Centro de Investigación en Computación del IPN

Java - curso básico

Dra. Erandi Castillo Montiel

Arreglos

Un arreglo o Array, es un objeto en el que se pueden agrupar un número de objetos o datos primitivos , del mismo tipo.
Cada uno de los objetos dentro el array, tiene un identificador numérico único asignado dependiendo de sus posición en el arreglo.

Sintaxis:

tipo [] variable_arreglo;
tipo variable_arreglo[];

Ejemplo:

int [] arrI;
String [] arrS;
float arrf[];

Dimensionando un arreglo

Consiste en crear un arreglo con el tamaño que tendrá

variable_array= new tipo[tamaño];

*Los indices de un arreglo se encuentran entre 0 y tamaño-1

arrI.length;

```
arrI = new int [10];
arrS = new String[5];
arrf= new float[10];
```

```
int numeros [] = new int [10];


for (int i =0; i<numeros.length; i++){
    numeros[i]=i*2;
    System.out.println(numeros[i]);
}
```

Array como argumento

Sintaxis:

Como argumento:

```
tipo_retorno nombreMetodo (tipo []arreglo){
//sentencias
}
```

Retorno de un arreglo:

```
tipo [ ] nombreMetodo (){
return arreglo;
}
```

```java
public static void main(String[] args) {
    int[] arreglo = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
    imprimeNumeros(arreglo);
    cambiaNumeros(arreglo);
    imprimeNumeros(arreglo);

}

public static void cambiaNumeros(int[] array) {
    for (int i = 0; i < array.length; i++) {
        array[i] = array[i] * 2;

    }
}

public static void imprimeNumeros(int[] array) {
    for (int i = 0; i < array.length; i++) {

        System.out.print("valor: " + array[i] + " ");
    }
    System.out.println("");
}
```

Ejercicio

Cree un programa que genere un arreglo de 10 elementos.

Inicialice los elementos con un número aleatorio.

Sume el contenido del arreglo.

Imprima el mayor y el menor de todos los números dentro del arreglo.

```java
public static void main(String[] args) {

    int [] arregloAleatorio;

    arregloAleatorio= generaArreglo();
    menorMayorArreglo(arregloAleatorio);
    // TODO code application logic here
}


public static int [] generaArreglo (){

    Random grand = new Random();
    int randomNumber;
    int [] arreglo = new int[10];
    for(int i =0;i<10;i++){
    randomNumber= grand.nextInt(10) + 1;
    arreglo[i]=randomNumber;
        System.out.println("arreglo ["+i+"] : " + arreglo[i]);
    }
return arreglo;
}
```

Sentencia *for each*

Sintaxis:

```
for (tipo variable: array){
        //sentencias
}
```

Ejemplo

```
for (int n: arrayI){
    System.out.println(n);
}
```

Ordenamiento de burbuja

```
metodoBurbuja (a0,a1,a2, …,a(n-1))
        para i=0 hasta n hacer
            para j=0 hasta n-1 hacer
                si a(j)>a(j+1) entonces
                  cambiarlos de lugar
                fin sin
            fin for
        fin for
fin método
```

Arreglo bidimensionales

- Sintaxis:

  tipo [][] variable;

- Ejemplo

  int [][] arreglo2;

  arreglo2 = new int [3][4];

Recorrido

```
 */
public static void main(String[] args) {

    int[][] arregloBi = {{1, 2, 3, 4, 5, 6, 7, 8, 9, 10},
    {2, 4, 6, 8, 10, 12, 14, 16, 18, 20}};
    System.out.println("Filas= " + arregloBi.length);
    System.out.println("Columnas= " + arregloBi[0].length);
    imprimeNumerosBi(arregloBi);
    cambiaNumerosBi(arregloBi);
    imprimeNumerosBi(arregloBi);

}
```

```
public static void cambiaNumerosBi(int[][] array) {
    for (int i = 0; i < array.length; i++) {
        for (int j = 0; j < array[i].length; j++) {

            array[i][j] = array[i][j] * 2;
        }

    }
}

public static void imprimeNumerosBi(int[][] array) {
    for (int[] i : array) {
        for (int j : i) {
            System.out.print("valor: " + j);
        }
        System.out.println("");
    }
    System.out.println("");
}
```

