



Argentina
programa
4.0

Funciones

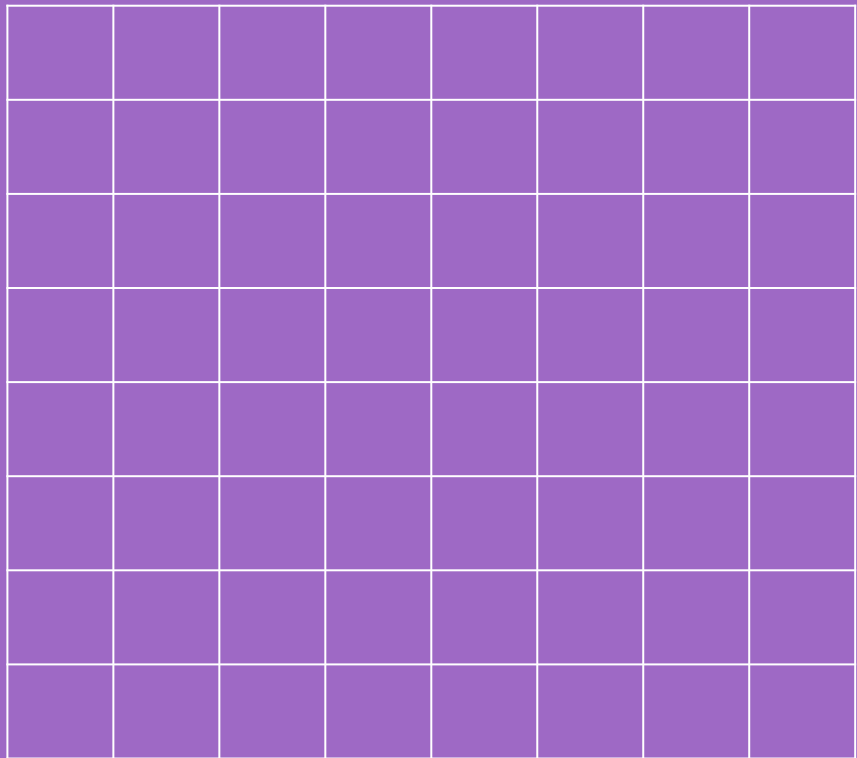
“Desarrollador Java Inicial”

Agenda

- Funciones
 - Definición general
 - Declaración y uso en Java
 - Ámbito
 - Parámetros y retornos
 - Ejercicios

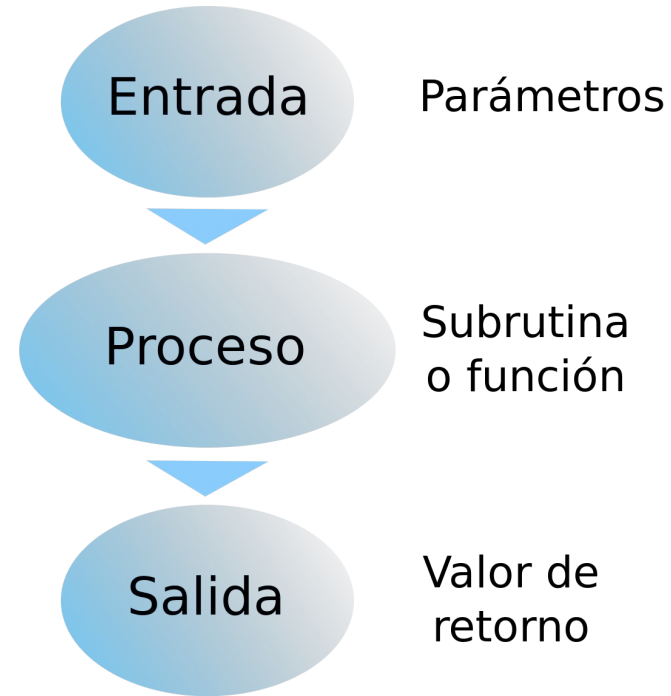


Funciones / Métodos



FUNCIONES

- Es un bloque o sección de código que sirve para ejecutar una tarea específica y puede ser utilizado desde cualquier parte con acceso a ese bloque.
- Puede o no tener parámetros de entrada y puede o no tener un valor de retorno.

