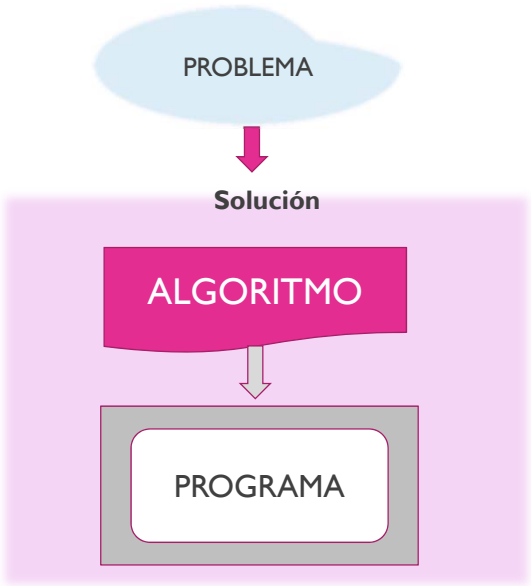
Tecnologías en Educación Matemática

MODULO II

Dpto. de Ciencias e Ingeniería de la Computación
UNIVERSIDAD NACIONAL DEL SUR
Año 2019

Problemas, Algoritmos y Programas



Problemas, Algoritmos y Programas

Un algoritmo es una secuencia de instrucciones, comprensibles para quien tenga que ejecutarlas, que permiten resolver un problema.

Un programa procedural es un algoritmo escrito en un lenguaje de programación imperativo.

Nos interesa escribir algoritmos y programas que terminen en un tiempo finito, resuelvan correctamente el problema propuesto, sean eficientes y estén bien estructurados.

Programas y Lenguajes

Un programa procedural es un algoritmo escrito en un lenguaje de programación imperativo.

```
[*]------------------------ programa.pas ------
program programa;

var n

begin
    w
    r
    w
    readln(num2);

    if num1 > num2 then
        write(num1,' es mayor a ', num2)
    else
        write(num1, 'es menor a ', num2);
end.
```

Un lenguaje de programación es una notación **formal** que puede ser interpretada por una computadora.

Programas y Lenguajes

Los humanos nos expresamos en un lenguaje natural.

En un lenguaje natural una misma palabra
puede estar asociada a distintos significados
y una misma oración
puede interpretarse de dos o más maneras diferentes.

Un lenguaje de programación es una notación formal
porque tiene una **sintaxis estricta** y una **semántica precisa**.

- La sintaxis se refiere a la **forma** de los programas escritos en el lenguaje.
- La semántica se refiere al **significado**.

Programas y Lenguajes

Cada lenguaje de programación tiene sus propias **reglas sintácticas**
que determinan la forma de las instrucciones
y la estructura general del programa.

Objeto.java | Parametros.java

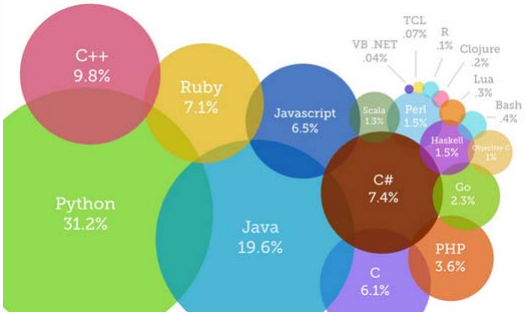
```
public class Parametros {
    public static void main(String[] args) {
        int var1;
        Objeto obj;
        obj = new Objeto(1);
        var1 = 2;
        System.out.println("Valor del objeto= "+obj.variable);
        System.out.println("Valor de la variable ="+var1);
        modifica(var1,obj);
        System.out.println("-Despues de llamar a modifica()-"
        System.out.println("Valor del Objeto= "+obj.variable);
        System.out.println("Valor de la variable= "+var1);

    }
    static void modifica(int vv, Objeto oo){
        vv++;
        oo.variable++;
    }

}
```

```
#include<stdio.h>
void main()
{
    int opcion;
    printf("1. Capital de Argentina\n");
    printf("2. Capital de España\n");
    printf("3. 10000+58000 = ?\n");
    printf("4. Capital de Uruguay\n");
    scanf("%i",&opcion);
    switch(opcion)
    {
    case 1:
        printf("\n\nBuenos Aires");
        break;
    case 2:
        printf("\n\nMadrid");
        break;
    case 3:
        printf("\n\n68000");
        break;
    case 4:
        printf("\n\nMontevideo");
        break;
    default:
        printf("\n\nOpcion erronea. Intenta de nuevo.");
    }
}
```

Programas y Lenguajes

No existe un lenguaje de programación *ideal,*
cada dominio de aplicación impone restricciones diferentes.

