Clase 7

PROGRAMACIÓN 1

Objetivos del tema

- **Desarrollar aplicaciones con uso de arreglos de una dimensión y multidimensionales**
- **Realizar operaciones de ordenamiento, eliminación, e inserción**
- **Resolver problemas aplicando diseño descendente**

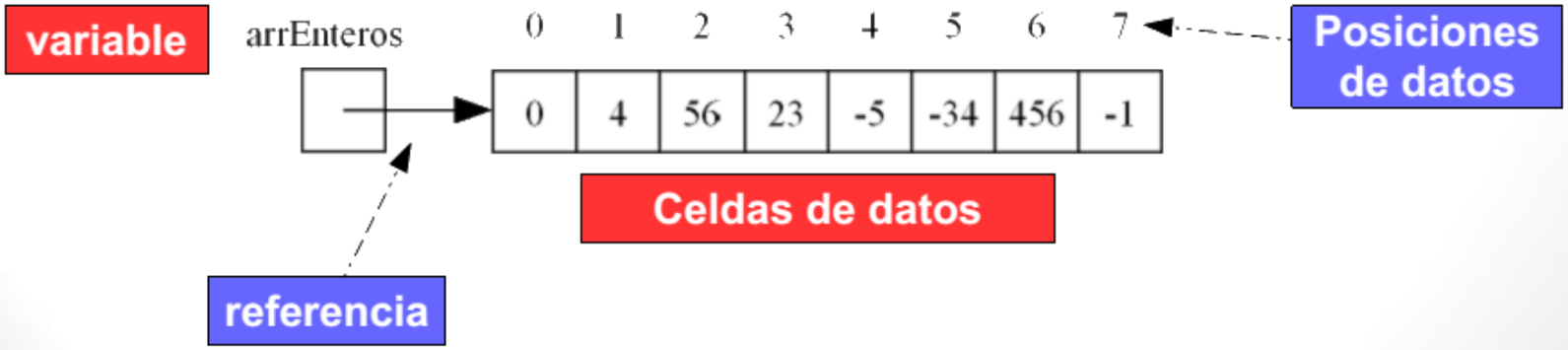
Arreglos

Los arreglos son estructuras de datos que consisten en elementos o celdas de información del mismo tipo

Los arreglos son considerados entidades “estáticas” ya que su tamaño no cambia una vez creados

Un arreglo tiene asignado un grupo de posiciones de memoria contiguas

Cada celda tiene asociado un número entero que indica la posición de la misma con respecto a las demás



Métodos

- Los arreglos sólo almacenan datos de un sólo tipo.
- Para definir un arreglo se debe indicar: el tipo de todos sus datos, la cantidad de celdas y un nombre de variable.
- Una vez definido el tamaño del arreglo se considera que no cambia.
- Un solo nombre representa a todos los elementos.
- A los elementos se accede en forma directa o aleatoria a través del nombre del arreglo y la posición deseada.
- Cada elemento del arreglo puede ser utilizado como cualquier variable.

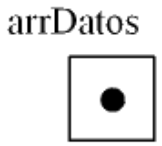
vec =	12	14	17	8	19	13	7	9	6	92
POSICIONES→	vec[0]	vec[1]	vec[2]	vec[3]	vec[4]	vec[5]	vec[6]	vec[7]	vec[8]	vec[9]

```
int A = vec[0] + vec[8];      // A = 12 + 6 =  18
int B = 2 + vec[3];           // B = 2 + 8 = 10
vec[0] = A + B;               // vec[0] = 18 + 10  = 28
```

Declaración de arreglos

Diferentes formas de declaración y definición de arreglos

```
int[] arrDatos;
```

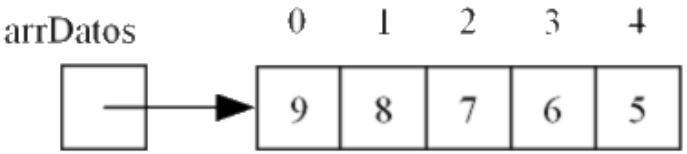


arreglo nulo
(sin espacio para datos)

```
int[] arrDatos = new int[5];
```

espacio de datos
inicializado con ceros

```
int[] arrDatos = {9, 8, 7, 6, 5};
```