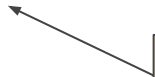


Propósito

- Modularización: Implementar soluciones más pequeñas que resulten más fáciles de manejar y controlar.
- Reutilización: Volver a utilizar la solución en varias ocasiones, logrando el ahorro de líneas de código y la disminución de errores.

Ejemplo:

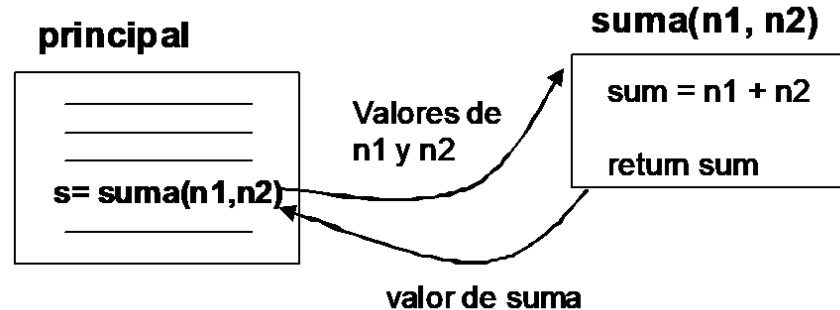
```
System.out.println("un texto");
```



La función **println** se puede reutilizar desde distintos lugares.

FUNCIONES

- Una función tiene un nombre o identificador.
- Con ese identificador la función puede ser activada todas las veces que sea necesario.



FUNCIONES EN JAVA

Sintaxis

Nota: *todo lo que está entre corchetes es opcional*

```
[acceso] [modificador] tipo nombreFuncion([tipo nombreArgumento,[tipo
nombreArgumento]...])
{
    /* Bloque de instrucciones */

    [return valor;]
}
```

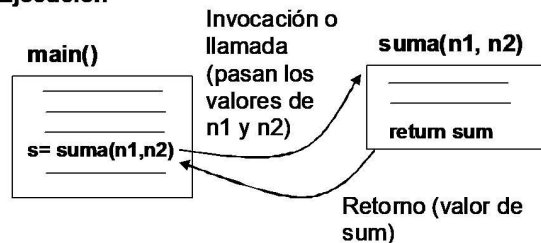
Ejemplo:

```
public static void main(String[] args) //void: sin tipo de retorno
{
    . . .
}
```

FUNCIONES EN JAVA

DEFINICIÓN	INVOCACIÓN (LLAMADA)
<pre>int suma(int n1, int n2) { int sum = n1 + n2; return sum; }</pre>	<pre>public static void main(String[] args) { Scanner sc = new Scanner(System.in); int n1, n2, s; System.out.println("Ingrese un número:") n1 = sc.nextInt(); System.out.println("Ingrese un número:") n2 = sc.nextInt(); // llama a la función suma s = suma(n1, n2); System.out.println(" La suma es :" + s) }</pre>

Ejecución

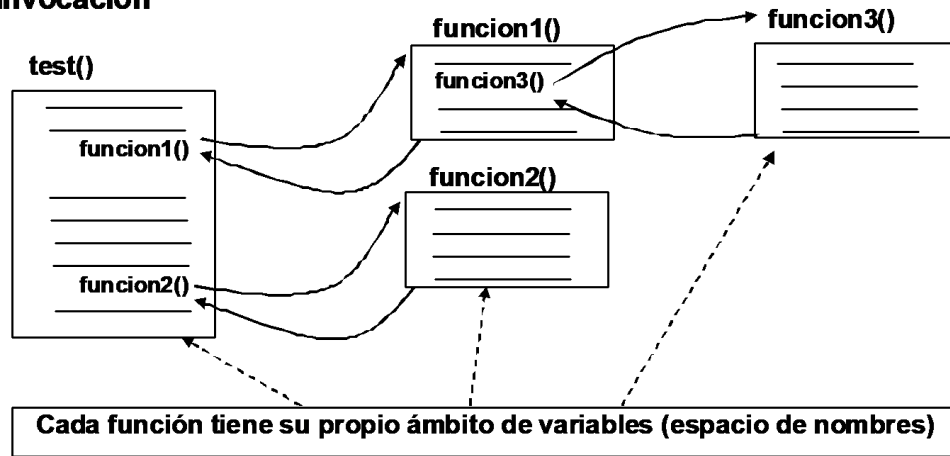


FUNCIONES: ÁMBITO

Ámbito de una función

- Región de un programa donde una variable es reconocida y utilizable
- Si una variable es local a un ámbito, sólo puede usarse dentro de ese ámbito.

