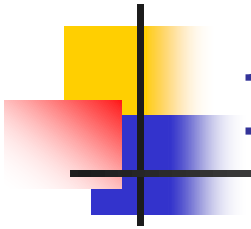
Fundamentos de Programación 2

Ing. Marco Aedo López

EPIS – 2024 marcoaedo@hotmail.com maedol@unsa.edu.pe

Arreglos (Arrays): Búsquedas y Ordenamientos

Capítulo 1



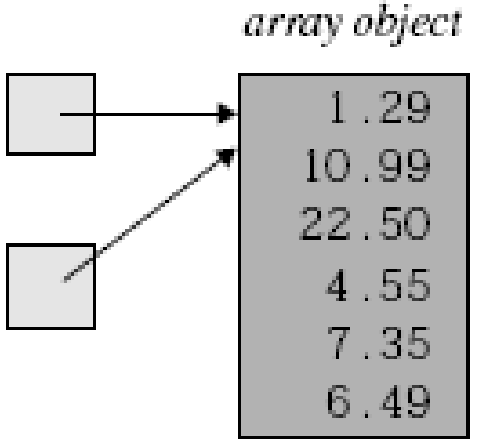
10. Arreglos llenados parcialmente

- Usamos parte de la capacidad de almacenamiento del arreglo, pero no toda
- Sería necesario llevar el rastro del número de elementos llenados en el arreglo, de tal manera que se procesarían sólo ellos, no todos los elementos del arreglo
- La variable `elementosIngresados` hace el seguimiento de la cantidad de elementos ingresados que tiene el arreglo

11. Copiando un arreglo

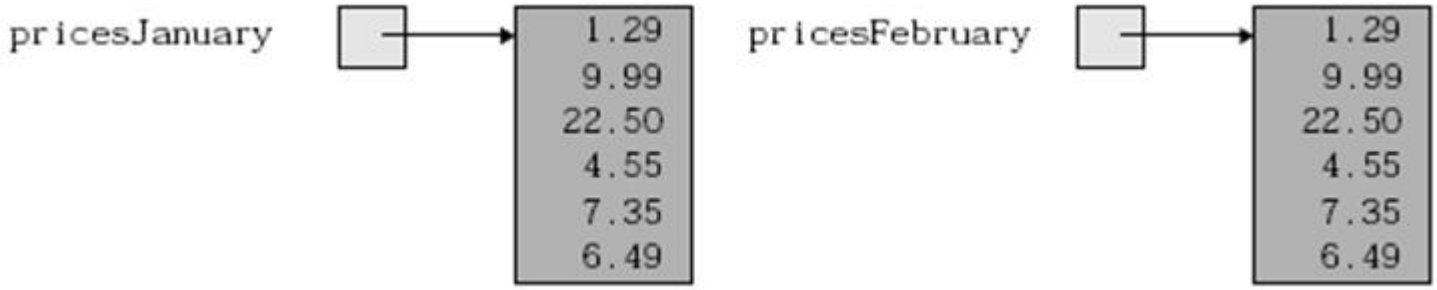
- Recordar: como con todos los objetos y sus variables de referencia, si asignamos una variable de referencia de un arreglo a otra variable de referencia de otro arreglo, ambas variables de referencia apuntarán al mismo objeto arreglo
- Si nuestra meta es hacer una copia de un arreglo `arr1`, ¿cuál sería el problema?

arr2 = arr1;



11. Copiando un arreglo

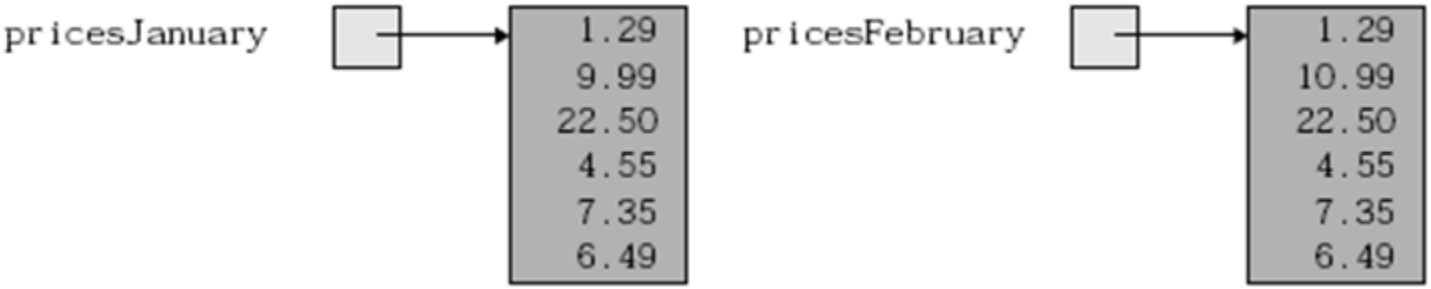
- Usualmente desearíamos que el original y la copia del arreglo apunten a diferentes objetos de arreglo
- Hacerlo asignando los elementos, uno a uno