Una capacidad fundamental para un profesional de la disciplina es justamente aprender de manera autónoma un nuevo lenguaje y elegir el más adecuado para una aplicación específica.

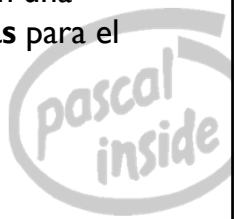


Lenguajes de programación más populares del 2015

El lenguaje Pascal

Pascal es un lenguaje de programación desarrollado por el profesor suizo Niklaus Wirth a finales de los años 60.

El objetivo fue crear un lenguaje que facilitara el aprendizaje de la programación a sus alumnos.
Sin embargo en los años siguientes y hasta los '80, su uso excedió el ámbito académico para convertirse en una **herramienta para el desarrollo de sistemas** para el comercio, la industria, educación, etc.



El lenguaje Pascal

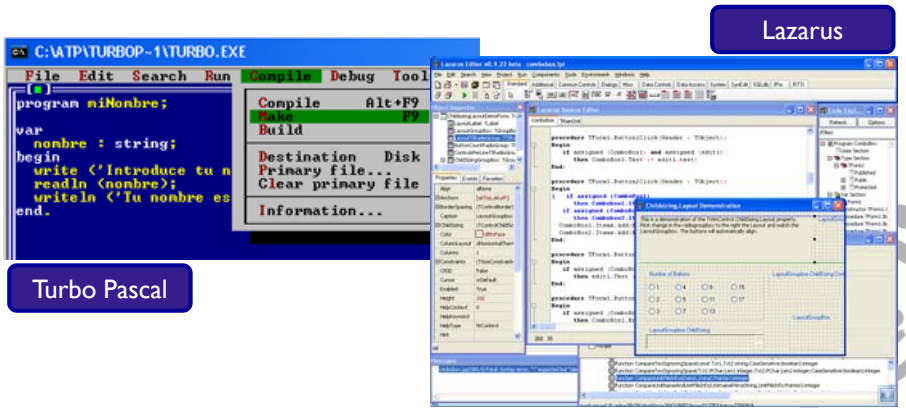
En la actualidad Pascal prácticamente no se utiliza para desarrollar sistemas comerciales pero sí se sigue usando para enseñar a programar.

Las características de Pascal que presentaremos están incluidas también en muchos de los lenguajes de programación que se usan para desarrollar software en la industria.

El concepto de asignación, los tipos elementales y las instrucciones condicionales e iterativas tienen una sintaxis y semántica similar a la de C y Java.

El lenguaje Pascal

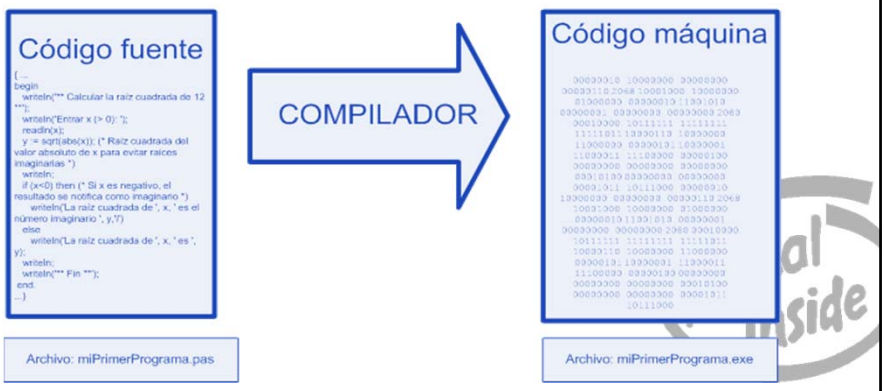
Existen diferentes versiones de Pascal
y cada una propone un **entorno de programación**
que permite **editar**, **compilar** y **ejecutar** programas.