Invocación



FUNCIONES: ÁMBITO

main()

```
int n1 = 20;  
int n2 = 10;  
  
int may = mayor(n1, n2);  
System.out.println("Mayor: "+may);
```

mayor(int n1, int n2)

```
int may;  
if (n1 > n2)  
    may = n1;  
else:  
    may = n2;  
return may;
```

20, 10

20

```
int n1 = 20  
int n2 = 10  
int may = ?
```

```
int n1 = 20  
int n2 = 10  
int may = 20
```

Cada función tiene su propio ámbito de variables (espacio de nombres)

FUNCIONES: PARÁMETROS

Parámetro es una variable cuyo valor se envía a una función para que ésta eventualmente lo procese.



- **Parámetros formales:** Son las variables que se declaran en la cabecera de una función.

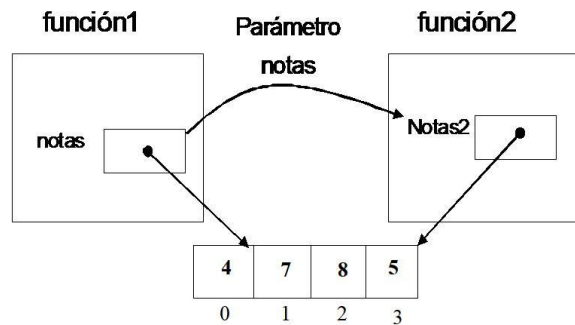
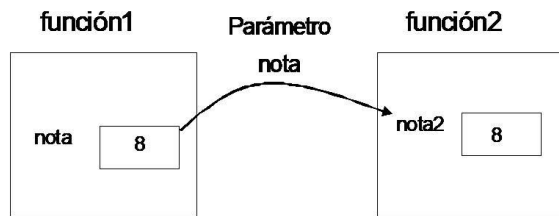
```
int suma(int n1, int n2)
```

- **Parámetros actuales:** Son los valores o las variables que se escriben entre los paréntesis de una función al invocarla.

```
s = suma(n1, n2);
```

FUNCIONES: PARÁMETROS

- **Parámetros por copia:** Se crea una copia de la variable que se está pasando por parámetro al ámbito de la función invocada, los valores que se cambien dentro de la función no afectan a los originales.
- **Parámetros por referencia:** Se crea una copia de la referencia o dirección a donde están los datos originales, por ejemplo un vector.



FUNCIONES: RETORNOS

- **Con retorno de valor (funciones):** son aquellas funciones que devuelven un valor como resultado de la acción que realizan, de forma tal que ese valor puede volver a usarse en alguna otra operación.

```
s = suma(n1, n2);    //int suma(int n1, int n2)  
n1 = sc.nextInt();
```

- **Sin retorno de valor (procedimientos):** son aquellas funciones que realizan alguna acción pero no se espera que retornen valor alguno como resultado de la misma. Son de tipo void y generalmente el desarrollador no escribe la sentencia return.

```
System.out.println("Ingrese un número:");
```

FUNCIONES: RETORNOS

Consideraciones con la sentencia return:

- Cualquier instrucción que se encuentre después de la ejecución de return **NO** será ejecutada. Es común encontrar funciones con múltiples sentencias return al interior de condicionales, pero una vez que el código ejecuta una sentencia return lo que haya de allí hacia abajo no se ejecutará.
- El tipo del valor que se retorna en una función **debe coincidir** con el del tipo declarado a la función, es decir si se declara int, el valor retornado debe ser un número entero.
- En el caso de los procedimientos (void) podemos usar la sentencia return pero **sin ningún tipo de valor**, sólo se usa para formalizar la terminación de la ejecución del procedimiento.