EJERCICIO

- Tiene un arreglo con 6 precios que maneja la tienda en Enero, se quiere un nuevo arreglo para Febrero con los mismos valores de Enero, pero cambiando el segundo valor a 10.99
- Salida:

```
   Ene     Feb
  1.29    1.29
  9.99   10.99
 22.00   22.00
  4.55    4.55
  7.35    7.35
  6.49    6.49
```



11. Copiando un arreglo

```
public class Fundamentos2 {
    public static void main(String[] args){
        double[] preciosEnero ={1.29, 9.99, 22.50, 4.55, 7.35, 6.49};
        double[] preciosFebrero = new double[preciosEnero.length];

        for (int i=0; i<preciosEnero.length; i++)
            preciosFebrero[i] = preciosEnero[i];

        preciosFebrero[1] = 10.99;

        System.out.println("\t"+"Ene\t"+ "Feb");
        for (int i=0; i<preciosEnero.length; i++)
            System.out.println("\t"+preciosEnero[i]+"\t"+ preciosFebrero[i]+"\n");
    }
}
```

11. Copiando un arreglo

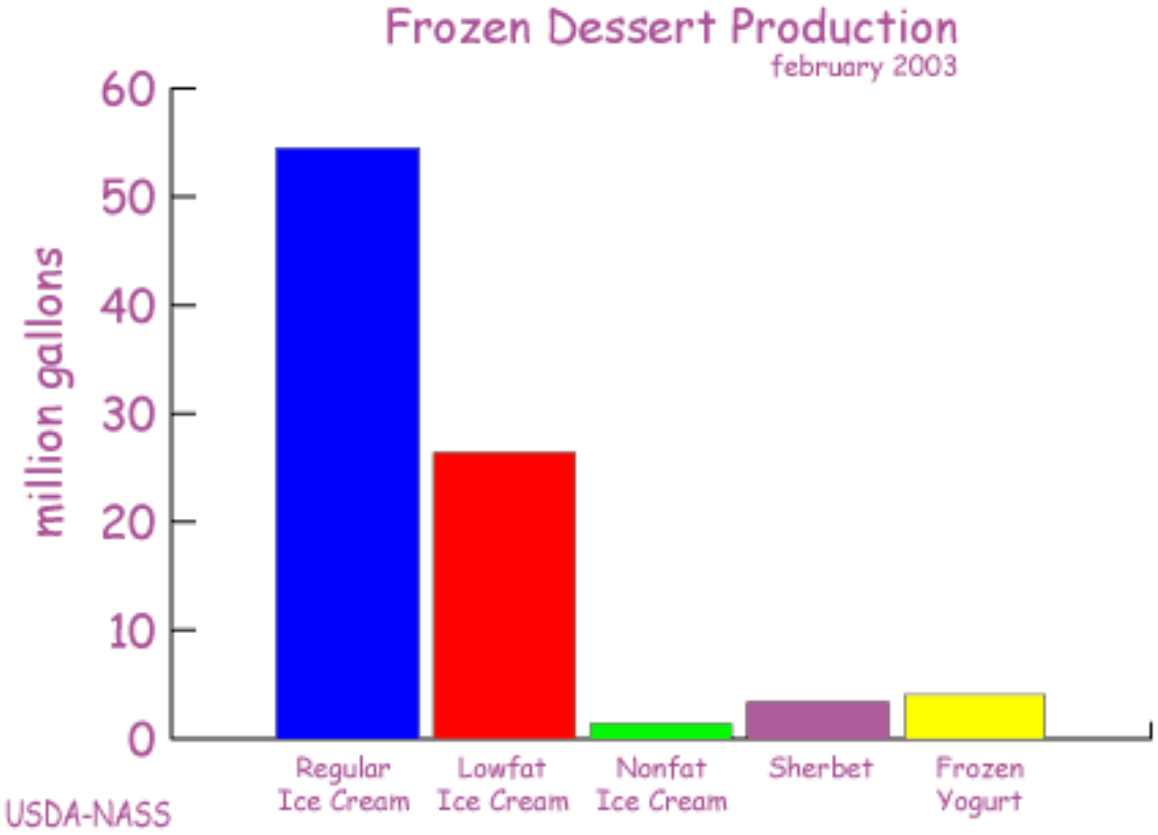
- Copiar datos de un arreglo a otro es una tarea común, así Java nos brinda un método especial **`System.arraycopy`**
- Permite copiar un número de elementos desde cualquier lugar en un arreglo a cualquier lugar en otro arreglo

