además, establece la cantidad de estas que contendrá (LimiteSuperior - LimiteInferior + 1).

Las operaciones válidas sobre este tipo de dato, son las correspondientes al tipo de dato al que pertenece, pero solo se realizan sobre las celdas y no sobre el vector en sí. La única excepción es la asignación entre vectores, y ambos deben tener la misma dimensión, el mismo Índice y TipoDato.

integras as profession de parametris por referencia. De este meder no se displicar y se ectin oralgastar commens, en ancia dicitiones en carato de recaned ex frame displicações (compessores) de monte esta en se be desem

MATRICES

REPROPERTY - ROUNT SHOW OF STATEMENT AND REPROPERTY OF A STATEMENT OF STATEMENT OF

Es un tipo de dato arreglo de dos dimensiones o índices.

| Elem. 1, 1 Elem. 1, 2 ··· Elem. 1, N-1 Elem. | 1 |
|--|--------|
| Elem. 2, 1 Elem. 2, 2 · · · Elem. 2, N-1 Elem. | 2, N |
| Elem. 3, 1 Elem. 3, 2 · · · Elem. 3, N-1 Elem. | |
| Elem. 4, 1 Elem. 4, 2 · · · Elem. 4, N-1 Elem. | 4, N |
| | • |
| Elem. M-1, 1 Elem. M-1, 2 · · · Elem. M-1, N-1 Elem. | M-1, N |
| Elem. M, 1 Elem. M, 2 · · · Elem. M, N-1 Elem. | M, N - |

Filas

EQUENTIAL H

Columnas

Cada celda de la matriz, ocupa un lugar en forma consecutiva a partir de la dirección inicial de memoria asignada. La forma de acceder a las celdas, es utilizando el nombre de la variable junto con los [] y las posiciones dentro de la matriz:

Matriz[Posición1, Posición2]

beth artists of a prostal as the selection

Donde Posición1 y Posición2 deben ser valores literales o variables correspondientes con el tipo de Índice, así como también, el valor literal o variable que se le pudiera asignar a la celda.

La definición de una matriz es de la forma.

sudati , ucertites, usan ku

Type NombreTipo = Array [Índice1, Índice2] of TipoDato;

Var NombreVariable: Array [Índice1, Índice2] of TipoDato;

Donde Índice1 e Índice2 son el rango de valores que se toman para direccionar a las celdas y, además, establecen la cantidad de estas que contendrá:

(LimiteSuperior1 - LimiteInferior1 + 1) * (LimiteSuperior2 - LimiteInferior2 + Limit

Índice1 referencia a las filas de la matriz, mientras que Índice2 a las columnas. Ambos, ordinales, pueden no pertenecer al mismo TipoDato o incluso poseer longitudes diversas. Por lo tanto, no se deben alterar el orden de los índices: no es lo mismo [1, 2] que [2, 1]. El primero direcciona a la celda de la fila 1, columna 1; mientras que el segundo, a la celda de fila 2, columna 1.

Las operaciones válidas sobre este tipo de dato, son las correspondientes al tipo de dato al que pertenece, pero solo se realizan sobre las celdas y no sobre la matriz en sí. La única excepción es la asignación entre matrices, y ambas deben tener las mismas dimensiones, los mismos índices (Índicel e Índicel) y TipoDato.

is all the combined right and some addition constructions of t

okunyas mornis kija iaz ius prapoje to bados par vientilos sati padas armitero se

2) ARREGLOS COMO PARÁMETROS

Si bien se recomienda el uso de parámetros por valor, en los casos en los que no se modifica la estructura, con los arreglos es preferible el uso de parámetros por referencia. De este modo, no se duplican y se evita malgastar recursos, ya que la eficiencia en cuanto al uso de memoria y tiempo disminuye. La única precaución que se debe tomar es de no alterarlos si no se lo desea.

CAPITULO VII: RECURSIVIDAD

A STORM BETARIE, STEER BARRETAND PARAGET STEEL A SE STEER STEER IN

1) RECURSIVIDAD

verternifregigen best infe infreiningten ich näcklichen. Ib

Es una técnica de resolución de problemas en donde el resultado de estos, se obtiene a través de módulos que se invocan a sí mismos.

Cuando un módulo se invoca a sí mismo, reduce al problema P a una instancia de sí, P₁. Al volverse a invocar, esta instancia se reduce a otra, P₂, y después de otra invocación se reduce a P₃, y así sucesivamente. A medida que cada instancia se establece, se va alcanzando el caso degenerado, el cual corta la recursividad por ser una instancia distinta a las demás (ya que no vuelve a invocar al módulo), y además, es la solución de P. El caso degenerado debe

figurar siempre, de manera implícita o explícita, de lo contrario no tendrían fin las invocaciones.

El'ejémplo más claro de entender, és el de la búsqueda de una persona en una guía: Proviso el confidera de la búsqueda de una persona en una guía: Proviso el confidera de la confideración del la confideración de la confideración d

Si (Guía contiene una sola página) entonces patron entonces BuscarPersonaEnPagina (Caso Degenerado)

Sino

Existent than language the correlated throughts the effectivities

Inicio

egicos secion complividir Guíasa la mitad openio la mitado de Guía (Persona está encla de <mark>amitado de Guía) centonces</mark> de combo Buscar (1º mitad de Guía, Persona)

provide as journed si**Sino** among in moraldory on standardory of springership speed to set a presume (A. (all astro to munico ship**Buscard.(2)^a mitadadesGuía, «Persona).** Its sont sontingle to se trust-continues of actorists to alternate and actorists actorists and actorists actorists and actorists actorists and actorists actorists actorists actorists actorists and actorists acto

Aunque esta metodología es similar a la delidiseño Top-Down, su diferencia recae en que el problema Pires de la misma naturaleza que el problema original P, pero más simple.