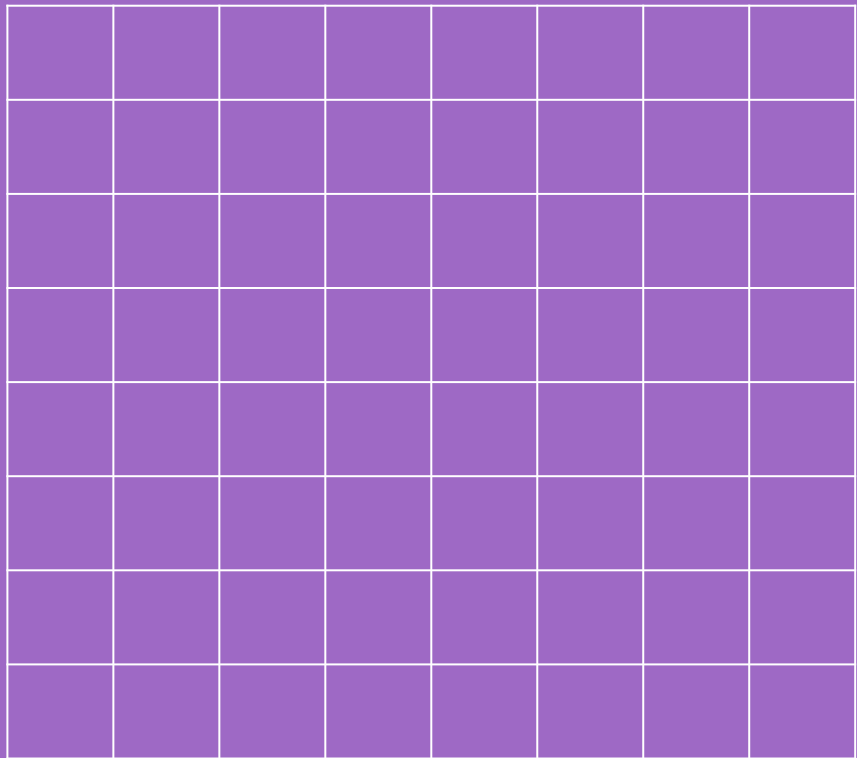
Parámetros de comando

Cuando se inicia un programa cualquiera, al mismo se le pueden pasar parámetros. En el caso de Java, si al programa se le pasan, este los leerá desde la variable **args** en **main**. Los parámetros pueden ser cualquier string y en cualquier cantidad.

```
                                tipo devuelto    parámetro: vector de String
                                ↙                ↘
public class Ejemplo {
    public static void main(String[] args) {
        int resultado = 0;
        for (int i = 0; i < args.length; i++) {
            int numero = Integer.parseInt(args[i]);
            resultado = resultado + numero;
        }
        System.out.println(resultado);
    }
}
```



Ejercicios



EJERCICIOS

1. Ingresar la cantidad de estudiantes de un curso y la cantidad de estudiantes que aprobaron el mismo, calcular y mostrar el porcentaje que representan estos últimos respecto del total de estudiantes.

```
9 public class Ejercicio4_1 {
10     public static void main(String[] args) {
11         Scanner sc = new Scanner(System.in);
12         // variables de entrada
13         int cantTotal, cantAprobados;
14         float porcentaje; //porcentaje = parcial * 100 / total
15
16         System.out.print("Ingrese la cantidad total de estudiantes: ");
17         cantTotal = sc.nextInt();
18
19         System.out.print("Ingrese la cantidad de aprobados: ");
20         cantAprobados = sc.nextInt();
21
22         // llama a la función que carga el porcentaje .
23         porcentaje = calcularPorcentaje(cantTotal, cantAprobados);
24         System.out.println("El porcentaje de aprobados es: " + porcentaje);
25     }
```

```
28 // calcula el porcentaje .
29 public static float calcularPorcentaje(int total, int parcial) {
30     //porcentaje = parcial * 100 / total
31     float porcentaje = 0;
32
33     // calculo el porcentaje
34     if (total != 0)
35         porcentaje = parcial * 100 / total;
36
37     return porcentaje;
38 }
39
40 }
```

EJERCICIOS

2. Dado un vector de números enteros, retornar el promedio de los números.

Planteado sin funciones:

```
public class Ejemplo4_2 {  
    public static void main(String[] args) {  
        int numeros[] = new int[] { 1, 37, 16 };  
        int suma = 0, promedio = 0;  
  
        for (int numero : numeros) {  
            suma = suma + numero;  
        }  
  
        promedio = suma / numeros.length;  
  
        System.out.println("El promedio de los  
números es: " + promedio);  
    }  
}
```