espacio con datos
inicializados
explícitamente

Ejemplo

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Programa {
    final static int MAX = 5;
    public static void main(String args[]) {
            //MAS ADELANTE SE DEBE MODULARIZAR LA INICIALIZACION, CARGA E IMPRESION
            int B[]=new int[MAX];
            for (int pos=0;pos<MAX;pos++) {
                        System.out.println ("Ingrese integer: "+pos);
                        B[pos]=obtenerEntero();
            }
            for (int pos=0;pos<MAX;pos++)
                        System.out.println(B[pos]);
    }
    public static int obtenerEntero(){
              int valor    = 0;
              boolean enterovalido = false;
              BufferedReader entrada = new BufferedReader(new InputStreamReader(System.in));
              do {
                          try {
                                      valor = new Integer(entrada.readLine());
                                      enterovalido = true;
                          }
                          catch (Exception exc ) {
                                      enterovalido = false;
                          }
              } while (!enterovalido);
              return valor;
    }
}
```

Ejemplo

```
public class Programa {
    final static int MAX = 10;
    public static void main(String args[]) {
        //MAS ADELANTE SE DEBE MODULARIZAR LA INICIALIZACION,
        //CARGA E IMPRESION
         int B[]=new int[MAX];
         for (int pos = 0 ; pos < MAX; pos++) {
                  B[pos]=(int) (MAX*Math.random() + 1); // Entre 1 y MAX
         }
         System.out.println("el promedio es: "+promedio(B));
    }

    public static float promedio (int[] arr){
         float prom = 0.0f;
         for (int pos = 0 ; pos < MAX; pos++) {
                  prom += arr[pos];
         }
         prom=prom/MAX;
         return prom;
    }
}
```

Pasaje de parámetros

Paso de parámetros por valor

- Cuando se invoca al método se crea una nueva variable (el parámetro formal) y se le copia el valor del parámetro actual.
- El parámetro actual y el formal son dos variables distintas aunque puedan llegar a tener el mismo nombre.
- El método trabaja con la copia de la variable por lo que cualquier modificación que se realice sobre ella dentro del método no afectará al valor de la variable fuera.

Paso de parámetros por referencia

- Cuando se invoca al método se crea una nueva variable (el parámetro formal) a la que se le asigna la dirección de memoria donde se encuentra el parámetro actual.
- En este caso el método trabaja con la variable original por lo que puede modificar su valor.

Pasaje de parámetros en Java

- En Java todos los parámetros se pasan por valor.

- **Cuando el argumento es de tipo primitivo**, el paso por valor significa que cuando se invoca al método se reserva un nuevo espacio en memoria para el parámetro formal. **El método no puede modificar el parámetro actual**.

- **Cuando el argumento es una referencia (por ejemplo, un array o cualquier otro objeto)** el paso por valor significa que el método recibe una copia de la dirección de memoria donde se encuentra el objeto. **Por lo tanto el método puede modificar el contenido.**

Ejemplo

```
public class Programa {
    final static int MAX = 5;
    public static void main(String args[]) {
         //MAS ADELANTE SE DEBE MODULARIZAR LA INICIALIZACION E IMPRESION
          int B[]=new int[MAX];
          int a = 20;
          cargar_arreglo(B);
          System.out.println("Los datos son:");
          for (int pos = 0 ; pos < MAX; pos++)
                    System.out.println(B[pos]);
          cargar_variable_simple(a);
          System.out.println("La variable es:");
          System.out.println(a);
    }

    public static void cargar_variable_simple(int c) {
          c = 10;
    }

    public static void cargar_arreglo(int[] arr) {
          for (int pos = 0 ; pos < MAX; pos++)
                    arr[pos]=pos*2;
    }
}
```

Variable	Dirección	Valor
B	132	205
B[0]	205	?
B[1]	206	?
B[2]	207	?
B[3]	208	?
B[4]	209	?
a	301	20

Ejemplo

```
public class Programa {
    final static int MAX = 5;
    public static void main(String args[]) {
        //MAS ADELANTE SE DEBE MODULARIZAR LA INICIALIZACION E IMPRESION
         int B[]=new int[MAX];
         int a = 20;
         cargar_arreglo(B);
         System.out.println("Los datos son:");
         for (int pos = 0 ; pos < MAX; pos++)
                  System.out.println(B[pos]);
         cargar_variable_simple(a);
         System.out.println("La variable es:");
         System.out.println(a);
    }

    public static void cargar_variable_simple(int c) {
         c = 10;
    }

    public static void cargar_arreglo(int[] arr) {
         for (int pos = 0 ; pos < MAX; pos++)
                  arr[pos]=pos*2;
    }
}
```

Variable	Dirección	Valor
B	132	205
B[0]	205	?
B[1]	206	?
B[2]	207	?
B[3]	208	?
B[4]	209	?
a	301	20
arr	**452**	**205**

Ejemplo