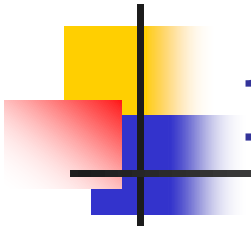
```
System.arraycopy(preciosEnero,0,preciosFebrero,0,6);
preciosFebrero[1]=10.99;
```

- Argumentos:
  - Nombre arreglo origen
  - Índice del primer elemento del origen a copiar
  - Nombre del arreglo de destino
  - Índice del primer elemento a reemplazar en el destino
  - Número total de elementos a copiar

12. Histogramas

- Gráfico que muestra cantidades para un conjunto de categorías. Muestra con barras, de mayor o menor tamaño





12. Histogramas

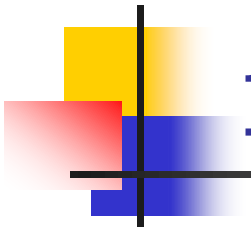
- Ejercicio 1:
  - Suponga que tiene 3 monedas, al lanzar las 3 monedas una cantidad de veces Ud. desea saber cuántas veces salió 0 caras, 1 caras, 2 caras y 3 caras. En otras palabras desea saber la probabilidad (distribución de frecuencia) para el número de caras
- Solución:
  - Escribir un main() que simula el lanzamiento de 3 monedas un millón de veces
  - Imprimir los resultados de la simulación en forma de histograma:
    - Para cada posible caso (4 posibles), imprime una barra que represente el número de veces que el caso ocurra
    - Para simular una barra, usar serie de *, donde cada uno represente un 1% del número total de lanzamientos

12. Histogramas

- Salida:

```
Número de veces que cada caso ocurría (0, 1, 2, 3 caras):
 0  124685 ************
 1  374759 ************************************
 2  375420 ************************************
 3  125136 ************
```

- Tips para la implementación:
  - Usar un arreglo `frequencia` para mantener el seguimiento del número de veces que cada valor de cuenta de caras ocurre. La `frecuencia[0]` tiene el número de veces en que ninguna de las 3 salió cara. La frecuencia`[1]` tiene el número de veces en que salió 1 cara, etc.
  - Después de cada lanzamiento simulado, añadir 1 al elemento apropiado



13. Paso de arreglos a métodos

- Los métodos también pueden aceptar argumentos de tipo arreglo
- Siempre lo hace por valor, pero no se pasa toda una copia del arreglo
- Se pasa una copia de la dirección del arreglo o referencia al arreglo (con esa dirección se puede alterar los valores del arreglo)

13. Paso de arreglos a métodos

Code

A

```
double[] arrayUno=new double[20];
minimo = buscaMinimo(arrayUno);
```

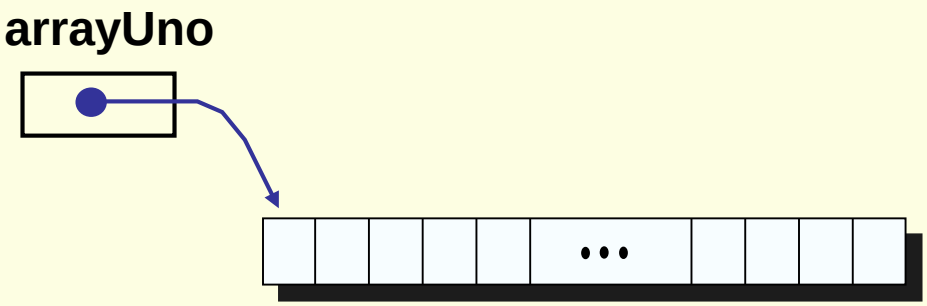
```
public int buscaMinimo(double[] number)){

      …


}
```