Planteado con funciones:

```
public class Ejercicio4_2 {  
    public static void main(String[] args) {  
        int numeros[] = new int[] { 1, 37, 16 };  
        int suma = sumatoria(numeros);  
        int promedio;  
        promedio = suma / numeros.length;  
        System.out.println("El promedio de los  
números es: " + promedio);  
    }  
  
    private static int sumatoria(int[] numeros) {  
        int suma = 0;  
        for (int numero : numeros) {  
            suma = suma + numero;  
        }  
        return suma;  
    }  
}
```

3. Usando el ejercicio 2 de la clase anterior, lo vamos a plantear con funciones:

Cargar por teclado una lista de números positivos, de tamaño n y mostrar:

1. Los números de la lista.
2. La sumatoria de los números.
3. La cantidad de números mayores a 10.
4. El porcentaje de números pares.
5. El promedio de todos los números.

EJERCICIOS

Resolución sin modularización o funciones adicionales, código lineal y secuencial

```
1 package clase3;
2 import java.util.Scanner;
3
4 /** 2. Cargar por teclado una lista de números positivos, de tamaño n y mostrar:
5 1. Los números de la lista.
6 2. La sumatoria de los números.
7 3. La cantidad de números mayores a 10.
8 4. El porcentaje de números pares.
9 5. El promedio de todos los números.
10 */
11 public class Ejercicio3_2 {
12     public static void main(String[] args) {
13         Scanner sc = new Scanner(System.in);
14         int n, num;
15         int vec[];
16         System.out.print("Ingrese el valor de n: ");
17         n = sc.nextInt();
18         // crea el vector de n elementos
19         vec = new int[n];
20         // cargar el vector
21         for (int i = 0; i < vec.length; i++)
22         {
23             //resultados parciales
24             System.out.print("Ingrese el "+ i+ " valor: ");
25             num = sc.nextInt();
26             // guarda el num en el vector en la posición i
27             vec[i] = num;
28         }
```

```
30 // 1. mostrar el vector
31 String lista = "";
32 for (int numero : vec)
33 {
34     // guarda los valores en lista
35     lista += "\n"+numero;
36 }
37 System.out.println("Los valores del vector son: "+lista);
```

```
39 // 2. La sumatoria de los números.
40 int suma = 0;
41 for (int numero : vec)
42 {
43     suma = suma + numero;
44 }
45 System.out.println("La suma es: "+suma);
46
```

EJERCICIOS

3. Resolución con modularización usando funciones simples

Declaración de variables y vector

```
12 public class Ejercicio4_3 {
13     public static void main(String[] args) {
14         // definición de variables y entrada de datos
15         Scanner sc = new Scanner(System.in);
16         // variables de entrada
17         int n;
18         int vec[];
19         // variables de salida
20         String lista;
21         int suma;
22         int sumaMayor10;
23         float porcentaje;           //porcentaje = parcial * 100 / total
24         float promedio;             // promedio = sumatoria / cantidad
25
26         System.out.print("Ingrese el valor de n: ");
27         n = sc.nextInt();
28         // crea el vector de n elementos
29         vec = new int[n];
```

EJERCICIOS

3. Resolución con modularización usando funciones simples

Invocación de funciones y visualización de resultados

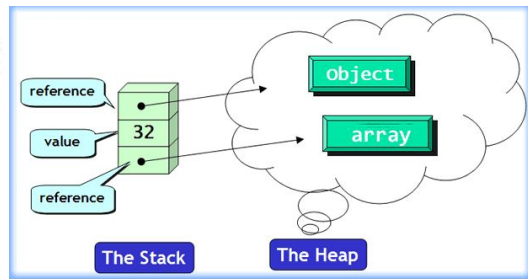
```
31 // llama a las funciones para manejar el vector
32
33 // cargar el vector
34 cargar(vec);
35 // 1. mostrar el vector
36 lista = mostrar(vec);
37 System.out.println("Los valores del vector son: "+lista);
38 // 2. La sumatoria de los números.
39 suma = sumar(vec);
40 System.out.println("La suma es: "+suma);
41 // 3. La cantidad de números mayores a 10.
42 sumaMayor10 = sumarMayores10(vec);
43 System.out.println("La cantidad valores mayores a 10 es: "+sumaMayor10);
44 // 4. El porcentaje de números pares.
45 porcentaje = calcularPorcentaje(vec);
46 System.out.println("El porcentaje de valores pares es: "+porcentaje);
47 // 5. El promedio de todos los números.
48 //el total es igual a la cantidad de elementos del vector
49 promedio = calcularPromedio(suma, vec.length);
50 System.out.println("El promedio de valores es: "+promedio);
51 }
```