```
public class Programa {
    final static int MAX = 5;
    public static void main(String args[]) {
         //MAS ADELANTE SE DEBE MODULARIZAR LA INICIALIZACION E IMPRESION
          int B[]=new int[MAX];
          int a = 20;
          cargar_arreglo(B);
          System.out.println("Los datos son:");
          for (int pos = 0 ; pos < MAX; pos++)
                    System.out.println(B[pos]);
          cargar_variable_simple(a);
          System.out.println("La variable es:");
          System.out.println(a);
    }

    public static void cargar_variable_simple(int c) {
          c = 10;
    }

    public static void cargar_arreglo(int[] arr) {
          for (int pos = 0 ; pos < MAX; pos++)
                    arr[pos]=pos*2;
    }
}
```

Variable	Dirección	Valor
B	132	205
B[0]	205	0
B[1]	206	2
B[2]	207	4
B[3]	208	6
B[4]	209	8
a	301	20
arr	452	205

Ejemplo

```
public class Programa {
    final static int MAX = 5;
    public static void main(String args[]) {
         //MAS ADELANTE SE DEBE MODULARIZAR LA INICIALIZACION E IMPRESION
          int B[]=new int[MAX];
          int a = 20;
          cargar_arreglo(B);
          System.out.println("Los datos son:");
          for (int pos = 0 ; pos < MAX; pos++)
                    System.out.println(B[pos]);
          cargar_variable_simple(a);
          System.out.println("La variable es:");
          System.out.println(a);
    }

    public static void cargar_variable_simple(int c) {
          c = 10;
    }

    public static void cargar_arreglo(int[] arr) {
          for (int pos = 0 ; pos < MAX; pos++)
                    arr[pos]=pos*2;
    }
}
```

Variable	Dirección	Valor
B	132	205
B[0]	**205**	**0**
B[1]	**206**	**2**
B[2]	**207**	**4**
B[3]	**208**	**6**
B[4]	**209**	**8**
a	301	20

Ejemplo

```
public class Programa {
    final static int MAX = 5;
    public static void main(String args[]) {
         //MAS ADELANTE SE DEBE MODULARIZAR LA INICIALIZACION E IMPRESION
          int B[]=new int[MAX];
          int a = 20;
          cargar_arreglo(B);
          System.out.println("Los datos son:");
          for (int pos = 0 ; pos < MAX; pos++)
                    System.out.println(B[pos]);
          cargar_variable_simple(a);
          System.out.println("La variable es:");
          System.out.println(a);
    }

    public static void cargar_variable_simple(int c) {
          c = 10;
    }

    public static void cargar_arreglo(int[] arr) {
          for (int pos = 0 ; pos < MAX; pos++)
                    arr[pos]=pos*2;
    }
}
```

Variable	Dirección	Valor
B	132	205
B[0]	205	0
B[1]	206	2
B[2]	207	4
B[3]	208	6
B[4]	209	8
a	301	20
c	**510**	**20**

Ejemplo

```
public class Programa {
    final static int MAX = 5;
    public static void main(String args[]) {
         //MAS ADELANTE SE DEBE MODULARIZAR LA INICIALIZACION E IMPRESION
          int B[]=new int[MAX];
          int a = 20;
          cargar_arreglo(B);
          System.out.println("Los datos son:");
          for (int pos = 0 ; pos < MAX; pos++)
                    System.out.println(B[pos]);
          cargar_variable_simple(a);
          System.out.println("La variable es:");
          System.out.println(a);
    }

    public static void cargar_variable_simple(int c) {
          c = 10;
    }

    public static void cargar_arreglo(int[] arr) {
          for (int pos = 0 ; pos < MAX; pos++)
                    arr[pos]=pos*2;
    }
}
```

Variable	Dirección	Valor
B	132	205
B[0]	205	0
B[1]	206	2
B[2]	207	4
B[3]	208	6
B[4]	209	8
a	301	20
c	**510**	**10**

Ejemplo