Empaquetado de clases

- Consiste en definir el paquete al cual pertenece la clase.
- Generalmente el nombre del paquete coincide con el nombre del directorio que contiene el archivo .java

Sintaxis

```
package  nombredelpaquete;
```

Consideraciones: la sentencia package es la primer sentencia dentro del archivo .java

Va definida en el import

Afecta a todas las clases dentro del archivo

Modificadores de acceso

- private: Aplicable a atributos y a métodos. Su uso está restringido al interior de la clase.
- (default): Si un atributo o método no define un modificador de acceso, se dice que su acceso es (por default), sólo las clases del mismo paquete tendrá acceso a estos elementos.
- protected: los elementos protected sólo puede ser usado por clases del mismo paquete o por una subclase.
- public: máximo nivel de visibilidad, todos pueden hacer uso de los elementos públicos

Encapsulamiento

```java
package figuras;

public class Rectangulo {
    public int ancho;
    public int alto;

    public int area(){
        return ancho * alto;
    }
}
```

```java
package cbasico;

import figuras.Rectangulo;

public class Cbasico {
    public static void main(String[] args) {
        // TODO code application logic here
        Rectangulo rec = new Rectangulo();
        System.out.println(rec.area());

        rec.alto = 5;
        rec.ancho = 6;
        System.out.println(rec.area());
        rec.alto = 5;
        rec.ancho = -6;
        System.out.println(rec.area());
    }
}
```

Encapsulamiento 2

```java
package figuras;

public class Rectangulo {
    private int ancho;
    private int alto;

    public int area(){
        return ancho * getAlto();
    }

    public void setAncho(int ancho){
        if(ancho>0)
            this.ancho = ancho;
    }

    public int getAncho(){
        return ancho;
    }

    public int getAlto() {
        return alto;
    }

    public void setAlto(int alto) {
        if(alto>0)
            this.alto = alto;
    }
}
```

Sobrecarga de métodos

- Un constructor es un método especial que sirve, generalmente para realizar las tareas que deban realizarse en el momento de crear un objeto. Se invoca al llamar al operador new.
- Características:
  - El nombre del constructor debe ser el mismo que el de la clase.
  - El constructor NO DEBE TENER UN TIPO DE RETORNO.
  - Toda clase debe tener amenos un constructor.
  - Los constructores se pueden sobrecargar,

- Sintaxis:

```
public NombreClase(){

}
```

```
public NombreClase(argumentos){


}
```

```java
public class SobreCarga {
    public void imprime(){
        System.out.println("Cadena por default...");
    }
    public void imprime(String s){
        System.out.println("Valor asignado : "+s);
    }
}
```

```java
public class Cbasico {
    public static void main(String[] args) {
        // TODO code application logic here
        SobreCarga sc = new SobreCarga();

        sc.imprime();
        sc.imprime("Algo");
    }
}
```

Ejercicio

- Defina una clase que implemente un Punto refiriéndose a sus coordenadas cartesianas.
- Implemente el constructor por defecto.
- Implemente el constructor que reciba como argumentos las coordenadas cartesianas.
- Los atributos deben ser enteros.
- Implemente un método que imprima las coordenadas cartesianas con el siguiente formato: “las coordenadas son (x,y)”.

```java
public class Punto{
    private int x;
    private int y;

    Punto(){
    }

    Punto(int x, int y){
        setX(x);
        setY(y);
    }

    public int getX(){
        return x;
    }
    public int getY(){
        return y
    }
    public void setX(int x){
        this.x = x;
    }
    public void setY(int y){
        this.y = y;
    }
}
```

Proyecto 1

- Desarrollar una aplicación que simule a un cajero automático, se debe crear una clase llamada Cuenta, que gestione las operaciones sobre una cuenta bancaria, con los siguientes métodos:

  - void deposito(float c) : Incrementa el saldo de la cuenta.
  - void retiro(float c) : Descuenta saldo de la cuenta.
  - float saldo(): regresa el saldo de la cuenta.

- Por otro lado desarrollar una clase que implemente el main, deberá presentar un menú con las siguientes opciones.

  -crear cuenta vacía.
  -crear cuenta con saldo
  -Realizar un depósito.
  -Realizar un retiro.
  -ver saldo.
  -Salir.

El menú se debe presentar hasta que se presione la opción salir.