La desventaja de la recursividad recae en ineficiencia que conlleva las sucesivas invocaciones: mayor tiempo de ejecución y mayor gasto de memoria. La constanta de la constanta de constan

2) EJECUCIÓN DE UN PROGRAMA Y LA PILA DE EJECUCIÓN

La pila de activación es una estructura de datos que se comporta de la siguiente manera; Veneral de la siguiente de la siguien

- Posee un tope y un fondo.
- Cada elemento es código más datos.
- Los elementos solo pueden ser sacados del tope uno por uno.

राजिन्द्रात्रकुरकोर्दे एक सक्तावरकोप्तानुके 👯

son algorithms and product the area of the area of the contractions of the second contractions of

La pila se crea cuando comienza la ejecución de un programa, en donde cada elemento que se agrega a la misma, ocupará un segmento de la memoria que recibe el nombre de *frame*. Cuando el programa micia, el primer *frame* se crea conteniendo dos clases de datos: los declarados en sección de declaración del programa principal y una variable que contiene al *Contador de Programa*.

Cuando se invoca a un procedimiento o una función, se asigna un nuevo frame para ese módulo que permanecerá en la pila hasta que finalice se ejecución. Logrado esto, se libera la memoria reservada para ese frame, y el programa o módulo que había realizado la invocación retoma el control.

the saturance of the form of months to recognize the recognized of the saturance of the sat

TO DEPOSIT HEALT STATE

群人们的逻辑的 医海绵性 经补偿债 的复数法国

CAPITULO VIII: CONCEPTO DE EFICIENCIA

1) Análisis de eficiencia de un algoritmo

Los algoritmos pueden ser analizados bajo dos aspectos de eficiencia: de analidades de asercia de analizados bajo dos aspectos de eficiencia: de analidades de asercia de analizados bajo dos aspectos de eficiencia:

Tiempo de Ejecución: Se consideran más eficientes aquellos algoritmos que cumplan con las especificastand described del problema en el menor tiempo posible. Habita de la la companya de la companya del companya de la companya de la companya del companya de la companya del la companya de aban om dano e o Anomateromes los e o Por part de estre propresa nota do caración y 🔑 de estrado o especial de

Para poder medir este tiempo de ejecución, se tienen en cuenta dos operaciones elementales: las comparaciones y las asignaciones. Y puede suceder, por lo tanto, que un algoritmo varié su tiempo de ejecución dependiendo de los datos ingresados, ya que pueden realizar más o menos operaciones.

Esto establece la existencia del mejor caso (menor tiempo), del caso promedio (tiempo promedio), y del pero caso (mayor tiempo). En general, se tiene mayor interés en el comportamiento del algoritmo en el peor caso o en el caso promedio, que en el mejor caso, dado que importa que el programa sea más rápido en los casos adversos que en los favorables. Por lo tanto, no importa cuan largo sea un algoritmo, sino que cantidad de operaciones realice. a acquain.

Existen dos formas de medir el tiempo de ejecución:

Análisis Teórico: Consiste en calcular el número de comparaciones y asignaciones que requiere un algoritmo, sin importar cuál es la máquina en el que se ejecuta.

Al someter a las diversas alternativas de solución de un problema al mismo análisis teórico, se obtiene cual es el algoritmo más eficiente y, teóricamente, más rápido. Dado que permite estimar el orden del tiempo de respuesta, que servirá como comparación relativa, sin depender de los datos de experimentación.

Análisis Empírico: Consiste en someter a un algoritmo a diversos juegos de datos a fin de calcular su tiempo de respuesta. Strate average. and more experiences for the passing many and the

Este análisis empírico depende de la máquina en el que se realiza y en mayor parte de los datos que se emplean, ya que pueden favorecer a un algoritmo, pudiendo no ser este el caso general, con lo cual las conclusiones del análisis puede ser erróneas.

as people in a construction and analysis for a construction of the

modely a product a substitute of the confidence of the superior of

State of the state of the state of the

on the section and the section through the contract of the con

Uso de la memoria: Se consideran eficientes aquellos algoritmos que utilicen las estructuras de datos adecuadas de manera de minimizar la memoria ocupada. El énfasis está puesto en el volumen de información a manejar en memoria.

2) Algoritmos de Búsqueda

Son algoritmos que permiten ubicar información en una colección de datos.

Existen dos estilos de busqueda:

p<mark>usqueux:</mark> The Survivaria of Change Continued, of conferencies odered was accommon af also cautages an insquee The Survivaria for a form to the continued of conferencies odered was accommon af also cautages an insquee Búsqueda Lineal o Secuencial: Consiste en recorrer el vector en forma secuencial desde el inicio, e ir analizando uno a uno los elementos contenidos hasta encontrar el dato deseado o hasta llegar al final de la misma.

Esta búsqueda es utilizada cuando una estructura no posee o se le conoce un orden.

Búsqueda Binaria o Dicotómica: Consiste en comparar el elemento buscado con el que se encuentra en la mitad de el vector: si coinciden, la búsqueda termina, si no, se analiza si el elemento solicitado se encuentra en la "primera mitad" (si es menor que el del medio) o en la "segunda mitad" (si es mayor que el del medio). Después de desechar una de las dos mitades, se vuelve a repetir el mismo procedimiento sobre la mitad elegida, y de no hallarse el elemento buscado, se continua dividiendo hasta que la porción de estructura sea tan pequeña que no puede almacenar datos en ella o hasta haber localizar el elemento buscado.

Esta búsqueda se utiliza cuando el vector posee un orden.

3) Algoritmos de Ordenación

Son algoritmos por los cuales, un grupo de elementos pueden ser ordenados?

Se dividen en dos categorías:

- Ordenación Ineficiente: Son aquellos que son no recursivos.
- Selección: Consiste en buscar, de a uno, los elementos men<mark>ores (o mayores d</mark>ependiendo del criterio de ordenación) y colocarlos al comienzo del vector, en posiciones consecutivas.

Se busca el primer menor y se lo intercambia con el con la primera posición, luego se busca el segundo menor y se lo intercambia con la segunda posición; así sucesivamente hasta que el elemento n-1 ocupe la penúltima posición. El último elemento, el mayor (o menor), queda ubicado en la última posición.

