

Algorítmica y Programación II

Puntero

Estructuras estáticas

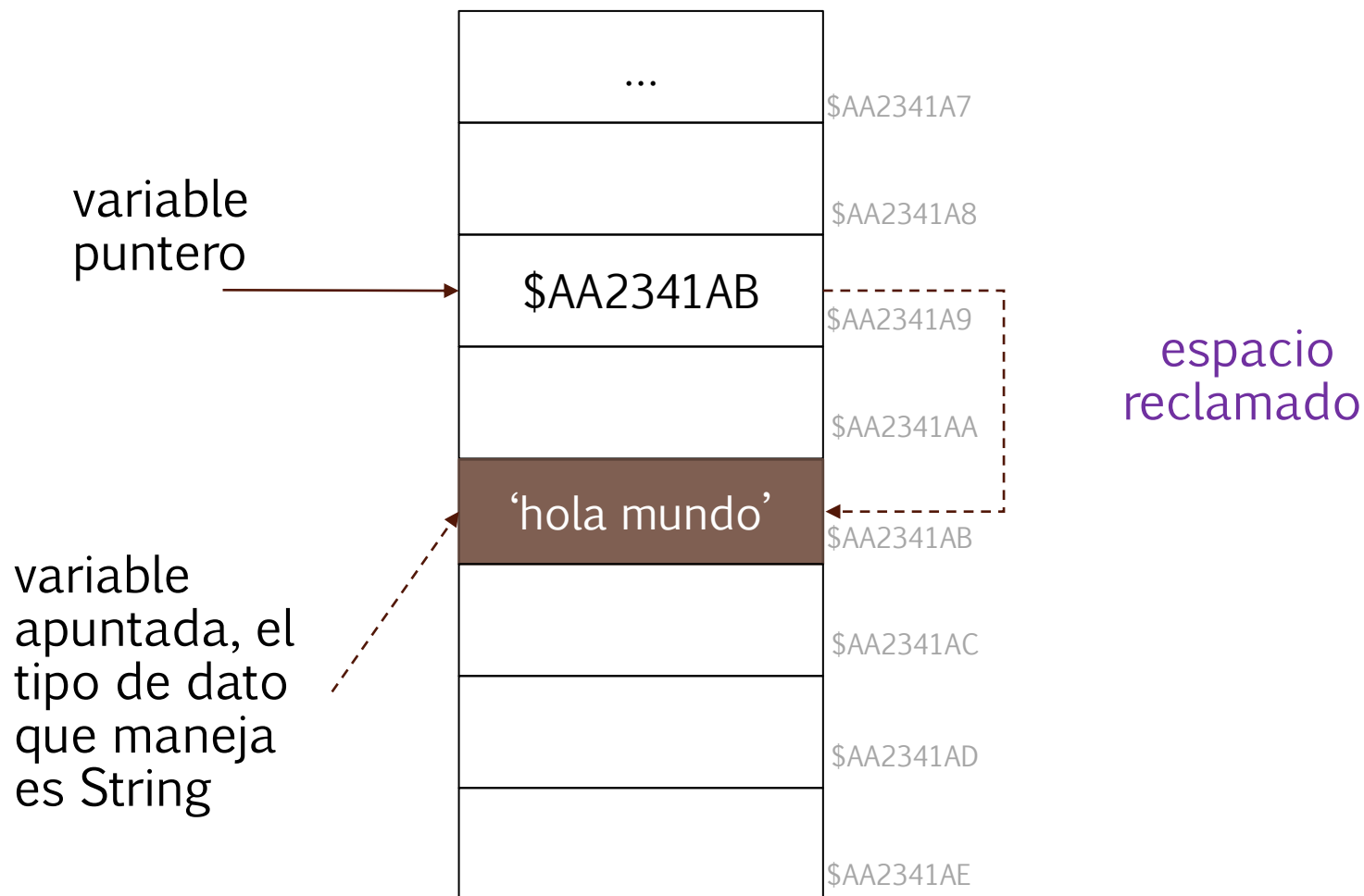
- › Las estructuras de datos vistas hasta el momento, salvo los archivos, se **almacenan de forma estática** en la memoria.
- › El espacio que ocuparán se reclama y reserva al comenzar la **ejecución** del programa.
- › **No podemos cambiar el tamaño** reservado durante la ejecución.

Estructuras dinámicas

- › Los lenguajes de programación proveen de mecanismos para **crear estructuras dinámicas**.
- › En pascal el mecanismo será la **utilización de punteros**
- › Se reclamará espacio y se liberará al terminar su uso.