EJERCICIOS

3. Resolución con modularización usando funciones simples

Declaración de funciones: cargar el vector

```
53 // cargar el vector
54 // pasa la referencia del vector por eso no retorna,
55 //trabaja con los datos del vector original
56 public static void cargar(int[] vec) {
57     Scanner sc = new Scanner(System.in);
58     int num;
59     for (int i = 0; i < vec.length; i++)
60     {
61         //resultados parciales
62         System.out.print("Ingrese el "+ i+ " valor: ");
63         num = sc.nextInt();
64         // guarda el num en el vector en la posición i
65         vec[i] = num;
66     }
67 }
```



EJERCICIOS

3. Resolución con modularización usando funciones simples

Declaración de funciones: mostrar y sumar

```
69 // 1. mostrar el vector
70 public static String mostrar(int[] vec) {
71     String lista = "";
72     for (int numero : vec)
73     {
74         // guarda los valores en lista
75         lista += "\n"+numero;
76     }
77     return lista;
78 }
```

```
80 // 2. La sumatoria de los números.
81 public static int sumar(int[] vec) {
82     int suma = 0;
83     for (int numero : vec)
84     {
85         suma = suma + numero;
86         //suma += numero;
87     }
88     return suma;
89 }
```


EJERCICIOS

4. Se tienen los datos de tres postulantes a un empleo, a los que se les realizó un test para conocer el nivel de formación previa de cada uno. Por cada postulante, se tienen los siguientes datos: nombre del postulante, cantidad total de preguntas que se le realizaron y cantidad de preguntas que contestó correctamente. Se pide confeccionar un programa que lea los datos de los tres postulantes, informe el nivel de formación previa de cada uno según los criterios de aprobación que se indican más abajo, e indique finalmente el nombre del postulante que ganó el puesto. Los criterios de aprobación son en función del porcentaje de respuestas correctas sobre el total de preguntas realizadas a cada postulante:

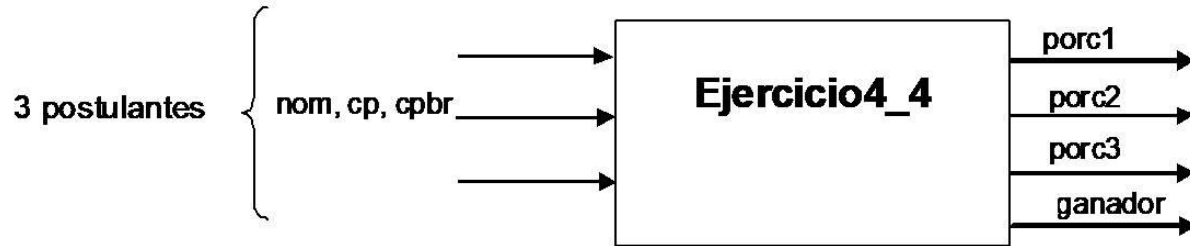
Nivel Superior:	Porcentaje $\geq 90\%$
Nivel Medio:	$75\% \leq \text{Porcentaje} < 90\%$
Nivel Regular:	$50\% \leq \text{Porcentaje} < 75\%$
Fuera de Nivel:	Porcentaje $< 50\%$

Aclaración: Si ningún postulante superó el 50% se considera que ninguno ganó el puesto.

EJERCICIOS

4. Análisis y solución:

- Datos: 3 postulantes, c/u: nombre, cantidad de preguntas totales, cantidad de preguntas respondidas correctamente. (nom, tp, cpbr)
- Resultados:
 - Nivel de formación de cada postulante (porc1, porc2, porc3).
 - El ganador (ganador).