En A antes de buscaMinimo

State of Memory



A. Variable local number no existe antes de la ejecucion del método

13. Paso de arreglos a métodos

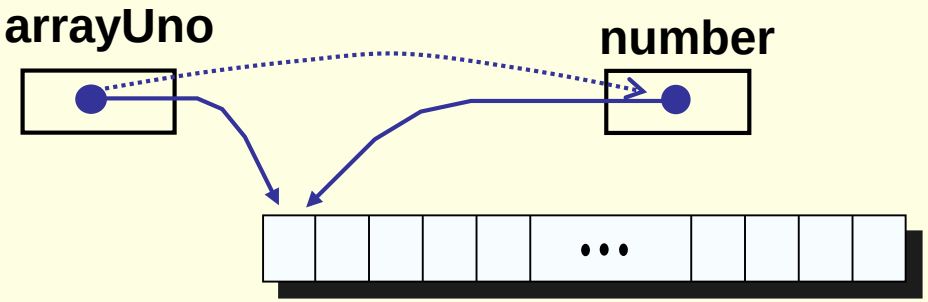
Code

```
double[] arrayUno=new double[20];
minimo = buscaMinimo(arrayUno);
```

```
public int buscaMinimo(double[] number)){
      …
}
```

B

La dirección se copia en B



State of Memory

B. El valor del argumento, que es una direccion, es copiado en el parametro

13. Paso de arreglos a métodos

Code

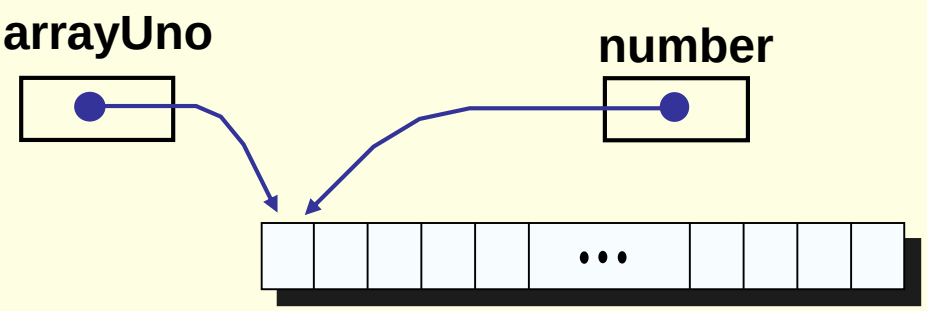
```
double[] arrayUno=new double[20];
minimo = buscaMinimo(arrayUno);
```

```
public int buscaMinimo(double[] number)){
    ...
}
```

C

State of Memory

Mientras en C dentro del método



C. El arreglo es accedido via number dentro del metodo.

13. Paso de arreglos a métodos

Code

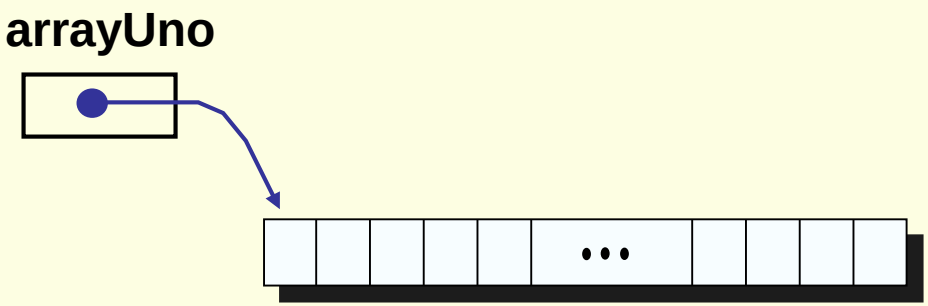
```
double[] arrayUno=new double[20];
minimo = buscaMinimo(arrayUno);
```

D

```
public int buscaMinimo(double[] number)){
    …
}
```

State of Memory

En D después de buscaMinimo



D. El parametro number es borrado. El argumento arrayUno aún apunta al mismo objeto

13. Paso de arreglos a métodos

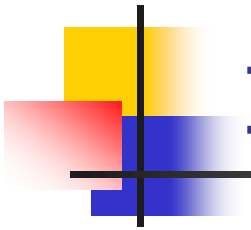
```
public class JavaApplication3{

    public static void main(String[] args){
        int[] a={1,3,7,8};
        imprimir(a);
    }
    static void imprimir(int[] b){
        for(int i=0;i<b.length;i++)
            System.out.println(b[i]);
    }
}
```

13. Paso de arreglos a métodos

```
public class JavaApplication3{

    public static void main(String[] args){
        int[] a={1,3,7,8};
        imprimir(a);
        multiplicarx2(a);
        imprimir(a);
    }
    static void imprimir(int[] b){
        for(int i=0;i<b.length;i++)
            System.out.println(b[i]);
    }
    static void multiplicarx2 (int[] b){
        for(int i=0;i<b.length;i++)
            b[i]*=2;
    }
}
```



