

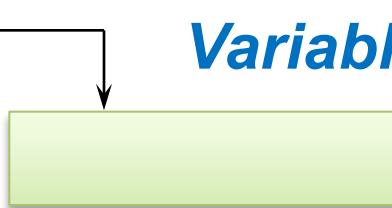
Apuntadores

Ing. Armandina J. Leal Flores

Concepto

- ▶ Las variables llamadas **apuntadores** almacenan una dirección de memoria.

Variable tipo pointer



Variable tipo target

Declaración de variables apuntador

- ▶ Ejemplos:

```
int * ipApuntador;
```

```
int* ipFrente;
```

```
double dFinal, *dpFrente; //dFinal no es apuntador
```

```
int *ipTop, iBottom; //iBottom no es apuntador
```

Operador & (address operator)

&*nombrevariable*

- ▶ Útil para obtener la dirección de la variable.
- ▶ El apuntador al que se le asigne la dirección debe ser del tipo de la variable.

```
int *ipApunt;
int iM;
ipApunt = &iM;
```

Toma nota.....

- ▶ Lo siguiente:

```
int iM, *ipApunt = &iM;
```

- ▶ Es equivalente a:

```
int iM, *ipApunt;  
ipApunt = &iM;
```

Operador & (address operator)

▶ Ejemplo

int iM, *ipApunt;

ipApunt

iM

*dirección de
apunt = 2c3456*

*dirección de
m = 2c3460*

The diagram illustrates a pointer assignment. On the left, the variable **ipApunt** is assigned the value of the memory location **iM**. The variable **ipApunt** is shown in bold black text. To its right is an equals sign followed by the address of the variable **iM**, which is **2c3460**. An arrow points from the variable **ipApunt** to the value **2c3460**, indicating that **ipApunt** contains the memory address of **iM**. The variable **iM** is represented by a green rectangular box.

Operador &

- ▶ Útil para manejar la variable tipo target desde la variable apuntador.
- ▶ Ejemplo:

```
int iM, *ipApunt = &iM;
```

```
*ipApunt = 5;           //mismo efecto que iM=5
```

- ▶ ***ipApunt** es una variable entera.

Cuidado.....

```
int iM = 5, *ipApunt = &iM;
```

- ▶ **ipApunt** se refiere a la dirección almacenada dentro de la variable ipApunt.
Ej. ipApunt = 2C3460
- ▶ ***ipApunt** se refiere al contenido de la variable a la que apunta ipApunt.
Ej. *ipApunt = 5



Cuidado....

- ▶ Se debe asignar una dirección a un apuntador antes de emplearlo.
- ▶ Ejemplo con errores:

```
int *ipP, *ipQ;
```

```
*ipP = 7;
```

```
cout << *ipQ;
```

Null

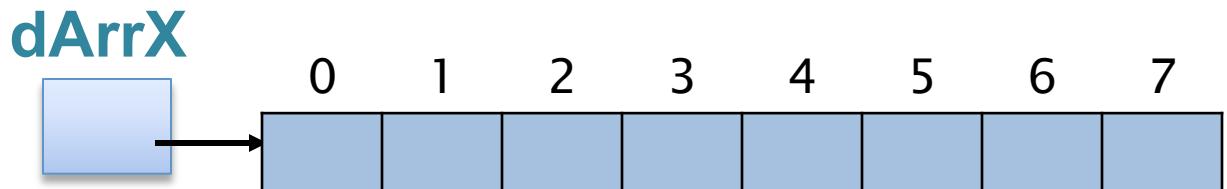
- ▶ Útil para indicar que un apuntador no contiene una dirección.

```
int *ipApunt;  
ipApunt = NULL;
```

- ▶ **NULL** está incluido en la biblioteca iostream

Arreglos y Apuntadores

Representación física en C++ de double dArrX[8]



double dArrX[8];

```
int main()
{
    double dY, dArrX[8];
    ....
    dY = dArrX[5];
}
```

Asigna a **dY** el valor de la quinta celda posterior a la celda a la que apunta **dArrX**

Arreglos como parámetros

```
void haceAlgo( double dArrZ[ ] )
{
    dArrZ[5] = 99.9;
}
```

dArrZ es un apuntador a una secuencia de celdas.

La función puede recibir arreglos de diferentes tamaños.

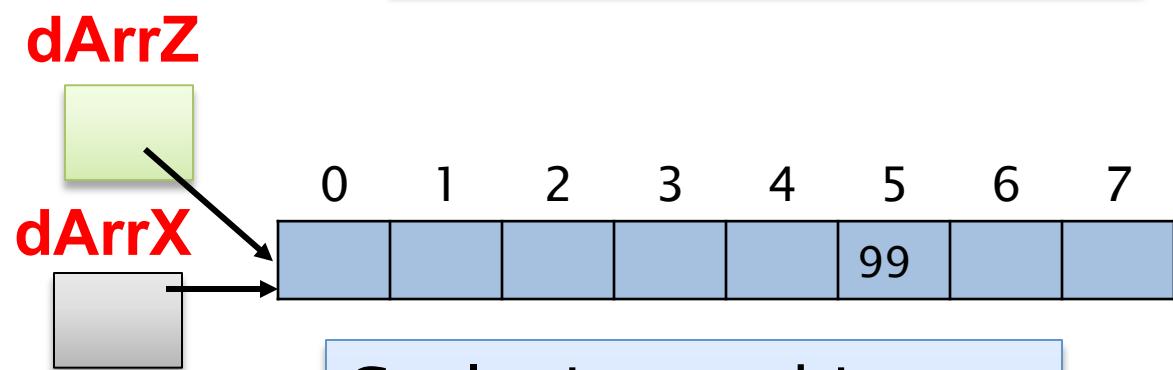
La función NO sabe cuantas celdas tiene el arreglo, solo conoce donde empieza.

Arreglos y Parámetros

```
void haceAlgo( double dArrZ[ ] )  
{  
    dArrZ[5] = 99;  
}
```

Los arreglos como parámetros son **parámetros por referencia** (de entrada-salida).

```
int main()  
{  
    double dArrX[8];  
    -----  
    haceAlgo(dArrX);  
}
```



Cualquier cambio que se realice en **dArrZ** se está haciendo en **dArrX**.

Arreglos y Apuntadores

void haceAlgo(**int *ipZ**); **es igual a** void haceAlgo(**int iArrZ[]**);

Asigna a **iW** el valor de la
celda apuntada por **iArrZ**
más **5** celdas más

iW = iArrZ[5]; **es igual a** **iW = * (iArrZ + 5);**

iArrX[iN] = 21; **es igual a** *** (iArrX + iN) = 21;**

***** significa “*pointed to by*”