Este método realiza N-1 pasadas para ordenar la estructura. Este método realiza nel pasadas para ordenar la estructura.

```
Procedure Selection (Var Vector: Array of Integer; NroEle: Integer);
Var
   X, Y, Z, Pos: Integer;
   Aux: Integer;
Begin
   Z := NroEle - 1;
                                                           ন হেচ্ছের
   For X := 1 to Z do
      begin
         Pos := X; ...
         For Y := X+1 to NroEle do
             If Vector[Y] < Vector[Pos] then</pre>
                Pos := Y;
         Aux := Vector[Pos];
         Vector[Pos] := Vector[X];
         Vector(X) := Aux;
      end:
End;
```

© Burbujeo o Intercambio: Consiste en comparar los elementos adyacentes del vector y moverlos si estos subjectes describados. Ada trib me vicamo to apare están desordenados. Ada trib me vicamo to apare de aparesado abrandos en contra en como traballo y de como en discourse degan languación de analysica en colonia. Establica en colonia en colonia en colonia.

Al igual que la selección, el burbujeo realiza N-1 pasadas para ordenar la estructura. Pero solo muevo los elementos a una distancia de una posición. Por lo tanto, después de una pasada, el elemento mayor (o empresada menor dependiendo del criterio de ordenación) es arrastrado al final del vector. En la segunda pasada, el segundo mayor es arrastrado a la penúltima posición; así sucesivamente hasta que el elemento menor (o frame procumayor) se encuentre en la primera posición.

```
was in the bodie meaning of any ingression are not the state of contained with a footbooms. The contained by the property of Integer; NroEle: Integer);

Var

X, Y, Z, Pos: Integer;

Aux: Integer;

Begin

For X := NroEle downto 2 do 40 and 2 for the contained by the contained by
```

```
who cannot sades and begin as common a stack a stable of a common sobject of the modes of the account of the stable of the stabl
```

common On their and red hader. After

ya están ordenados consiste en insertar los elementos del vector en la posición correcta, entre los elementos que ya están ordenados consiste en insertar los elementos del vector en la posición correcta, entre los elementos que ya están ordenados.

Se comienza del segundo elemento, suponiendo que el primero ya está ordenado. Compara ambos elementos, si no están ordenados, los ordena. Luego toma el tercero y lo inserta en la posición correcta con respecto a los dos primeros. Así sucesivamente hasta terminar con el último elemento.

Si un elemento está ubicado correctamente no lo mueve de lugar.

```
Procedure Insercion (Var Vector: Array of Integer; NroEle: Integer);
Var
   X, Y: Integer;
   Aux: Integer;
Begin
   For X := 2 to NroEle do
      begin
         Aux := Vector[X];
         Y := X - 1;
         While (Y > 0) and (Vector[Y] > Aux) do
                Vector[Y+1] := Vector[Y];
                Y := Y - 1;
             end:
         Vector[Y+1] := Aux;
      end;
                                                           . into
End;
```

- Ordenación Eficiente: Son algoritmos recursivos.
 A son anticipante de la la composition de la composition della composition della
- Ordenación por Mezcla (Sorting by Merging): Consiste en dividir varias veces al vector en partes iguales y ordenar cada una de ellas, para luego mezclarlas entre sí y obtener de esta manera, la estament obse ou tructura ordenada, todo o para estament de esta manera, la estament obse ou tructura ordenada, todo o para estament de esta manera, la estament obse ou tructura ordenada, todo o para estament de esta manera, la estament obse ou tructura ordenada, todo o para estament de esta manera, la estament de esta manera, la estament obse ou tructura ordenada, todo o para estament de esta manera, la estament de esta

BORDERSON TO PROPERTY OF MARKET PROPERTY OF THE OWN OF THE

Es un algoritmo recursivo que trabaja dividiendo al vector por la mitad, en forma iterativa, hasta que cada subvector contenga un único elemento; en la primer invocación del procedimiento se pasa al vector entero, junto con los limites del mismo; en la segunda invocación, se pasan la primera mitad (n/2 elementos) y los limites de esta; en la tercera invocación, se pasan la primera mitad del vector original (n/4 elementos) y los limites... así sucesivamente. Finalizadas las divisiones, es la mezcla de los subvectores (merge) la que ordena a la estructura.

```
Type

TVector = Array [1..N] of Integer;
```

```
Procedure Mezclar (Var Vector: TVector; Inicio, Medio, Fin: Integer);
                                  Var
ls us refined refine to the control of the control 
                                         VectorAux: TVector;
                                                                                       tark mercia Algebia Dae Elara Cameracca dork
                                                             absorbed autorese of positions with at somethic module to our
                                  Begin
                                         X := 0;
on le oraș Marak nog Inicio2d:=dMedio k+u1; rescue alegalem șep o despot ordrande na 🥢
nizof / zoidangrain a While a (Inicio Kandedio) and (Inicio 24 Kan Fin) a do your any ing
                                                begin
                                                       X := X + 1;
omes (edseros si a sum elso ca<mark>lse, copia el elemento, más, chico de los dos subve</mark>ctores
observed as ence i since in the vector [Inicio]
                                                                                                                                         then with the record
                                                                                                < Vector[Incio2]</pre>
Tripedivision bi extent so to mustiere a
                                                                             ម៉ាត្ 🗯 ១០ឆ្នាំ១០១២៧ ១៨៤១២៦៤ ខណ្ឌៈ
to supsequent character as resulter (VectorAux [X]) := Vector [Intoio] ; poly count out it.
of the completion accomplete and chaptering of Inicio. = Inicio. +r1; the residence commercial the control
                                                              end
                                                       else
varonunt y sanoyou solusunais ab udassin
                                                                                                    := Vector[Inicio2];
                                                                     VectorAux[X]
-wo to not augul constant to not extensive inicio2 := Inicio2 := Inicio2 + 1,000 at one q to true, to the
                                                end;
emploid the servery of Schrop party by the company of the servery and company to servery of the
                                         While Inicio <= Medio do {Mientras haya elementos en el 1°, subvector}
                                                begin
                                                       X := X + 1;
                                                       VectorAux[X] := Vector[Inicio];
                                                       Inicio := Inicio + 1;
                                         While Inicio2 <= Fin do {Mientras haya elementos en el 2º subvector}
                                                begin
                                                       X := X + 1;
                                                       VectorAux[X] := Vector[Inicio2];
                                                       Inicio2 := Inicio2 + 1;
                                                end;
                                                                                                                                                    SALAKVI
                                                                                                                                            dived
                                         Vector := VectorAux;
                               End;
                                  Procedure OrdenarPorMezcla (Var Vector: TVector; Inicio, Fin:
                                                                                                   Integer);
                                  Var
                                        Media: Integer;
                                                                                                                               Charles
                                  Begin
                                         If Inicio < Fin then
                            OF
                                                       Medio := (Inicion Fin) div 2;
                                                       OrdenarPorMezcla (Vector, Inicio, Medio);
                                                       OrdenarPorMezcla (Vector, Medio + 1, Fin);
                                                       Mezclar (Vector, Inicio, Medio, Fin);
                                                end:
                                  End;
                                                                                                                               , bno
```