13. Paso de arreglos a métodos

PROBLEMA:

Método que devuelva la suma de todos los elementos de un arreglo

13. Paso de arreglos a métodos

```
public class JavaApplication3{

    public static void main(String[] args){
        int suma;
        int[] a={1,3,7,8};
        suma=sumArreglo(a);
        System.out.println("la suma de los elementos del arreglo es: "+suma);
    }
    static int sumArreglo(int[] b){
        int sum=0;
        for(int i=0;i<b.length;i++)
            sum=sum+b[i];
        return sum;
    }
}
```

13. Paso de arreglos a métodos

PROBLEMA:

Sumar 2 arreglos unidimensionales de tamaño 10 con valores de notas enteras que vayan entre 0..20 y almacenarlos en un nuevo arreglo. Luego que imprima los valores de la suma con el formato:

"x + y = z"

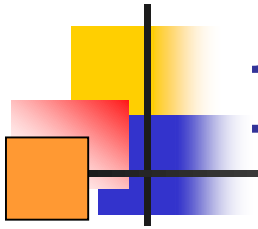
Crear métodos:

1) generar genera los valores aleatorios de los 2 arreglos
2) sumar calcula la suma de los 2 arreglos
3) imprimir imprime con el formato indicado

```java
public class JavaApplication3{

    public static void main(String[] args){
        final int tam=10;
        int[] a=new int[tam];
        int[] b=new int[tam];
        int[] c=new int[tam];

        generar(a,b);
        sumar(a,b,c);
        imprimir(a,b,c);
    }
    static void generar(int[] x,int[] y){
        Random rand=new Random();
        for(int i=0;i<x.length;i++){
            x[i]=rand.nextInt(21);
            y[i]=rand.nextInt(21);
        }
    }
    static void sumar(int[] x,int[] y,int[] z){
        for(int i=0;i<x.length;i++)
            z[i]=x[i]+y[i];
    }
    static void imprimir(int[] x, int[] y,int[] z){
        for(int i=0;i<x.length;i++)
            System.out.println(x[i]+" + "+y[i]+"\t="+z[i]);
    }
}
```



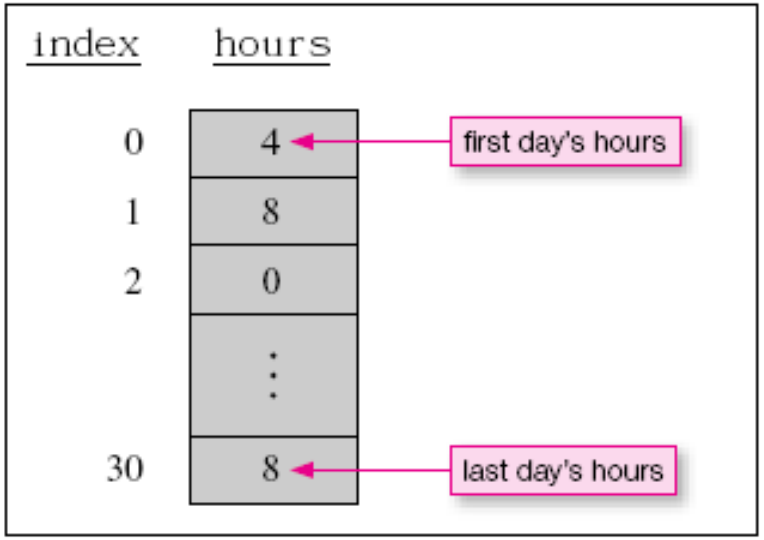
13. Paso de arreglos a métodos

Ejercicio 2:

Crear métodos ingresar(), modificar(), imprimir(), que permitan ingresar los n enteros de un arreglo, modificar multiplicando por 2 sus elementos e imprimir sus elementos. Los métodos deben permitir trabajar con arreglos unidimensionales de cualquier tamaño. Crear un main() que pruebe dichos métodos

EJERCICIO 3

- Ud. tiene el arreglo `horas`, inicializarlo con aleatorios entre 0 y 8 con la cantidad de horas que debe trabajar desde hoy hasta dentro de 30 días. Este arreglo debe actualizarse al día siguiente, desplazando todas las cantidades de horas en 1 día hacia adelante, e ingresando la cantidad de horas a trabajar en el día 30. Ingresar la cantidad de días que desea que pasen y simular cómo se va actualizando el arreglo `horas` día a día



Arreglos (Arrays)

Gracias