```
public class Programa {
    final static int MAX = 5;
    public static void main(String args[]) {
         //MAS ADELANTE SE DEBE MODULARIZAR LA INICIALIZACION E IMPRESION
          int B[]=new int[MAX];
          int a = 20;
          cargar_arreglo(B);
          System.out.println("Los datos son:");
          for (int pos = 0 ; pos < MAX; pos++)
                    System.out.println(B[pos]);
          cargar_variable_simple(a);
          System.out.println("La variable es:");
          System.out.println(a);
    }

    public static void cargar_variable_simple(int c) {
          c = 10;
    }

    public static void cargar_arreglo(int[] arr) {
          for (int pos = 0 ; pos < MAX; pos++)
                    arr[pos]=pos*2;
    }
}
```

Variable	Dirección	Valor
B	132	205
B[0]	205	0
B[1]	206	2
B[2]	207	4
B[3]	208	6
B[4]	209	8
a	**301**	**20**

Como trabajar con arreglos

- Para hacer un programa que contiene arreglos se tendrá en cuenta el siguiente esquema:

- Definir constantes;
- Dentro de main(){
  - Definir variables
  - Inicializar arreglo/s;
  - Cargar arreglo/s;
  - Procesar;
  - Imprimir arreglos/s;

}

Estructuras ordenadas

- Las estructuras ordenadas se refiere a una propiedad que posee su composición respecto de sus elementos.
- Los métodos de consulta y modificación tendrán que considerar las propiedades que posee la estructura.

```
public static int buscar_pos_des(int[] arr,int valor) {
      int pos = 0;
      while ((pos<MAX)&&(arr[pos]!=valor))
            pos++;
      if (pos<MAX) return pos;
      else return -1;
}

public static int buscar_pos_ord(int[] arr,int valor) {
      int pos = 0;
      while ((pos<MAX)&&(arr[pos]>valor))
            pos++;
      if ((pos<MAX)&&(arr[pos]==valor)) return pos;
      else return -1;
}
```

Ejercicios

En el caso de hacer corrimientos pueden quedar elementos repetidos

1. Hacer un programa que inicialice y luego cargue un arreglo (sin orden) de tamaño MAX=10 con números enteros ingresados por teclado. Finalmente imprima el arreglo por pantalla.
2. Implementar un método que realice un corrimiento a derecha en un arreglo ordenado de tamaño MAX=10 a partir de una posición.
3. Implementar un método que realice un corrimiento a izquierda en un arreglo ordenado de tamaño MAX=10 a partir de una posición.
4. Hacer un programa que inserte un elemento en un arreglo (ordenado decrecientemente) de tamaño MAX=10.
5. Hacer un programa que elimine un elemento en un arreglo (ordenado decrecientemente) de tamaño MAX=10.
6. Hacer un programa que elimine los valores pares en un arreglo de tamaño MAX=10.
7. Hacer un programa que invierta el orden de los elementos de un arreglo de tamaño MAX=10.

Ejercicios

Dado un arreglo de tamaño de arreglo MAX=20 de secuencias. En el caso de eliminar secuencias, hacer corrimientos y completar con 0 al final.

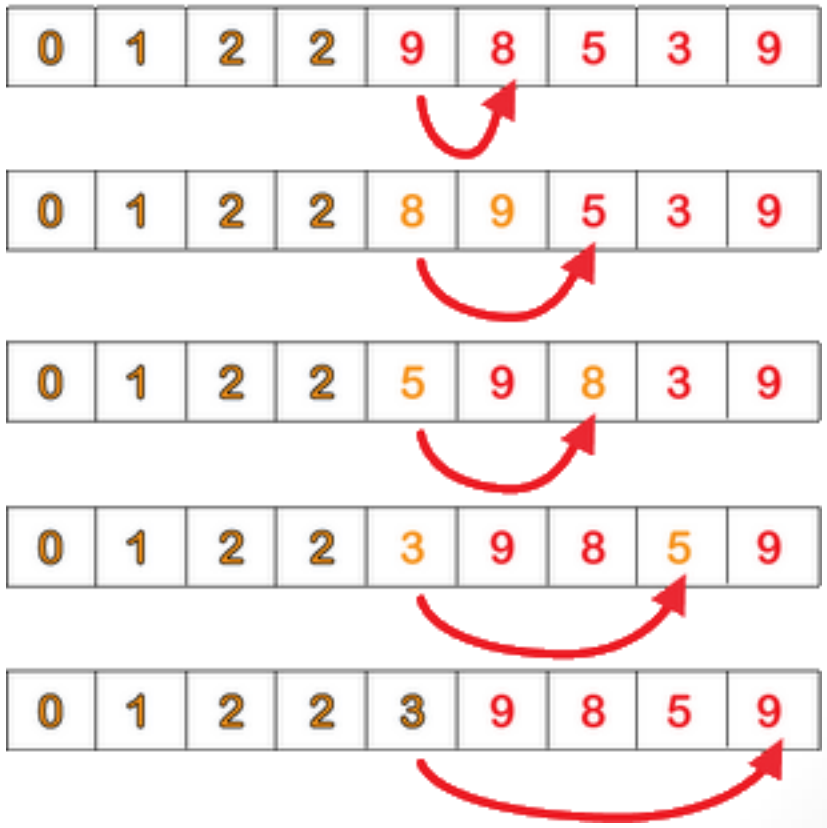
8. Hacer un programa que devuelva la posición de inicio y fin de la primer secuencia de números distinta de ceros.
9. Hacer un programa que devuelva la posición de inicio y fin de la secuencia de números distintos de ceros cuya suma del contenido sea mayor.
10. Hacer un programa que devuelva la posición de inicio y fin de la anteúltima secuencia de números distintos de ceros.
11. Hacer un programa que dado un número N ingresado por el usuario, elimine las secuencias de tamaño N de números distintos de cero.
12. Hacer un programa que elimine de un arreglo todas las ocurrencias de una secuencia patrón dada por otro arreglo.
13. Hacer un programa que elimine de un arreglo todas las secuencias que tienen orden descendente entre sus elementos.

Ejercicios

14. Hacer un programa que reemplace de un arreglo A todas las ocurrencias de una secuencia patrón dada en un arreglo P, por la secuencia contenida en el arreglo R.
15. Hacer un programa que invierta el orden de la última secuencia en un arreglo.
16. Hacer un programa que extraiga todas las secuencias con cantidad par de elementos de un arreglo A y las copie en un arreglo P, quedando separadas por un 0.
17. Considerando un arreglo A que tiene un orden ascendente según la cantidad de elementos que poseen las secuencias, insertar en A manteniendo su orden una secuencia dada en un arreglo P.

Métodos de ordenamiento

- Los algoritmos de ordenamiento permite ordenar sus elementos. Los métodos más populares son:
  - Selección
  - Inserción
  - Burbujeo



Ejemplo

