G-TeC
Facultad de Informática
Universidad Complutense de Madrid

Ejercicios de evaluación de fundamentos de programación en Java

Jorge Martínez Ladrón de Guevara

Editorial EME

ISBN 978-84-96285-40-8

Contenido

1. Introducción a Java

Test de evaluación

1. ¿Qué define la estructura de las expresiones de un lenguaje de programación?

   a) Sus palabras reservadas

   b) Sus reglas sintácticas

   c) Sus reglas semánticas

2. ¿En qué consiste el proceso de programación?

   a) Escritura, compilación y verificación del código fuente de un programa

   b) Compilación del código fuente de un programa

   c) Compilación y verificación del código Bytecode de un programa

3. Un algoritmo es:

   a) Un conjunto ordenado de operaciones que permite hallar la solución de un problema

   b) Un conjunto ordenado y finito de operaciones que permite hallar la solución de un problema

   c) Un conjunto aleatorio y finito de operaciones que permite hallar la solución de un problema

4. Un programa Java compilado es portable porque:

   a) El código Bytecode es ejecutable por los principales sistemas operativos del mercado

   b) El entorno de ejecución de Java incluye una máquina virtual que interpreta el código Bytecode

   c) El entorno de ejecución interpreta el código Java, independientemente de la máquina virtual

5. Java es:

   a) Un lenguaje de programación exclusivamente

   b) Un lenguaje de programación Java, una plataforma de desarrollo, un entorno de ejecución y un conjunto de librerías para desarrollo de programas sofisticados

   c) Un sistema para ejecutar programas en distintas plataformas informáticas

6. El entorno de ejecución de Java (Java Runtime Environment)

   a) Es un conjunto de librerías para desarrollo de aplicaciones Java

   b) Es una pieza intermedia entre el código Bytecode y los distintos sistemas operativos existentes en el mercado. Incluye la máquina virtual de Java

   c) Es la máquina virtual de Java

7. El proceso clásico de desarrollo de software se compone de las siguientes fases:

   a) Codificación, diseño, pruebas y validación

   b) Especificación, diseño, codificación, prueba y mantenimiento

   c) Diseño, pruebas, validación y mantenimiento

8. Durante la fase de Especificación de una aplicación se define:

   a) La funcionalidad, las características técnicas de una aplicación y sus condiciones de uso

   b) La funcionalidad y las características técnicas de una aplicación

   c) Las características técnicas y las condiciones de uso de la aplicación

9. El compilador de Java analiza el código fuente y:

   a) Comprueba que todos sus elementos son palabras válidas en Java y su semántica

   b) Comprueba que todos sus elementos son palabras válidas en Java

   c) Comprueba que todos sus elementos son palabras válidas en Java, verifica la estructura sintáctica del programa y su semántica

10. La legibilidad de un programa Java es importante porque:

   a) Facilita el mantenimiento del software y permite corregir errores o modificar la funcionalidad con menor coste

   b) Evita errores del compilador

   c) Permite corregir errores, aunque no facilita el proceso de mantenimiento de una aplicación

Ejercicios

1. ¿Cuáles son los elementos de un lenguaje de programación?

2. Explique por qué los programas Java son portables.

3. Explique brevemente las acciones que realiza el compilador de Java durante el proceso de compilación.

4. Explique qué se entiende por legibilidad, corrección y eficiencia.

5. Indique las fases del proceso clásico de desarrollo de software y explique brevemente cada fase.

2. Estructura de un programa Java

Test de evaluación

1. Java es un lenguaje que distingue letras mayúsculas y minúsculas.

   a) Falso

   b) Verdadero

2. El nombre de un elemento de Java debe cumplir la regla:

   a) Debe empezar por una letra que puede estar seguida de más letras

   b) Debe empezar por una letra que puede estar seguida de más letras o dígitos

   c) Debe empezar por una letra que puede estar seguida de dígitos

3. En Java los nombres de variables y métodos deben empezar por minúscula. Si el nombre es compuesto, cada palabra debe empezar por mayúscula.

   a) Falso

   b) Verdadero

4. En Java, los nombres de clases deben empezar siempre con mayúsculas. Si el nombre es compuesto, cada palabra debe empezar por mayúscula.

   a) Falso

   b) Verdadero

5. Un tipo de dato indica los valores que puede almacenar una variable y el rango de valores que admite.

   a) Falso

   b) Verdadero

6. ¿Cuál de las siguientes expresiones es correcta?

   a) double radio = 2;

   b) double radio = 2,0;

   c) double radio = 2.0;

