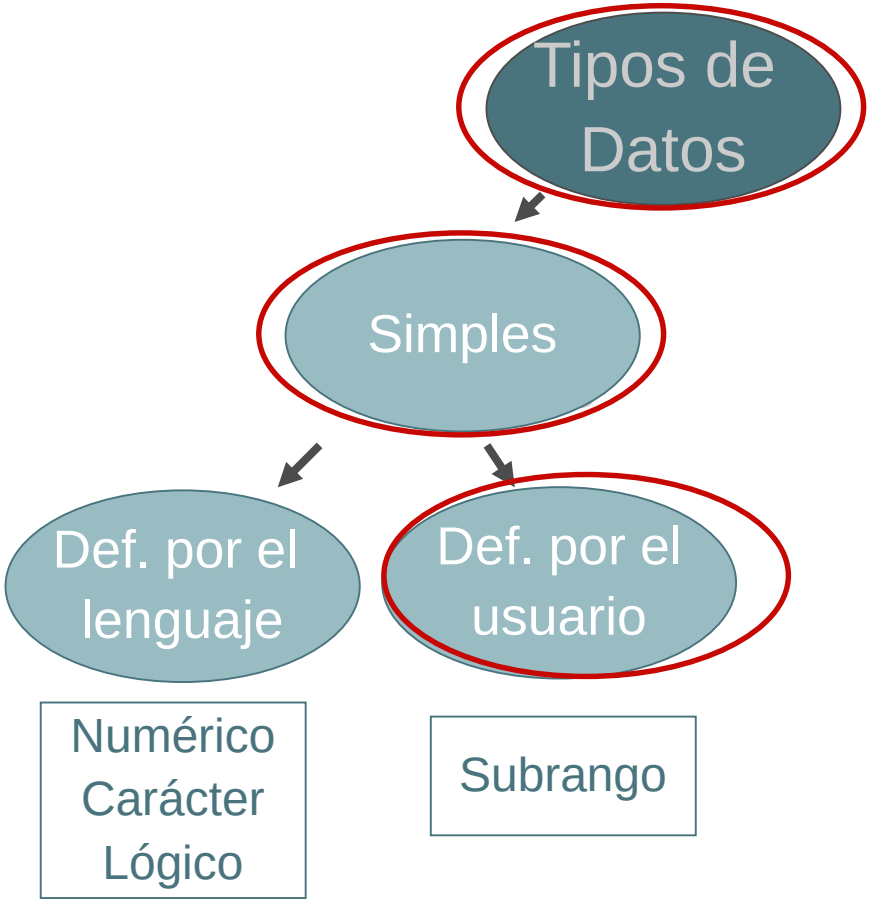
TEORIA 4

TEMAS de la CLASE

1. Tipos de datos definidos por el usuario
2. Ejercitación

Clasificación de Tipos De datos

Hasta aquí presentamos los tipos de datos simples que son aquellos que toman un único valor, en un momento determinado, de todos los permitidos para ese tipo.



Recordemos

Hemos trabajado los **tipos de datos simples** que se pueden considerar **estándar** en la mayoría de los lenguajes de programación.

Que los tipos de datos vistos se consideren estándar significa que el conjunto de valores de ese tipo, las operaciones que se pueden efectuar y su representación **están definidas** y **acotadas** por el lenguaje.

Recordemos además que

Un DATO en nuestras soluciones se utiliza para representar un objeto del mundo real.

El tipo de dato se caracteriza por:

- ✓ Un conjunto de valores o estados posibles.
- ✓ Un conjunto de operaciones permitidas.
- ✓ Una representación interna

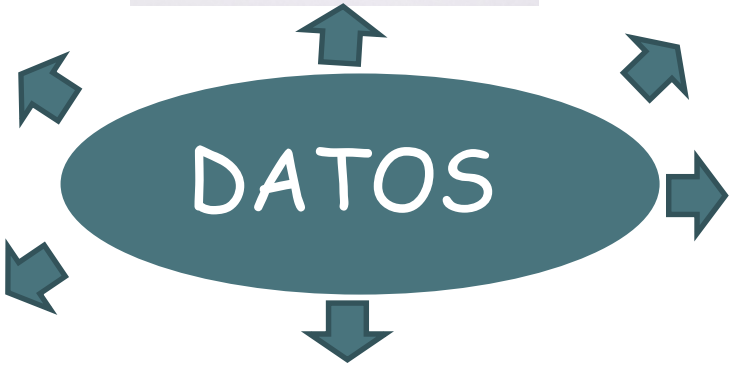
Puede surgir la necesidad de representar objetos del mundo real que utilicen tipos de datos diferentes a los estándard