```
public static void seleccion(int arr[]) {
  int i, j, menor, pos, tmp;
  for (i = 0; i < MAX; i++) { // tomamos como menor el primero
       menor = arr[i]; // de los elementos que quedan por ordenar
       pos = i; // y guardamos su posición
       for (j = i + 1; j < MAX; j++){ // buscamos en el resto
              if (arr[j] < menor) { // del array algún elemento
                     menor = arr[j]; // menor que el actual
                     pos = j;
              }
       }
       if (pos != i){ // si hay alguno menor se intercambia
              tmp = arr[i];
              arr[i] = arr[pos];
              arr[pos] = tmp;
       }
  }
}
```

Ejemplo

```
public static void insercion(int arr[]) {
   for (int i = 1; i < MAX; i++) {
         int aux = arr[i];
         int j = i - 1;
         while ((j >= 0) && (arr[j] > aux)){
                  arr[j+1] = arr[j];
                  j--;
         }
         arr[j+1] = aux;
   }
}
public static void burbujeo(int[] arr){
   int temp;
   for(int i = 1;i < MAX;i++){
         for (int j = 0 ; j < MAX - 1; j++){
                  if (arr[j] > arr[j+1]){
                           temp = arr[j];
                           arr[j] = arr[j+1];
                           arr[j+1] = temp;
                  }
         }
   }
}
```

Ejercicios

1. Hacer un programa que cargue un arreglo (sin orden) de tamaño MAX=10 con números enteros aleatorios entre 0 y 100. Finalmente imprima por pantalla el arreglo ordenado.
2. Modularizar el método de ordenamiento por selección, considerando los métodos

   obtener_posicion_menor_arreglo(int[] arr, int pos_ini, int pos_fin)

   intercambiar_contenido_arreglo(int[] arr, int pos_1, int pos_2)