El lenguaje Pascal

La **compilación** permite detectar **errores sintácticos** y algunos **errores semánticos**.

Si el programa no tiene errores sintácticos ni semánticos, el compilador lo traduce a **lenguaje máquina**.



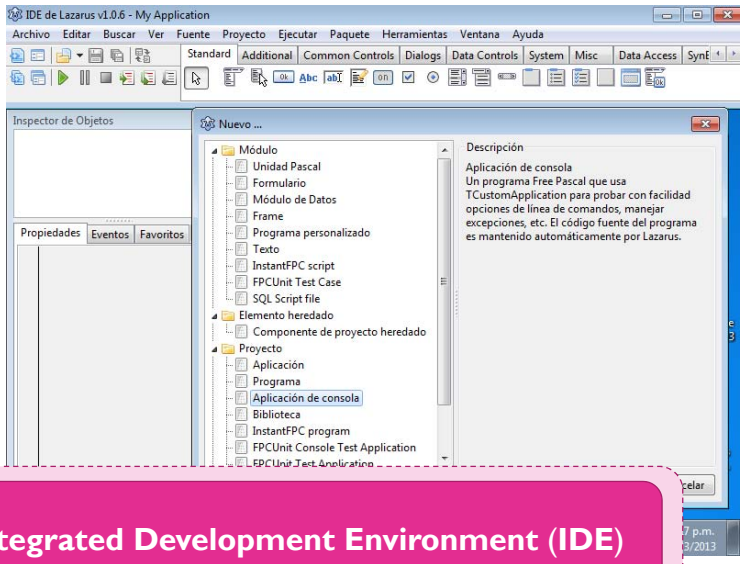
Elementos de Pascal

Problema: Calcular y mostrar el área de un cuadrado a partir de la longitud de lado **ingresada** por el usuario.

```
program areaCuadrado;
var lado, area: integer;
begin
    write('Ingrese la longitud del lado:');
    readln(lado);
    area := lado * lado;
    writeln('El área del cuadrado es ', area);
end.
```

Algoritmo areaCuad
de: lado
ds: área
área ← lado * lado

El entorno de implementación



Integrated Development Environment (IDE)

Elementos de Pascal

Todo programa comienza por la palabra *program* seguido de un nombre.

El ';' es un separador de instrucciones.

```
program areaCuadrado;
var lado, area: integer;
begin
  write('Ingrese la longitud del lado: ');
  readln(lado);
  area := lado * lado;
  writeln('El área del cuadrado es ', area);
end.
```

Elementos de Pascal

Todo programa va a incluir una o más variables.
Cada variable tiene un nombre y un tipo.

```
program areaCuadrado;
var lado, area: integer;
begin
  write('Ingrese la longitud del lado: ');
  readln(lado);
  area := lado * lado;
  writeln('El área del cuadrado es ', area);
end.
```

Elementos de Pascal

Las variables se declaran antes de usarse.
El bloque de declaraciones comienza con la palabra var.

```
program areaCuadrado;
var lado, area: integer;
begin
  write('Ingrese la longitud del lado: ');
  readln(lado);
  area := lado * lado;
  writeln('El área del cuadrado es ', area);
end.
```

Elementos de Pascal

La sección ejecutable de un programa se encierra siempre entre las palabras begin – end y termina con '.'

```
program areaCuadrado;
var lado, area: integer;
begin
  write('Ingrese la longitud del lado: ');
  readln(lado);
  area := lado * lado;
  writeln('El área del cuadrado es ', area);
end.
```

Elementos de Pascal

El programa se comunica con el usuario a través de instrucciones de entrada y salida.

```
program areaCuadrado;
var lado, area: integer;
begin
  write('Ingrese la longitud del lado: ');
  readln(lado);
  area := lado * lado;
  writeln('El área del cuadrado es ', area);
end.
```