Este método de ordenación es más eficiente que aquellos que no son recursivos, pero posee un costo adicional relacionado la estructura auxiliar: si VectorAux es muy grande, las sucesivas invocaciones de Ordenar consumirían mucha memoria. Por lo tanto, el uso de este método está recomenda-

do para vectores de poco tamaño.

Ordenación Rápida (QuickSort): Consiste en dividir varias veces al vector en partes iguales (en el caso promedio), en donde los elementos de la primera mitad son menores que los de la segunda, y que al ordenar cada una de ellas, se obtiene la estructura ordenada.

Es un algoritmo recursivo que trabaja de manera similar que la Ordenación por Mezcla, pero al no utilizar una estructura auxiliar, resulta ser más eficiente ya que realiza menos intercambios. Consta de dos etapas:

División del vector por la "mitad": Se toma al último elemento (el que está más a la derecha) como punto de división, el cual se "saca" de la estructura dejando un hueco. Luego se busca, comenzando desde la izquierda (Izq), un elemento mayor que el punto de división. Si existe, se lo transfiere a dicho hueco. Después, comenzando desde la derecha (Der), se busca un elemento menor que el punto de división, que, si existe, será colocado en el hueco dejado por el elemento transferido de la izquierda.

De esta manera, se repite sucesivamente la búsqueda y reubicación de elementos mayores y menores que el punto de división, hasta que las marcas Izqy Der se encuentren en el mismo lugar (en el cual se ubicará el punto de división).

Ordenación de los subvectores: Se invoca tanto para la primera mitad como para la segunda el procedimiento de división, anteriormente descripto.

```
Procedure Dividir (Var Vector: Array of Integer; Var Medio:
                   Integer; Izq, Der: Integer);
Var
                                             ed tille
  Aux: Integer;
                                           17000I
Begin
  Aux := Vector[Ultimo];
  While Izq < Der do
     begin
         {Se mueve la marca Izq hacia la derecha}
         While (Izq < Der) and (Vector[Izq] <= Aux) do
            Izq := Izq + 1;
         If Izq < Der then
            begin
               Vector[Der] := Vector[Izq];
               Der := Der - 1;
            end;
         {Se mueve la marca Der hacia la izquierda}
         While (Izq < Der) and (Vector[Der] >= Aux) do
            Der := Der - 1;
         If Izq < Der then
            begin
               Vector[Izq] := Vector[Der];
                                                than
               Izq := Izq + 1;
            end;
      end;
```

enas an open, com experimenta paragravitana, one mande en el extendore de cimbre de l'eniment de l'eniment de l enimentale en en el comme de l'orde en l'entre en l'entre en mande en l'eniment de commé de l'eniment de l'eniment

PARTY 177 SEVERE EDUARDO JOSÉ CASEY

Vector [Medio] /: ⇒ 'Aux; 2002 \ 723 2002 \ 7000 \ 000 \ 000 \ 2002 \ End;

weary talgray to 1 \$

Procedure OrdenacionRapida (Var Vector: Array of Integer;

viels much a commendation of Fin: o.Integer); is some placing a mass is

Var

Medio: Integer;

2.1 \$ ky \$ 24 K

Begin

an serongih shrommaigi II. Inicio. ≤ Fingthen shrab gamed compande ab sajabah gamed sajabah gamed ab sajabah gamed sajabah game

- Supply ton aqueins and power in animal addition que sus nodes sole idealing prensity que has tos
 climates Posace an pelacipio y fin. pero signific el neorde el los discusors elementos contenidos a neorde del
 pantero priocipal y en acesado santido (larcio d'int.)
- Circultores, Son parecidas a)as listas simplés, puro differen en que al ser circulares no poscen un fin, proque al otunto elemento aparecidad primero o principo de la tista. Pescen un medo especial one indica ence os el micha de la misma por se puedes, nor estanten el cantenio elemento de la misma, nor estanten el se puedes, nor el malar el cantenio elemento de la cantenio norte.
- Indefendable Kirkersky, kan aquellas que pascan dos enhaces (debido a que sus nodes contienen dos pantes ros que los vincara) y paceden sen recesaridas en dos sentidos Chricio-Cia. Fin-Inicio) ya que las dos panteros principales Poscan na principio y un fin neo dendo los dates paralen ser accentidos dosde condiguens de instancios companios compris dados recupias anapies.
- Metrics Anno. Son squally specified no nois critices, debide a que sas nodes conficuent unios paramagas tos sentions of no nodes en distinct industrial per les conferments en elementes de distinct industriales per libraries en la configue de la configue.

is a sergebbert, precise some complete at the exist of the process of a company and a division and os makes

- e i per el el el minero de come en cometante de esta cometante con en en en entre de cimentales de clipte en e Perconnecte de la cometante de
- ing dividity and the native of the first of the control of the con

region of present and successful extension of the contraction of the c

SALVE

かから たからなな様 とした。

Thomas I in the section in

1.53

Fans accios sus panents in agentos, por le tanto diferensei ar malos sel cationdas. Es sa anni liste es ma Lentifice en mante o que escribe el metalec de proceso finicial e cul referil, e e satora de discrebia de inicio de de misse um

tros editorio sussi trifos accidos o compenoresian vando grappo ed l

CAPÍTULO IX: DATOS COMPUESTOS ENLAZADOS: LISTAS Y ÁRBOLES

1) Punteros

ocedura studenci kangula . Yar sangula kangula Array of uning sang

Es una variable numérica que almacena la dirección en memoria de un dato.