- › Puntero – es un tipo de dato que:
 - el contenido es una **dirección de memoria**.
 - la dirección de memoria contiene el valor de interés.
 - el tamaño que ocupa un puntero es fijo
 - el tamaño que ocupa el contenido no existe hasta que se reclama.
 - el espacio que ocupa el contenido puede ser liberado

PUNTEROS



DECLARACION DE PUNTEROS

el circunflejo
antecede al
tipo

Type

TPunteroStr = ^String;

TPunteroInt = ^Integer;

TPunteroPer = ^TPersona

TPersona = Record

nombre,

apellido : String

...

end;

var ptr : TPunteroPer;

OPERACIONES (de puntero)

- › Las variables dinámicas son por definición aquellas que se **crean cuando se necesitan** y se destruyen cuando ya han cumplido con su cometido.
- › Creación, asignación y destrucción de variables dinámicas :
 - **New (puntero)** : reclama espacio y asigna a la variable puntero la dirección para poder utilizarla
 - **Dispose (puntero)** : libera el espacio apuntado por la variable puntero.
 - **Asignación**: punteros del mismo tipo o **NIL** (apunta a una dirección nula)

EJEMPLO - OPERACIONES

```
Type
    TPuntero = ^String;
var
    ptr:TPuntero;
begin

    new(ptr);

    ptr^:= 'hola mundo';

    writeln(ptr^);

end.
```



OPERACIONES (contenido)

- › Cuando nos referimos al contenido, al espacio apuntado por el puntero, podemos realizar **todas las operaciones conocidas**:
 - Lectura
 - Escritura
 - Asignación
 - Comparación

- › Para hacer referencia al contenido debemos utilizar el circunflejo ^ y que el puntero haya reclamado espacio (new).

- › Si nos referimos al contenido de un puntero que no “apunta” a ningún espacio tendremos un error en tiempo de ejecución.

EJEMPLO – OPERACIONES

Type

```
TPuntero = ^String;
```

var

```
ptr, aux: TPuntero;
```

Begin

```
aux := nil;
```

```
ptr := aux;
```

```
readln(ptr);  
writeln(ptr);
```

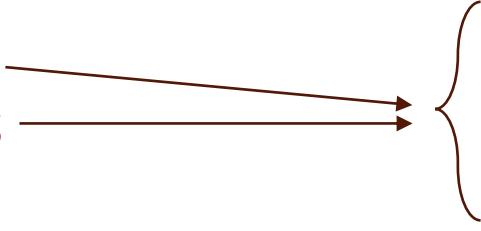
```
//new(ptr)
```

```
dispose(ptr);  
writeln(ptr^);
```

```
end.
```



operaciones no permitidas



Si no se reserva espacio no se pueden realizar

EJEMPLO

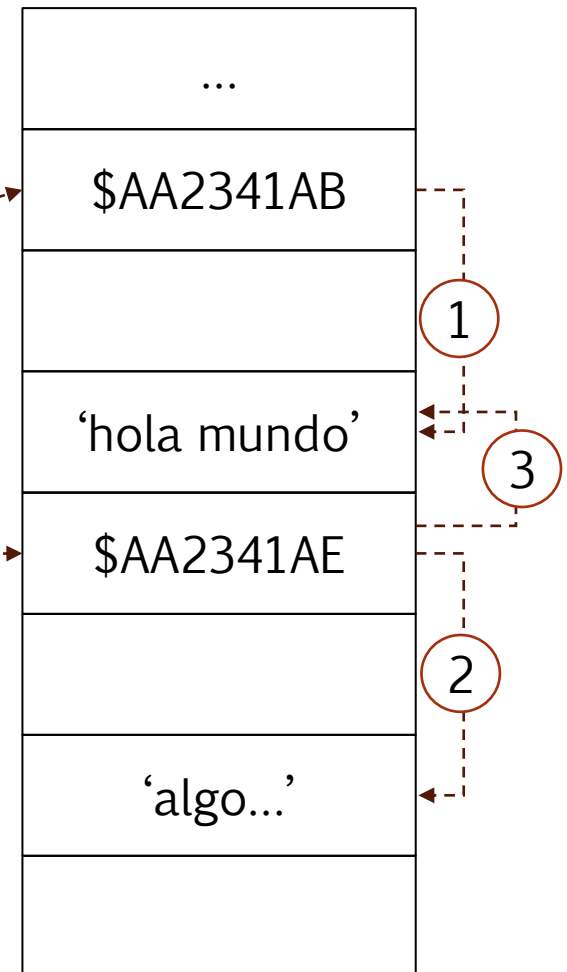
Type

```
TPuntero = ^String;  
var a,b:TPuntero;
```

```
new(a); ①  
a^:= 'hola mundo';
```

```
new(b); ②  
b^:= 'algo...';
```

```
writeln(b^);  
b := a; ③  
writeln(b^);
```



Ejercicio

- › Realice un programa que permita el ingreso por parte del usuario de 2 cadenas de texto e informe finalmente si las mismas son iguales.
 - Utilice variables de tipo puntero para su implementación.

TDA - Bibliografía

- Data Structures. Nalle Dale.
- Algoritmos, Datos y programas. Armando, De Gusti.
- Estructuras de datos y algoritmos. Mark, Weiss.
- <http://wiki.freepascal.org/Pointer>