7. El operador + está sobrecargado porque permite sumar números y concatenar cadenas de caracteres.

   a) Falso

   b) Verdadero

8. Indique el valor de: x = -1 + 5 * 7 – 12 / 3

   a) 28

   b) 30

   c) 6

9. Indique el valor de: x = (-1 + 5) * 7 – (12 / 3)

   a) 26

   b) 30

   c) 24

10. Indique el valor de: x = 3 > 2 && 6 < 10 || true

    a) true

    b) false

Ejercicios

1. Indique el orden de precedencia de los operadores aritméticos.

2. Indique el orden de precedencia de los operadores lógicos.

3. Indique la salida por pantalla del siguiente programa.

```
public class Calculo1 {
   public static void main (String[] args) {
      int x = 2, y = 2, z;

      z = x*y + 10 + 4 / 2 - 2 * 4 + 2;

      System.out.print("El resultado es ");
      System.out.print(z);
   }
}
```

4. Indique la salida por pantalla del siguiente programa.

```
public class Calculo2 {
   public static void main (String[] args) {
      boolean w, x = true, y = true, z = false;

      w = x && y || x && z || y && z;

      System.out.print("El resultado es ");
      System.out.print(w);
   }
}
```

5. Complete la tabla de tipos primitivos de Java.

Tipo	Descripción	Valor mínimo y máximo
byte		
short		
int		
long		
float		
double		
char		
boolean		

3. Clases y objetos

Test de evaluación

1. Una clase describe a un tipo de objetos con características comunes.

   a) Falso

   b) Verdadero

2. Un objeto es una representación abstracta de una clase.

   a) Falso

   b) Verdadero

3. Los atributos de un objeto solo pueden almacenar tipos primitivos de Java.

   a) Falso

   b) Verdadero

4. Cuando se ejecuta el método constructor de una clase:

   a) Se crea un alias y se inicializan los atributos del objeto

   b) Se asigna un espacio de memoria al objeto instanciado y se inicializan los atributos del objeto

   c) Se asigna un espacio de memoria al objeto instanciado pero no se inicializan los atributos del objeto

5. El método constructor se ejecuta cada vez que se instancia un objeto de la clase.

   a) Falso

   b) Verdadero

6. El estado de un objeto puede cambiar durante la ejecución de un programa Java.

   a) Falso

   b) Verdadero

7. Un objeto se compone de:

   a) Atributos

   b) Atributos y métodos

   c) Atributos y métodos constructores

8. Un método es una función que:

   a) Determina el comportamiento de una clase

   b) Determina el comportamiento de una clase y de sus objetos

   c) Determina el comportamiento de un objeto

9. El método `main()` se invoca cuando se ejecuta un programa Java.

   a) Falso

   b) Verdadero

10. La sobrecarga de métodos es útil para:

    a) Que el mismo método opere con parámetros de distinto tipo o que un mismo método reciba una lista de parámetros diferente

    b) Que el mismo método opere con parámetros de distinto tipo

    c) Que distintos métodos operen con parámetros de distinto tipo

Ejercicios

1. Explique brevemente los conceptos clase y objeto.

2. ¿Qué es un método? ¿Qué ocurre cuando se invoca un método?

3. Explique brevemente el concepto sobrecarga de métodos.

4. Defina la clase `Empleado`. Esta clase debe almacenar la siguiente información: dni, nombre, apellidos, domicilio, fecha de contratación y sueldo bruto. Todos estos datos son de tipo `String`, excepto el sueldo bruto que es `double`. Todos los atributos son de acceso privado.

   Defina el método constructor de la clase, los métodos 'get', 'set' y el método `getAtributos()`. Este método debe devolver una cadena con todos los atributos concatenados, como se muestra en el siguiente ejemplo:

   DNI: 202020X Fernández López, Juan 10/10/2010 32.500.00

   Defina la clase `MiPrograma` con el método `main()`. Defina un objeto `empleado1` y muestre sus atributos por la consola.

5. Defina dos métodos constructores para la clase `Persona`. El primer método debe recibir los parámetros dni, nombre y apellidos. El segundo método debe recibir los parámetros dni, nombre, apellidos y domicilio.

```
public class Persona {
   private String dni;
   private String nombre;
   private String apellidos;
   private String domicilio;

}
```

Desarrolle los métodos ‘get’, ‘set’ y el método `getAtributos()`. Este método debe devolver una cadena con todos los atributos concatenados, como se muestra en el siguiente ejemplo:

DNI 40902819M Rodríguez López, Juan c/Gran Vía, 10 Madrid.

Realice el diagrama de la clase `Persona`. Indique sus atributos y métodos y el tipo de acceso de cada elemento de la clase.