2) Listas

Es una colección dinámica de elementos homogéneos, donde los mismos pueden aparecer fisicamente dispersos en memoria, pero manteniendo un orden lógico interno. Son estructuras lineales, por ló tanto, cada elemento de la misma, menos el primero y el último, posee un antecesor y un sucesor.

Existen 4 tipos básicos de listas:

- Simples: Son aquellas que poseen un único enlace, debido a que sus nodos solo tienen un puntero que los vincula. Poseen un principio y fin, pero siempre se accede a los diversos elementos contenidos a través del puntero principal y en un solo sentido (Inicio-Fin).
- Circulares: Son parecidas a las listas simples, pero difieren en que al ser circulares no poseen un fin, porque el último elemento apunta siempre al primero o principio de la lista. Poseen un nodo especial que indica cual es el inicio de las mismas, pero se pueden acceder a cualquier elemento desde cualquier nodo.
- Doblemente Enlazadas: Son aquellas que poseen dos enlaces (debido a que sus nodos contienen dos punteros que los vincula) y pueden ser recorridas en dos sentidos (Inicio-Fin, Fin-Inicio) ya que hay dos punteros principales. Poseen un principio y un fin (en donde los datos pueden ser accedidos desde cualquiera de los dos) y ocupan más memoria que las listas simples.
- Multienlaces: Son aquellas que poseen dos o más enlaces, debido a que sus nodos contienen varios punteros que los vinculan. Estos enlaces son índices por los cuales se puede acceder a los nodos en distintos órdenes. Poseen varios principios y/o fines.

La creación de una lista simple es a través de la unión de nodos (los elementos) que se dividen en dos partes:

- Dato: Es la información que se maneja y puede estar constituida por un conjunto de elementos de algún tipo previamente definido.
- Identificador: Es una variable puntero cuyo contenido es la dirección del próximo elemento de la estructura.

El nodo de una lista simple, en Pascal, esta formado de la siguiente manera:

Estos nodos son punteros de registros, por lo tanto, deben ser creados antes de ser utilizados. En sí, una lista es en realidad un puntero que recibe el nombre de puntero inicial o principal, y contiene la dirección de inicio de la misma.

Las operaciones más comunes que se realizan sobre listas simples son:

day a single 📈

April 1 April 1

- Inicialización: Se asigna el valor Nil al puntero principal.
- Inserción: Se busca la posición correcta en la que irá el nuevo elemento, que será creado antes de insertarlo. Luego se enlaza los punteros de los elementos adyacentes a este.

Si se inserta al inicio de la lista, el puntero principal apuntará al nuevo y este al que ahora es el segundo. Si se inserta al final, el puntero del último elemento apuntará al nuevo, convirtiéndose este en el último, con un identificador al siguiente en Nil. Y si se inserta entre los extremos de la lista, el nuevo puntero apuntará al elemento Actual y el identificador al siguiente de Anterior, apuntará al nuevo.

```
Procedure Insertar (Var ListaNumeros: Lista; K: Integer);
            Nuevo, Actual, Anterior: Lista;
                                                     083
         Begin
            Actual := ListaNumeros;
            While (Actual <> Nil) and (K > Actual^.Numero) do
ali di svenevo de dise beseto que en concestor el cientemo de comb o desserio pel llaguel di fiedi de d
                   Anterior := Actual;
                                                                              0000000
                   Actual := Actual^.Siquiente;
               end;
                                                                          049 C
            New (Nuevo);
            If Actual = ListaNumeros
               ListaNumeros := Nuevo
            else
ed ong ng sandibab Anterior^:Siquientes:=\Nuevo;bar was sap as mentorested of osa seb salways as b
```

The course and the control of the co

l'a desenuaja que confloran las bases es el acceso secuences de el decreace de las mismas. Ve ens**tena** entranacion

• Borrado: Se busca la posición correcta del elemento que se borrará. Se enlazan los punteros adyacentes a este y el elemento se borra haciendo un dispose del mismo.

Si se borra el elemento del inicio de la lista, el puntero principal apuntará al siguiente elemento del primero. Si se borra el del final, el identificador al siguiente del penúltimo elemento será nulo (Nil), convirtiéndose este en el último. Y si se borra un elemento entre los extremos de la lista, el identificador al siguiente de Anterior a puntará al siguiente del elemento Actual.

```
Procedure Borrar (Var ListaNumeros: Lista; K: Integer);
      Var
                                            Shari in stong articles y thing our many copyrights and to
          Actual, Anterior: Lista;
      Begin
                                                   t dat apikiromene å avskag kaladali kel kijaglj/
          Actual := ListaNumeros;
before A. Fro While (Actual o <> Nil) (and b (Ke <> Actual A. Numero) adomesa, oben sign)
             begin
             d v&Anterior :=: Actual: ( ora class stand) zataventa op fabril moucepoz stat)
Alege adomentos
                 Actual := Actual^.Siguiente;
              end;
                Cheb. Charles one so expensely. Needs travily heads that however, is, of members de warms.
          If Actual^.Numero = K then non beadour gomes around no neon dong obour be 1
             begin
                 If Actual = ListaNumeros then
                     ListaNumeros := Actual^.Siguiente;
```

else

le Impain ording h. ... Noch the Lagran of Ministerior Anterior ... Siguiente := Actual ... Siguiente;

Altares no some control process of process of the control of the some control of the some control of the some of t