Analicemos diferentes situaciones

Notas

Letras Minúsculas

Letras Mayúsculas



Meses

Días del mes

Edades de personas

Tipos de datos definidos por el usuario

- Sobre estos ejemplos de clases de datos no estándar pueden pensarse operaciones permitidas que sean específicas de cada caso.
- Pensemos operaciones posibles sobre estos datos:

Notas

- Obtener el promedio de notas de exámenes.
- Obtener el promedio de exámenes aprobados.

Días del mes

- Determinar la cantidad de días entre dos fechas
- Día siguiente a una fecha determinada
- Comparar dos fechas

Edades de personas

- Indicar la persona mas longeva.
- Obtener el promedio de las edades

Tipos de datos definidos por el usuario

Un aspecto **muy importante** en los lenguajes de programación es la capacidad de **especificar y manejar datos no estándar**, indicando valores permitidos, operaciones válidas y su representación interna.

Esto permite:

- Aumento de la riqueza expresiva del lenguaje, con mejores posibilidades de abstracción de datos.

- Mayor seguridad respecto de las operaciones que se realizan sobre cada clase de datos.

- Límites preestablecidos sobre los valores posibles que pueden tomar las variables que corresponden al tipo de dato.

Tipos de datos definidos por el usuario

¿Qué ventajas tiene DECLARAR tipos?

- **Flexibilidad:** en el caso de ser necesario modificar la forma en que se representa el dato, sólo se debe modificar una declaración en lugar de un conjunto de declaraciones de variables.

- **Documentación:** se pueden usar como identificador de los tipos, nombres autoexplicativos, facilitando de esta manera el entendimiento y lectura del programa.

- **Seguridad:** se reducen los errores por uso de operaciones inadecuadas del dato a manejar, y se pueden obtener programas más confiables.

Tipos de datos definidos por el usuario

Un *tipo de dato definido por el usuario* es aquel que no existe en la definición del lenguaje, y el programador es el encargado de su especificación.

Sintéticamente entonces un Tipo significa una clase de datos que tiene asociado:

- Un rango de valores posibles.
- Una forma de representación.
- Un conjunto de operaciones permitidas.
- Un conjunto de condiciones de valores permitidos que se pueden verificar.

Tipos de datos definidos por el usuario

En Pascal, los tipos deben ser declarados antes de ser usados.

La declaración de tipos se hace a través de la palabra clave TYPE de la siguiente forma:

```
TYPE identificador = tipo;
```

Nombre con que se conocerá al tipo de dato en el programa.

Puede ser un tipo estándar o alguno de los tipos de datos definidos por el usuario.

Esquema general de un programa que usa tipos definidos por el usuario

```
Program nombre;
Const
   ……
Type
   ……
Var
   …..
Begin      { Cuerpo del programa}
    ……
   …..   {Instrucciones ejecutables}
End.
```