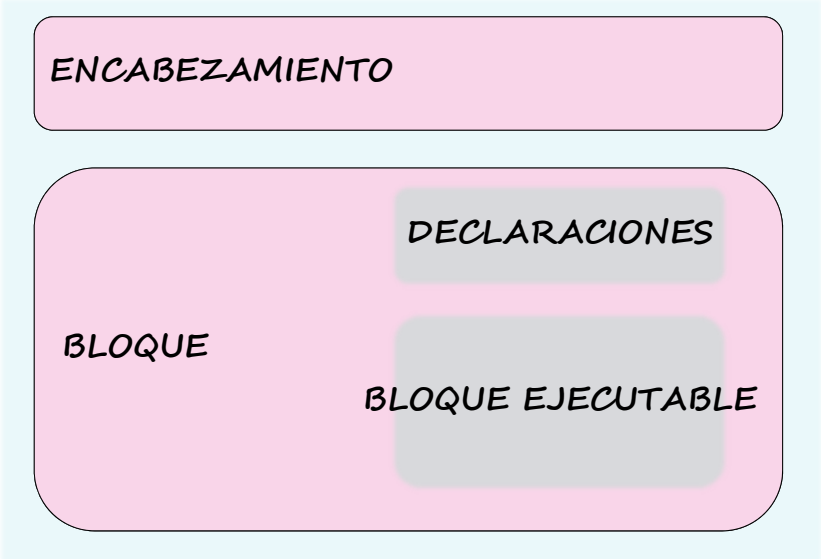
Mostrar y Leer

Elementos de Pascal

La asignación es la instrucción fundamental en un lenguaje de programación imperativo como Pascal.

```
program areaCuadrado;
var lado, area: integer;
begin
  write('Ingrese la longitud del lado: ');
  readln(lado);
  area := lado * lado;
  writeln('El área del cuadrado es ', area);
end.
```

Elementos de Pascal



Elementos de Pascal

Escribir un programa en Pascal que lea el valor del radio de un círculo y calcule y muestre el perímetro y la superficie.

¿Cuáles son los datos de entrada?

¿Alguno de los datos es *constante*?

¿Cuáles son los datos de salida?

```
Algoritmo perYarea
DE: radio
DS: perim, area
perim ← pi * 2 * radio
area ← pi * radio * radio
```

Elementos de Pascal

Escribir un programa en Pascal que lea el valor del radio de un círculo y calcule y muestre el perímetro y la superficie.

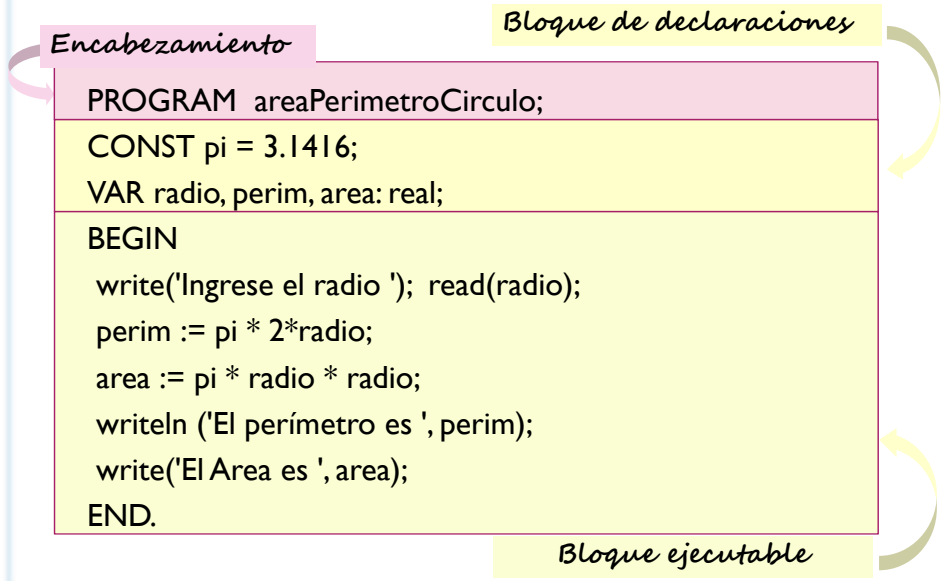
```
PROGRAM  areaPerimetroCirculo;
CONST pi = 3.1416;
VAR radio, perim, area: real;
BEGIN
 write('Ingrese el radio '); read(radio);
 perim := pi * 2*radio;
 area := pi * radio * radio;
 writeln ('El perímetro es ', perim);
 write('El Area es', area);
END.
```