Is some a suice of the state of purpose property and the control of the administrative of the segment. State of purpose of the control of the

• Recorrido: Se comienza leyendo el puntero inicial, siempre y cuando no este vacía la estructura, el cual contiene el identificador del tercero y este, a su vez, el del cuarto. Así sucesivamente hasta leer un nodo que posea un identificador al siguiente elemento nulo (Nil), el cual es el último.

```
While (Actual <> Nil) do
    Actual := Actual^.Siguiente;
```

Búsqueda: Se recorre la lista hasta que se encuentre el elemento deseado o hasta que se llegue al final de la misma.

```
While (Actual <> Nil) and (Actual^.Numero <> N) do
begin
   Anterior := Actual;
   Actual := Actual^.Siguiente;
end;
```

Las ventajas del uso de listas recaen en que son más fáciles de manipular que otras estructuras estáticas, ya que las operaciones de borrado y de inserción son más simples de implementar, porque no realizan un corrimiento masivo de elementos, como hacen los vectores. Además, al ser una estructura dinámica solo ocupan la memoria necesaria.

La desventaja que conllevan las listas es el acceso secuencial a los elementos de las mismas, ya que no se encuentran fisicamente consecutivos.

a komissa zpr. k albima, koj misalim pr. mismant sv sag. carbinos (ab etimbras invakcija) dikarić (k) (abimbol. **3) Árboles**analina lab partykla nia obimboli in analina (ab expekta nia obimboli) in analinada (ab y abib

3) Árboles omaine ist veryik na obasisal med a masaulo e 7 sto

Es una estructura, dinámica, homogénea y recursivas, que satisface tres propiedades: a memoris la mandia se estado de la companya de actual de la mandia de la companya de actual de la mandia de la companya de actual de la mandia de la companya de actual de la companya del la companya de la companya del la companya de la companya

- A Cada elemento del árbol se puede relacionar con cero o más elementos, los cuales se llaman hijos.
- Si el árbol no está vacío, existe un único elemento que no posee un padre y recibe el nombre de raíz.
- Todo elemento posee un padre, excluyendo a la raíz, y es descendiente de esta.

Además, los árboles poseen 6 características más:

alled

- Cada nodo que no posea descendientes, recibe el nombre de hoja y se encuentra siempre al final del árbol.
- Cada secuencia lineal de elementos, donde cada uno es el padre del próximo ítem de la secuencia, recibe el nombre de camino.
- Cada camino que se extienda desde la raíz hasta una hoja, recibe el nombre de rama.
- Cada nodo junto con sus descendientes, recibe el nombre de subárbol.
- Cada nodo posee un nivel que es el número de ítems que hay en el único camino que lo une a la raíz.

:Dag

Ъľ

Fredrik

nagent

La altura de un árbol es el máximo entre los niveles de los nodos de este.

<u>ÁBOLES BINARIOS</u>

Son aquellos árboles donde cada nodo no tiene más de dos hijos. A lo sumo, poseen un hijo izquierdo y un hijo derecho. For Children Service and Archael of Committee Control 1997

elegated leafing our concepts of sample grant approximation of

Son también conocidos como Árboles Binarios de Orden, dado que cada nodo de este es mayor que los elementos de su subárbol izquierdo (el que tiene como raíz a su hijo izquierdo) y menor o igual que los de su subárbol derecho (el que tiene como raíz a su hijo derecho).

La creación de un árbol binario es a través de la unión de nodos (los elementos) que se dividen en tres partes:

- Dato: Es la información que se maneja y puede estar constituida por un conjunto de elementos de algún tipo previamente definido.
- Identificador Izquierdo: Es una variable puntero cuyo contenido es la dirección del próximo elemento izquierdo de la estructura. Wilder Secretors of School sky to the man Pader i life-filler
- Identificador Derecho: Es una variable puntero cuyo contenido es la dirección del próximo elemento derecho de la estructura.

El nodo de un árbol, en Pascal, esta formado de la siguiente manera:

```
Type
                                                      Arbol = ^Hoja;
                                                      Hoja = Record
                                                                                                                                                                                                                                               Numero: Integer;
                                                                                                                                                                                                                                               HijoIzq, HijoDer: Arbol;
                                                                                                                                                                                        End;
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              April will be the consideration for the first three this will be the consideration of the con
```

Estos nodos son punteros de registros, por lo tanto, deben ser creados antes de ser utilizados. En sí, un árbol es en realidad un puntero (la raíz) que contiene la dirección de inicio del mismo.

Las operaciones más comunes que se realizan sobre árboles binarios son:

- Inicialización: Se asigna el valor Nil a la raíz.
- Inserción: Cada elemento nuevo se incorpora al árbol como una nueva hoja, preservando solo el orden del mismo sin importar su altura. Se busca primero el lugar en donde se agregará la nueva hoja y, luego se acomodan los punteros. steranolo lis vistantato appartamenta a servicinario di foriga te meneori di digenganti

```
arrang that that the
                                                        Procedure Insertar (Var Ar: Arbol; N: Integer);
                                                                                                                                                                                                       menga otromoto la la comeditado ebuesade el objetem o especial. H
is secrem a vikari (Ar = Nil) then
is secrem a vikari (Ar = Nil) essimens in 1441 social sus character carrected in 1451 sap adsaugance de las
-myonada marana begin
                                                                                                                                                                                                                               Northica was the first protein the common of the common processing and the common processing and
                                                                                                                     New (Ar);
                                                                                                                     Ar^{N}ro := N;
                                                                                                                     Ar^.HijoIzg := Nil;
                                                                                                                     Ar'. HijoDer := Nil;
                                                                                                  end
                                                                            else
                                                                                                                    (N < Ar^.Numero) then
                                                                                                                      Insertar (Ar^.HijoIzg, N)
                                                                                                  else
```

:45318

Insertant (ArthijoDer, o'N) is set veno emblem to so bette un ob wanter. I

ADDIES HEX SERVE

```
Recorrido: Existen tres formas de recorrer un árbol binario:
```