Zona de Declaración de constantes

Zona de Declaración de tipos

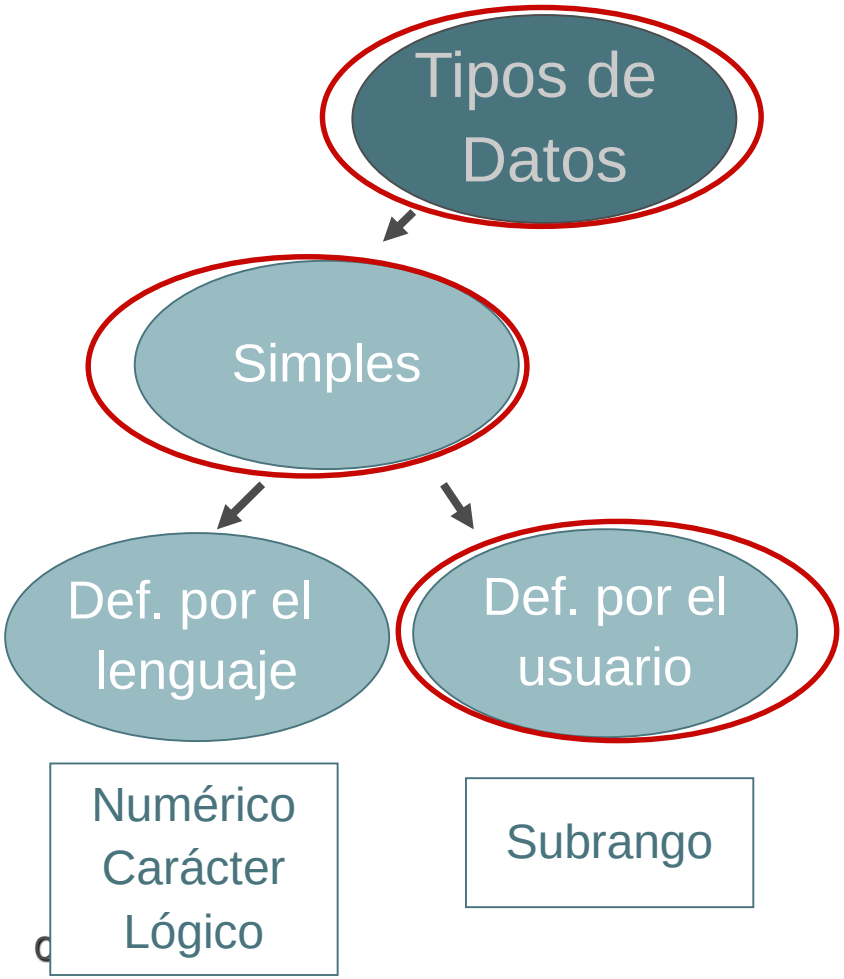
Zona de Declaración de variables

Zona de Instrucciones ejecutables

Clasificación de los Tipos de Datos

Recordemos la clasificación de los tipos de datos ya vista…

Ahora vamos a comenzar a trabajar con un tipo de dato **simple y definidos por el usuario**.



Tipo de dato definido por el usuario: SUBRANGO

En algunos casos, el tipo de dato puede tomar algunos de todos los valores de un tipo ordinal, esto es un SUBRANGO de los valores de ese tipo.

Notas

Entre 0 y 10

Letras Minúsculas

Entre 'a' y 'z'

Días del mes

Entre 1 y 31

Meses

Entre 1 y 12

Letras Mayúsculas

Entre 'A' y 'Z'

Edades de personas

Entre 0 y 130

Tipo de dato definido por el usuario: SUBRANGO

Un tipo de dato **subrango** es un tipo simple y ordinal que consiste de una sucesión de valores extraídos de un tipo ordinal base.

Para declarar un tipo **SUBRANGO** se deben especificar los valores inicial y final de la sucesión, separados por dos puntos seguidos:

```
Type  identificador  =  valor inicial .. valor final;
```

Tipo de dato definido por el usuario: SUBRANGO

Pensemos...

¿Cual es el tipo base?

¿Cual es el rango de valores?

Notas

Entre 0 y 10

Letras Minúsculas

Entre 'a' y 'z'

Días del mes

Entre 1 y 31

Meses

Entre 1 y 12

Letras Mayúsculas

Entre 'A' y 'Z'

Edades de personas

Entre 0 y 130

Tipo de dato definido por el usuario: SUBRANGO

```
Program nombre;
Const
  fin = 1999;
Type
  siglo_pasado= 1900 .. Fin;
   mayusculas = 'A' .. 'Z';
Var
  letra: mayusculas;
   año: siglo_pasado;
Begin      { Cuerpo del programa}
   ……
   …..     {Instrucciones ejecutables}
End.
```

Tipo SUBRANGO - Operaciones

- Las operaciones de un tipo de dato subrango se heredan del tipo base.

Los tipos base de los que se pueden definir subrangos son:

- ✓ Enteros
- ✓ Caracteres

- La ocupación en memoria estará condicionada por el tipo base

```
Program ejemplo;
Const  minimo = 1;
            maximo = 500;

Type
   rango  =  minimo .. maximo;
   meses = 1 .. 12;
   letras = 'A' .. 'z';

Var
    dato1, dato2 : rango;
    descanso : meses;
    letra: letras;

Begin
    dato1 := dato1 Div dato2;
    descanso:= 1;
    Read (letra);
    ............
```

Tipo SUBRANGO - Ventajas

¿Por qué son útiles los tipos subrango?