EJERCICIOS

4. Análisis y solución:

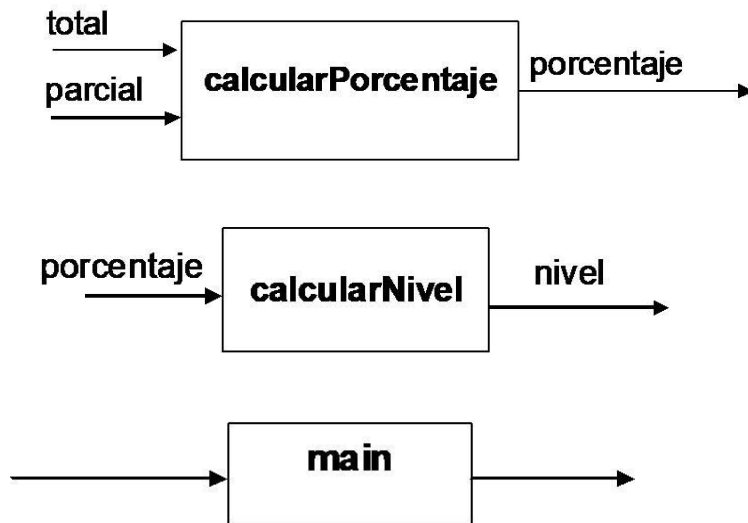
Subproblemas

1. Calcular el porcentaje

2. Obtener el nivel

3. Función principal

Funciones



EJERCICIOS

4. Implementación del cálculo del porcentaje:

```
// calcula el porcentaje .  
public static float calcularPorcentaje(int total, int parcial) {  
    //porcentaje = parcial * 100 / total  
    float porcentaje = 0;  
  
    // calculo el porcentaje  
    if (total != 0)  
        porcentaje = parcial * 100 / total;  
  
    return porcentaje;  
}
```



EJERCICIOS

4. Implementación del cálculo del nivel:

```
// calcula el nivel.  
public static String calcularNivel(float porcentaje) {  
    String nivel;  
  
    if (porcentaje >= 90)  
        nivel = "Superior";  
    else  
        if (porcentaje >= 75)  
            nivel = "Medio";  
        else  
            if (porcentaje >= 50)  
                nivel = "Regular";  
            else  
                nivel = "Fuera de Nivel";  
  
    return nivel;  
}
```



EJERCICIOS

4. Implementación de la función principal:

Declaración de variables

```
public class Ejercicio4_4 {  
    public static void main(String[] args) {  
        //para ingresar solo enteros  
        Scanner entero = new Scanner(System.in);  
        // para ingresar solo strings  
        Scanner string = new Scanner(System.in);  
        // variables de entrada  
        String nom;  
        int totalPreguntas, totalBienRespondidas;  
        float porcentaje;  
        String nivel, listado = "";  
        String postulanteMejor = "";  
        float porcentajeMejor = 0;  
    }  
}
```



EJERCICIOS

4. Implementación de la función principal:

Carga de los postulantes

```
// cargar los 3 postulantes
for (int i = 0 ; i < 3; i++) {
    System.out.println("Postulante " + (i+1) + "º");
    System.out.print("Ingrese el nombre: ");
    nom = string.nextLine();
    System.out.print("Ingrese el total de preguntas: ");
    totalPreguntas = entero.nextInt();
    System.out.print("Ingrese el total de preguntas bien respondidas: ");
    totalBienRespondidas = entero.nextInt();
    System.out.println();

    // llama a la función que calcula el porcentaje .
    porcentaje = calcularPorcentaje(totalPreguntas, totalBienRespondidas);
    // llama a la función que calcula el nivel
    nivel = calcularNivel(porcentaje);

    listado += "\n Postulante " + (i+1) + ": Nombre: " + nom + "- Nivel: " +
        nivel+ "- Porc.: " + porcentaje+ "%";
    // determinación del aspirante con mayor porcentaje...
    if (porcentaje > porcentajeMejor) {
        postulanteMejor = nom;
        porcentajeMejor = porcentaje;
    }
} // fin del ciclo
```



EJERCICIOS

4. Implementación de la función principal:

Resultados



```
// resultados
System.out.println("Los postulantes : " + listado);
if (porcentajeMejor > 50)
    System.out.println("Ganador:" + postulanteMejor + "- con porcentaje de:" + porcentajeMejor + "%");
else
    System.out.println("No hay ganador: todos tienen porcentaje menor al 50%");
}
```




**Argentina
programa
4.0**

Gracias!