Lee los DATOS DE ENTRADA

Calcula

Muestra los DATOS DE SALIDA

Elementos de Pascal



Elementos de Pascal

Un programa en Pascal incluye tres tipos de identificadores:

- Reservados
- Predefinidos
- Definidos por el programador

Todo identificador comienza con una letra
y puede seguir con otras letras, dígitos o guiones bajos.

Pascal no distingue mayúsculas y minúsculas.

Elementos de Pascal

```
PROGRAM  areaPerimetroCirculo;
CONST pi = 3.1416;
VAR radio,perim,area: real;
BEGIN
 write('Ingrese el radio ');read(radio);
 perim := pi * 2*radio;
 area := pi * radio * radio;
 writeln('El perímetro es ', perim);
 write('El área es', area);
END.
```

Los identificadores **reservados** permiten especificar la estructura del programa y no pueden ser redefinidos.

Elementos de Pascal

```
PROGRAM  areaPerimetroCirculo;
CONST pi = 3.1416;
VAR radio,perim,area: real;
BEGIN
 write('Ingrese el radio ');read(radio);
 perim := pi * 2*radio;
 area := pi * radio * radio;
 writeln('El perímetro es ', perim);
 write('El área es', area);
END.
```

Los identificadores **predefinidos** tienen un significado inicial pero el programador puede cambiarlo.

Elementos de Pascal

```
PROGRAM  areaPerimetroCirculo;
CONST pi = 3.1416;
VAR radio, perim, area: real;
BEGIN
 write('Ingrese el radio ');read(radio);
 perim := pi * 2* radio;
 area := pi * radio * radio;
 writeln('El perímetro es ', perim);
 write('El área es', area);
END.
```

Los identificadores definidos por el programador se declaran en el bloque de declaraciones, excepto el nombre del programa.

Elementos de Pascal

Variables y Constantes

```
CONST pi = 3.1416;
VAR radio, perim, area: real;
```

pi es el nombre de una constante, a lo largo de todo el programa mantiene el valor establecido en la declaración.

El valor asignado a una constante determina además su tipo.

En el ejemplo pi es de tipo real.

Elementos de Pascal

Variables y Constantes

```
CONST pi = 3.1416;
VAR radio, perim, area: real;
```

perímetro area y radio son identificadores de variables.

Cada variable aparece por lo menos dos veces, una en la declaración y otra cuando es usada.

Las variables pueden cambiar de valor dentro de un mismo bloque ejecutable y pueden tomar valores diferentes cada vez que el programa se ejecuta.

Elementos de Pascal

Variables y Constantes

```
CONST pi = 3.1416;
VAR radio, perim, area: real;
```

En la declaración de una variable se establece su tipo de dato.

`real` es un tipo de dato predefinido.

Una variable de tipo `real` puede tomar valores dentro de un subconjunto de los reales y puede aparecer en expresiones aritméticas con operadores predefinidos para reales (+, *, etc).

Elementos de Pascal

```
BEGIN
 write('Ingrese el radio '); read(radio);
 perim := pi * 2 * radio;
 area := pi * radio * radio;
 writeln('El perímetro es ',perim);
 write('El Area es',area);
END.
```

`write` es un procedimiento predefinido.

La primera instrucción `write` muestra un cartel para indicarle al usuario que el programa espera que ingrese la longitud del radio.

Elementos de Pascal

```
BEGIN
 write('Ingrese el radio '); read(radio);
 perim := pi * 2 * radio;
 area := pi * radio * radio;
 writeln('El perímetro es ',perim);
 write('El Area es',area);
END.
```

La segunda y la tercera instrucciones `write` permiten que el programa le muestre al usuario un cartel y el valor que está almacenado en una variable.

Elementos de Pascal

```
BEGIN
 write('Ingrese el radio '); read(radio);
 perim := pi * 2 * radio;
 area := pi * radio * radio;
 writeln('El perímetro es ',perim);
 write('El Area es',area);
END.
```

`read` es un procedimiento predefinido.