- Facilita el chequeo de posibles errores, pues permite que el lenguaje verifique si los valores asignados se encuentran dentro del rango establecido.

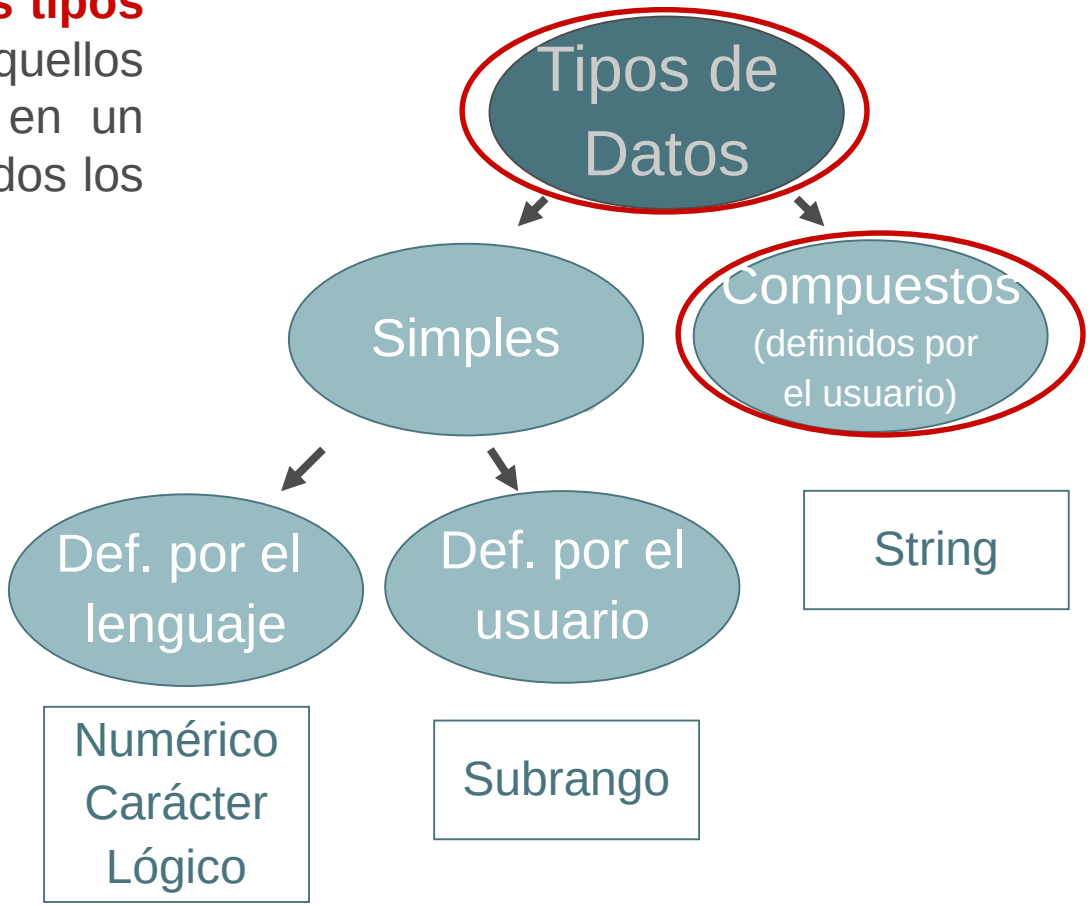
- Ayuda al mantenimiento del programa.

Ejercicio: Plantee una declaración de tres datos que puedan utilizarse para definir una fecha (día, mes, año) lo mas ajustada posible a fechas reales…

Realice un programa que lea edades de 20 personas e informe el promedio de edades y la edad mas grande.

Clasificación de Tipos De datos

- **Hasta aquí presentamos los tipos de datos simples** que son aquellos que toman un único valor, en un momento determinado, de todos los permitidos para ese tipo.