Sem aquelles minites devide ando no dene ques de dos hijos. A le samo, nosoen en hijo izquierdo y un hijo acre-

```
En Orden: Se recorre el árbol de la forma Hijo-Padre-Hijo.
```

actions of Procedure RecorrerArbolInOrden (Art. Arbol). At which the control of t

begin

Write (Ar^.Numero); {Padre}

eng, angle disconnection [Arch HijoDer]; who is decreased to end:

End;

 Karadio des Lances et estado para con concertado es la dacecea del presente elemente de culcado de la especiara

Pre-Orden: Se recorre el árbol de la forma Padre-Hijo-Hijo.

```
Procedure RecorrerArbolPreOrden (Ar: Arbol);

Begin

If (Ar <> Nil) then

begin

Write (Ar^.Numero); {Padre}

ImprimirArbolInOrden (Ar^.HijoIzq);

ImprimirArbolInOrden (Ar^.HijoDer);

end;

End;
```

Pos-Orden: Se recorre el árbol de la forma Hijo-Hijo-Padre.

```
Procedure RecorrerArbolPosOrden (Ar: Arbol);
Begin

If (Ar <> Nil) then

begin

ImprimirArbolInOrden (Ar^.HijoIzq);
ImprimirArbolInOrden (Ar^.HijoIzq);
Write (Ar^.Numero); {Padre}

end;
```

End;

lele polició lo elles observament a med a como una como casar la morpo da como el como de maria any morbi. La Como os ociones mied acomo el lingorar se obselo ao angal la cromisa acondició condició estación de casara.

Búsqueda: Se recorre el árbol de manera pre-orden hasta encontrar el elemento deseado o hasta que se llegue a final de la rama.

Es una variante de la búsqueda dicotómica: si el elemento apuntado es el deseado, la búsqueda termina. Sino, se comprueba que el elemento buscado sea menor que el apuntado, y de ser así, se vuelve a invocar al proceso pasando como parámetro a HijoIzq como árbol. En caso de que sea mayor, se invoca a dicho proceso pasando a HijoDer.

```
Function Pertenencia (Ar: Arbol; N: Integer): Boolean;

Begin

If (Ar = Nil) then

Pertenencia := False

else

If (Ar^.Numero = N) then

Pertenencia := True
```

```
else

If (N > Ar^.Numero) then

Pertenencia := Pertenencia (Ar^.HijoDer, N)

else

Pertenencia := Pertenencia (Ar^.HijoIzq, N);

End;
```

Altura del árbol: Es el valor máximo de altura que posee el árbol (máximo nivel + 1) y se calcula contando al nodo actual más el máximo de hijos entre HijoIzq e HijoDer.

```
Function Mayor (A, B: Integer): Integer;

Begin

If (A >= B) then

Mayor := A

else

Mayor := B;

End;

Function Altura (Ar: Arbol): Integer;

Begin

Altura := 0

else

Altura := 1 + Mayor(Altura (Ar^.HijoIzq), Altura (Ar^.HijoDer))

End;
```

Cantidad de Nodos: Es el número de nodos que posee el árbol y se calcula contando al nodo actual más los descendientes de sus hijos.

```
Function Numero_Nodos (Ar: Arbol): Integer;

Begin

If (Ar = Nil) then

Numero_Nodos := 0

else

Numero_Nodos := 1 + (Numero_Nodos (Ar^.HijoIzq) +

Numero_Nodos (Ar^.HijoDer));

End;
```

- Cantidad entre valores: Es el número de nodos que posee el árbol que cumplen con una condición dada de valores:
 - Iguales a un valor:

```
Function Cantidad N (Ar: Arbol; N: Integer): Integer;

Begin

If (Ar = Nil) then

Cantidad N := 0

else

IF (Ar^.Numero = N) then

Cantidad N := 1 + Cantidad N (Ar^.HijoDer, N))

else

If (N > Ar^.Numero) then

Cantidad N := Cantidad N (Ar^.HijoDer, N)

else

Cantidad N := Cantidad N (Ar^.HijoIzq, N);

End;
```

gale

```
Mayores que un valor:
                                          Denia Common
         Function Cantidad Mayor N (Ar: Arbol; Max: Integer): Integer;
         Begin
            If (Ar = Nil) then
               Cantidad Mayor N := 0
                                                                            (bull
            else
               If (Ar^.Numero > Max) then
                  Cantidad Mayor N := 1 + (Cantidad Mayor N (Ar^.HijoDer, Max) +
chandany choles we get in hear official false is sered "Cantidad Mayor Na (Ar . Hijotzq, Max))
                             else
                  Cantidad Mayor N := Cantidad Mayor N (Ar^.HijoDer, Max);
         End;
                                                                       สอมสาหมั่น
                                                                          rapos
                                                          1955 (193
                                                                          93
         Menores que un valor:
                                                                        60 4 443
         Function Cantidad Menor N (Ar: Arbol; Min: Integer): Integer;
         Begin
            If (Ar = Nil) then
               Cantidad Menor N := 0
            else
               If (Ar^.Numero < Min) then
                   Cantidad Menor N := 1 + (Cantidad Menor N (Ar^.HijoIzq, Min) +
                                            Cantidad Menor N (Ar^.HijoDer, Min))
               else
                  Cantidad Menor N := Cantidad Menor N (Ar . HijoIzq, Min);
         End;
regular. The Entre valores: many alumba or community responsible about the continue is a
         Function Cantidad Entre (Ar: Arbol; Max, Min: Integer): Integer;
         Begin
                                                                       Punckage
            If (Ar = Nil) then
                Cantidad_Entre := 0
            else
                If (Ar^.Numero >= Min) and (Ar^.Numero <= Max) then
                   Cantidad Entre := 1 + (Cantidad Entre (Ar^.HijoIzq, Max, Min) +
                                          Cantidad Entre (Ar^.HijoDer, Max, Min))
                else
                   If (Ar^.Numero > Max) then
                      Cantidad Entre := Cantidad Entre (Ar^.HijoIzq, Max, Min)
                      Cantidad Entre := Cantidad Entre (Ar^.HijoDer, Max, Min);
olegiais waters
                                                                             isumole, r
                                                                 Commenters of the Experience
                                                                          តារ៉ាប្រទា
                                             aeda
                                                                    683.9
                                                                           1,6003
```

tan anggang èngandin car

QMB 286 VI 6 200

CAPÍTULO XII: INTRODUCCIÓN AL CONCEPO DE ARCHIVOS

Es una estructura, dinámica y homogénea, de datos guardados (o colección de registros) en un dispositivo de almacenamiento secundo: cenamiento secundario.