La instrucción `read` permite que el usuario ingrese el valor y que el mismo quede almacenado en la variable `radio`.

Elementos de Pascal

```
BEGIN
 write('Ingrese el radio '); read(radio);
 perim := pi * 2 * radio;
 area := pi * radio * radio;
 writeln('El perímetro es ',perim);
 write('El Area es',area);
END.
```

La 1ra instrucción de asignación:

- Evalúa la expresión `pi*2*radio`
- Almacena el valor obtenido en la variable `perim`

La expresión consta de operadores y operandos.

Elementos de Pascal

La estructura del bloque ejecutable,
como la de muchos de los programas que desarrollaremos, es:

- **Leer** los valores de algunos de los **datos de entrada**.
- Procesar y generar datos de salida a partir de datos de entrada.
- **Mostrar** los valores de los **datos de salida**.

Para los mismos valores de datos de entrada un programa generará los mismos valores para los datos de salida.

Elementos de Pascal

Problema: Calcular y mostrar el promedio de tres notas ingresadas por el usuario.

```
program promedioNotas;
{Lee tres notas y muestra el promedio}
var n1, n2, n3: integer;
begin
  write('Ingrese las tres notas');
  read(n1, n2, n3);
  writeln('Promedio ',(n1+n2+n3)/3);
end.
```

Elementos de Pascal

El programa está formado
por una secuencia de tres instrucciones.

1ra: Instrucción de salida, se invoca al procedimiento predefinido *write* para mostrar un cartel.

2da: Instrucción de entrada, se invoca al procedimiento predefinido *readln* para leer los valores de dos variables.

3ra: Instrucción de salida, se invoca al procedimiento predefinido *writeln* para mostrar un cartel y el valor de una expresión.

Elementos de Pascal

```
program promedioNotas;
var n1, n2, n3: integer;
    promedio: real;
begin
  write ('Ingrese las tres notas');
  readln (n1, n2, n3);
  promedio := (n1+n2+n3)/3;
  write ('El promedio de ', n1,  n2, n3);
  writeln(' es ', promedio);
end.
```

La memoria

Dispositivo de almacenamiento de la computadora.

Los valores de las variables
se *almacenan* en *celdas* de memoria.

Una declaración como:

var a, b, c: integer;

Ocupa tres celdas de memoria asociadas a los nombres a, b y c.

Cuando el programa termina las celdas se *liberan* y en la próxima ejecución probablemente se ocupen otras.

Importancia de las trazas

¡FUNDAMENTAL!

Realizar trazas para valores representativos
de los datos de entrada.

- Sirven para depurar un algoritmo
  pero no son una técnica de demostración de correctitud.
- Sólo aseguran que un algoritmo funciona de la manera esperada para los valores de los datos de entrada usados.

La traza de una secuencia de asignaciones

```
PROGRAM abc;
VAR a, b, c: integer;
BEGIN
  a:= 1;    b:= a+1;     c:= b*2;
  a:= b+c;
  writeln (a, ' ', b, ' ', c);
END.
```

a	b	c
1	2	4

La traza de una secuencia de asignaciones

```
PROGRAM abc;
VAR a, b, c: integer;
BEGIN
  a:= 1;    b:= a+1;     c:= b*2;
  a:= b+c;
  writeln (a, ' ', b, ' ', c);
END.
```

a	b	c
6	2	4

La asignación destruye el valor anterior y almacena el nuevo.

La traza de una secuencia de asignaciones

```
PROGRAM abc;
VAR a, b, c: integer;
BEGIN
  a:= 1;     b:= a+1;     c:= b*2;
  a:= a+1;
  writeln (a,' ' ,b,' ' ,c);
END.
```

La misma variable puede aparecer a la izquierda y a la derecha de una asignación.

Primero se computa la expresión y luego se almacena el valor computado.

La traza de una secuencia de asignaciones

```
PROGRAM abc;
VAR a, b, c: integer;
BEGIN
  b:= a+1;
  c:= b*2;
  writeln (a,' ' ,b,' ',c);
END.
```

La variable a aparece en una expresión cuando no ha sido inicializada.