- **Ahora comenzaremos a trabajar con los tipos de datos compuestos** que son aquellos que pueden tomar varios valores a la vez que guardan alguna relación lógica entre ellos.

Tipo de dato definido por el usuario: STRING

Un tipo de dato string es una sucesión de caracteres de longitud determinada.

```
TYPE identificador = string [ longitud ];
```

- Longitud es el número máximo de caracteres que puede contener el dato.
- En PASCAL cuando no se especifica la longitud ese identificador podrá contener como máximo 255 caracteres.
- La cantidad de memoria que utiliza una variable está determinada por su longitud. Recordar que cada carácter ocupa 1 byte.

Tipo de dato definido por el usuario: STRING

```
Program nombre;
Type
  cadena10 = string [10];
  cadena25 = string [25];
  fecha    = string [8];
  día      = string [2];

Var
   nom1, nom2, nom3 :  cadena10;
   apellido :  cadena25;
   fecha1, fecha2 : fecha;

Begin      { Cuerpo del programa}
   ……
   …..     {Instrucciones ejecutables}
End.
```

Tipo de dato STRING: Operaciones

Las operaciones permitidas son:

- **Asignación (:=)**
- **Entrada/Salida (Read / write)**
- **De relación (>,<,=,…)**

Tipo de dato STRING: Operaciones

- **Asignación**

➢ Para asignar valor a una variable de tipo de dato string se hace igual que en una variable de tipo carácter (:=)

➢ Si se le asigna mayor cantidad de caracteres que lo declarado como longitud máxima, los últimos a partir de esa longitud se pierden y se dice que la hilera de caracteres “se trunca”.

```
Program uno;
Type
   cadena20= string [20];
   cadena5 = string [5];
Var
 cad1: cadena20;
 cad2: cadena5;

Begin
  cad1:= ‘buenos días!’;
  cad2:= cad1;
End.
```

Tipo de dato STRING: Operaciones

- **Entrada/Salida**

➢ El tipo de dato string admite las operaciones Read y Write de Pascal.

➢ Si la cadena ingresada supera la longitud declarada para el dato string entonces serán descartados los caracteres que se encuentran mas a la derecha.

```
Program uno;
Type
   cadena20= string [20];
   cadena5 = string [5];
Var
   cad1: cadena20;
   cad2: cadena5;
Begin
  read (cad1, cad2);
  write (cad1);
  write (cad2);
End.
```

Tipo de dato STRING: Operaciones

- **De Relación (=, <>, <=, =>)**

➢ Si las cadenas que se comparan son de igual longitud y contienen los mismos símbolos, en el mismo orden, el resultado de la operación es verdadero.

➢ Estos operadores realizan la comparación carácter por carácter. Si tienen distinta longitud el resultado de la comparación es falso.

```
Program uno;
Type
   cadena20= string [20];
   cadena5 = string [5];
Var
 cad1: cadena20;
 cad2: cadena5;
Begin
  cad1:= ‘buenos días!’;
  cad2:= ‘ggg´;
  if (cad1 = cad2) then…
……
End.
```

Se leen nombres y edades de alumnos que cursan CADP en el aula 5. Realice un programa que informe el nombre y la edad del alumno con mas edad. El ingreso de la información finaliza cuando se lee el nombre ‘ZZZ’.

Algoritmo

Analicemos los datos…

¿Escribimos el programa?

```
mientras haya alumnos para procesar
        Leo el nombre y la edad del alumno
        Si (edad > edad máxima) entonces
                actualizo la edad máxima
                guardo el nombre como nombre “máximo”
Informo el nombre y edad del alumno con edad máxima
```

Extender el problema para conocer de cada aula el nombre y la edad del alumno mas grande y el promedio general de edades. El ingreso de la información para cada aula finaliza cuando se lee el nombre 'ZZZ' y se procesan 5 aulas.

Algoritmo

Analicemos los datos...

¿Escribimos el programa?

```
Para cada aula
    mientras haya alumnos para procesar
            Leo el nombre y la edad de la persona
            incrementar Contador de alumnus
            sumar edad al total
            Si (edad > edad máxima) entonces
                    actualizo la edad máxima
                    guardo el nombre como nombre "máximo"
    Informo el nombre y edad del alumno con edad máxima
Calcular y mostrar promedio
```