Los archivos se dividen en dos categorias: dende la categoria de la categoria Físicos: Son aquellos que están almacenados en la memoria secundaria (discos rígidos, flexibles, etc).

Son los archivos creados por el sistema operativo. Por lo tanto, es este el encargado de realizar todas aque-llas operación de los datos en el disco, cantidad de espacio a utillas operaciones relacionadas con la creación y ubicación de los datos en el disco, cantidad de espacio a utilizar, tipos lizar, tipos de accesos permitidos, etc.

Lógicos: Son aquellos que se utilizan para referenciar a los archivos físicos.

En realidad son nombres lógicos a través de los cuales se interactuarán con los verdaderos archivos almace-nados en al contra y reciben información desde y hacia los archivos físicos. realidad son nombres lógicos a través de los cuales se interactual an actual de los archivos físicos.

nados en disco. Los programas solo envían y reciben información desde y hacia los archivos físicos.

La declaración de un archivo, en Pascal, es de la forma:

```
Type obsession madegrees a capable at nobase;
                                                {Tipados}
     NombreTipo = File of TipoBase;
                                                (Sin Tipo)
     NombreTipo2 = File;
                                                        क्षात्र क्षां कार्याच्या । एक क्षांत्रभाव समिति क्षांक्ष्मार
```

NombreVariable: File of TipoBase; Nombrevariable2: File;

Donde File es el indicador de archivo.

Los archivos pueden ser:

HOR HADOWN WATER HOP AND BERNOOM Tipados: Son aquellos que poseen un tipo (integer, real, string, char, registros).

्रेस तथा निर्देश रहिते हैं असे तहे का कोमपुष्ट हैं । विस्तार स्ट्री Colocando of más el tipo, se establece a que tipo base perteneceran los registros del archivo, lo cua Que sean más fáciles de usar, ya que no se necesita realizar ninguna conversión sobre ellos.

No se establece a que tipo pertenecerán los registros del archivo, por lo tanto, es el programador Sin Tipo: Son aquellos que no poseen un tipo. gado de hacer la conversiones necesarias para manipularlos. ne vitas vivas que communantes vanienses en el mentiones an en especiales).

as operaciones básicas sobre archivos son:

Vinculación: Consiste en relacionar al archivo lógico con el físico.

Assign (NombreLogico, NombreFisico);

Apertura: Realiza la apertura de los archivos indicando el tipo de operación que se va a realizar

```
(NombreLogico); {Abre/el archivé existente para l'ectura y escritura}
Reset
                                                ZO MAZZAR EDUARDO JOSÉ CASEY
Re Write (NombreLogico); {Abre el archivo por primera vez, solo para escritura.}
```

Dorade Rewrite crea el archivo o lo sobreescribe borrando todo el contenido, si existiera. En el caso de Doende Reset, debe existir el archivo para poder abrirlo minante somb ob anatomorali anticia de la caso de

Claresura: Cierra el archivo, colocando en fin del mismo al final, y vacia el buffer, para guardar los últimos Tegistros Hassando.

dum Lust dessa fil mediana ensenta sentina en manda en manda sentinga. En el manda en el mand

Close (Nonvignoyico),
when a college of the analysis of the second state of the second nd a Charles, and management of an entering at his remaining and analysis of the second and the second and and the second and and the second and analysis of the second analysis of the second analysis of the second and analysis of the second analysis of the second and analysis of the second analysis of the s Lectura: Lee un registro del archivo, siempre y cuando no este vacío, y lo almacena en una variable del mismo tipo, avanzando el puntero del mismo al siguiente registro. (NombreLogico; Variable) of the activities and an extensive subject to the first of the subject to the subject

servide as acceptatives) and have the many acceptance and single forms of any evidence the contraction of th Escritura: Escribe un registro en el archivo avanzando el puntero del mismo al siguiente registro.

, los archivos, visto desde un programa, son punteros que apuntan siempre a un registro contenido en ellos.

a memoria intermedia, entre un archivo y un programa, donde los datos residen provisoriamente hasta ser ala memoria internicula, cuito un alculvo y un programa, donde los datos residen provisoriamente hasta ser al-nados definitivamente en memoria secundaria o donde los datos residen una vez recuperados. Su propósito es

cesos a los registros pueden ser:

lead son. Leadenga santo su as sugarres equi no response palp pollompa negle sale qui Secuencial: Permite acceder a los registros uno tras otro y en el mismo orden físico en el que stán guarda-The adjoin coriston to I Permite acceuer a 103 regionos uno mas ono y en el mismo oruen usico en el que sian guessas servicios de autornar asellara ou sono de ou seu el ensei absolicia saldancemp

Directo: Permite obtener un registro determinado sin necesidad de haber accedido a los predecesses. ada un umad en entreoffendu brês add. Per

do a su organización, un archivo se clasifica en quant many a unadada con a su postables de de Secuencial: Consiste en un conjunto de registros almacenados consecutivamente de manera que paracceder l registro n-ésimo se debe acceder a los n-1 registros anteriores.

Directo: Consiste en un conjunto de registros donde el ordenamiento físico no corresponde con el ojen 16-Directo: Consiste an accedient accedient a su posición dentro del archivo, sin tener que acceder a n-1

cuencial Indizado o Indexado: Utiliza estructuras de datos auxiliares para permitir un acceso pseulo dicuencial insucues o marchivo. Esta tecnica posee un acceso más rápido que el secuencial, pero nesita