NO vamos a "usar" una variable que no está inicializada.

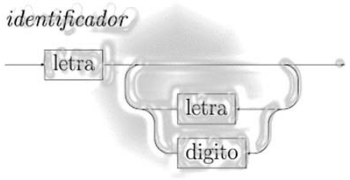
Elementos de Pascal

Identificadores definidos por el programador

El programador puede definir nuevos identificadores.

Cada identificador definido por el programador, excepto el nombre del programa, aparece al menos 2 veces: en la declaración y en el bloque ejecutable.

Pascal no es sensible a mayúsculas y minúsculas.



45

Elementos de Pascal

Palabras reservadas

Las palabras reservadas
tienen un significado especial en Pascal
y el programador sólo puede usarlas con ese significado.

program	if	of	file
uses	then	for	function
label	else	to	procedure
var	until	array	goto
const	repeat	div	not
begin	while	mod	or
end	do	with	and


QUE TENGAS
UN
BONITO DIA


Aburrido??

Elementos de Pascal

Problema: a partir de los valores de los catetos de un triángulo rectángulo, calcular y mostrar la hipotenusa.

```
program hipotenusa;
var c1, c2, h: real;
begin
    write('Escriba los valores de los catetos');
    readln(c1, c2);
    h:=sqrt(c1*c1+c2*c2);
    writeln('La hipotenusa es ',  h);
end.
```


Elementos de Pascal

El programa está formado
por una secuencia de cuatro instrucciones.

1ra: instrucción de salida, se invoca al procedimiento predefinido *write* para mostrar un cartel.

2da: instrucción de entrada, se invoca al procedimiento predefinido *readln* para leer un valor.

3ra: instrucción de asignación, computa un valor y lo almacena.

4ta: instrucción de salida, se invoca al procedimiento predefinido *writeln* para mostrar un cartel y un valor.

Elementos de Pascal

Identificadores Reservados

program var begin end

Identificadores Predefinidos

real readln write sqrt

Identificadores definidos por el programador

hipotenusa c1 c2 h

Símbolos

; , : ‘ := + * ( ) .

Elementos de Pascal

```
program hipotenusa;
var c1,c2,h : real;
begin
    write('Escriba los valores de los catetos');
    readln(c1,c2);
    h:=sqrt(c1*c1+c2*c2);
    writeln('La hipotenusa es ',h);
end.
```

Todo programa comienza por la palabra *program* seguido de un nombre.

El ; es un separador de instrucciones.

Elementos de Pascal

```
program hipotenusa;
var c1, c2, h: real;
begin
    write('Escriba los valores de los catetos');
    readln(c1,c2);
    h:=sqrt(c1*c1+c2*c2);
    writeln('La hipotenusa es ',h);
end.
```

Todo programa va a incluir una o más variables.
Cada variable tiene un nombre y un tipo.
Las variables se declaran antes de usarse.

Elementos de Pascal

```
program hipotenusa;
var c1, c2, h: real;
begin
    write('Escriba los valores de los catetos');
    readln(c1,c2);
    h:=sqrt(c1*c1+c2*c2);
    writeln('La hipotenusa es ',  h);
end.
```

La sección ejecutable de un programa se encierra entre las palabras *begin* – *end* y termina con '.'

Elementos de Pascal

La instrucción de asignación:

```
h:=sqrt(c1*c2+c2*c2);
```

Computa la expresión sqrt(c1*c2+c2*c2) y luego se almacena el valor en la locación de memoria ligada a la variable h.

El tipo de la expresión es compatible con el tipo de la variable h.

Elementos de Pascal

```
program hipotenusa;
{Lee los catetos y calcula la hipotenusa }
var c1,c2 : real;
begin
    write('Escriba los valores de los catetos');
    readln(c1,c2);
    writeln('La hipotenusa es ', sqrt(sqr(c1)+sqr(c2));
end.
```

La expresión puede computarse directamente en la instrucción de salida.

Tecnologías en Educación Matemática

FIN MODULO 11

Dpto. de Ciencias e Ingeniería de la Computación
UNIVERSIDAD NACIONAL DEL SUR
Año 2019