4. Extensión de clases

Test de evaluación

1. La composición consiste en crear una clase nueva agrupando objetos de clases que ya existen. Una composición agrupa uno o más objetos para construir una clase, de manera que las instancias de esta nueva clase contienen uno o más objetos de otras clases.

   a) Falso

   b) Verdadero

2. En una relación de composición, un objeto de la clase contenedora puede acceder a los métodos públicos de las clases contenidas.

   a) Falso

   b) Verdadero

3. La herencia es la capacidad que tienen los lenguajes orientados a objetos para extender clases. La clase original se denomina clase base o superclase, la nueva clase se denomina clase derivada o subclase.

   a) Falso

   b) Verdadero

4. En una relación de tipo herencia:

   a) Una subclase es una composición de la superclase. Normalmente una subclase añade nuevos atributos y métodos que le dan un comportamiento diferente al de la superclase

   b) Una superclase es una especialización de la subclase. Una superclase declara los atributos y métodos que definen el comportamiento de las subclases

   c) Una subclase es una especialización de la superclase. Normalmente una subclase añade nuevos atributos y métodos que le dan un comportamiento diferente al de la superclase

5. En una relación de tipo herencia, las subclases heredan los elementos públicos de la superclase y los métodos de la superclase.

   a) False

   b) Verdadero

6. La sintaxis de la declaración de una relación de herencia donde la clase base es `Persona` y la clase derivada `Empleado` es:

   a) public class Empleado extends Persona {

      }

   b) public class Persona extended by Empleado {

      }

   c) public class Persona extends Empleado {

      }

7. En una relación de tipo herencia un objeto de la superclase no puede almacenar un objeto de cualquiera de sus subclases.

   a) Falso

   b) Verdadero

8. En una relación de tipo herencia:

   a) Si un objeto de la clase base se asigna a una referencia de la clase derivada, se hace una conversión ascendente de tipos, denominada “upcasting”. La conversión ascendente de tipos siempre se puede realizar

   b) Si un objeto de la clase derivada se asigna a una referencia de la clase base, se hace una conversión ascendente de tipos, denominada “upcasting”. La conversión ascendente de tipos no siempre se puede realizar

   c) Si un objeto de la clase derivada se asigna a una referencia de la clase base, se hace una conversión ascendente de tipos, denominada “upcasting”. La conversión ascendente de tipos siempre se puede realizar

9. En una relación de tipo herencia:

   a) La conversión descendente de tipos, denominada “downcasting”, debe hacerse de forma implícita. El “downcasting” siempre es legal y no produce errores durante la ejecución del programa Java

   b) La conversión descendente de tipos, denominada “downcasting”, debe hacerse de forma explícita, indicando el nombre de la clase a la que se desea convertir. El “downcasting” no siempre es legal y puede producir un error durante la ejecución del programa Java

   c) La conversión descendente de tipos, denominada “downcasting”, debe hacerse de forma explícita, indicando el nombre de la clase a la que se desea convertir. El “downcasting” siempre es legal y no produce errores durante la ejecución del programa Java

10. Cualquier clase Java puede ser utilizada como una clase base para extender sus atributos y comportamiento. La clase derivada que se obtenga, puede a su vez, ser extendida de nuevo.

    a) Falso

    b) Verdadero

Ejercicios

1. Declare una clase `Empleado` con atributos de tipo `String` para dni, nombre, apellidos, domicilio, código postal y la ciudad. Esta clase debe incluir el atributo puesto de trabajo de tipo objeto de la clase `PuestoTrabajo`. Declare la clase `PuestoTrabajo` con atributos de tipo `String` para: código y descripción y un atributo `double` para el sueldo bruto. Desarrolle el método constructor en la declaración de ambas clases.

2. Desarrolle un programa Java que instancie al menos dos objetos de la clase `Empleado` declarada en la pregunta 1.

3. Declare las clases derivadas `Administrativo` y `Consultor` de la clase `Empleado`. La clase `Administrativo` tiene un atributo de tipo `int` para la antigüedad. La clase `Consultor` tiene un atributo de tipo `String` para la categoría profesional. Desarrolle el método constructor en la declaración de ambas clases. Este método debe invocar al constructor de la clase base.

4. Desarrolle un programa Java que instancie un objeto de las clases `Administrativo` y `Consultor` declaradas en la pregunta 3.

5. Defina el método `getAtributos()` de las clases `Empleado`, `Administrativo` y `Consultor`. El método `getAtributos()` de la clase `Empleado` debe utilizar el método `getDescripcion()` para mostrar la descripción del puesto de trabajo. Los métodos `getAtributos()` de las subclases `Administrativo` y `Consultor` deben sobreescribir el método de la superclase `Empleado`.

5. Ampliación de clases

Test de evaluación

1. Los elementos de clase son compartidos por todas las instancias de la clase.

   a) Falso

   b) Verdadero

2. Los atributos de clase deben tener un valor inicial aunque no exista ninguna instancia de la clase.

   a) Falso

   b) Verdadero

3. La palabra `final` se utiliza para indicar que el valor de un atributo es constante.

   a) Falso

   b) Verdadero

4. Los elementos privados de una clase:

   a) Se pueden utilizar libremente

   b) Solo se pueden utilizar dentro de la clase que los define

   c) Solo se pueden utilizar dentro de la clase que los define, en aquellas clases que la extiendan y cualquier clase definida en el mismo paquete

5. Los elementos protegidos de una clase:

   a) Se pueden utilizar libremente

   b) Solo se pueden utilizar dentro de la clase que los define

   c) Solo se pueden utilizar dentro de la clase que los define, aquellas clases que la extiendan y cualquier clase definida en el mismo paquete

6. Los elementos públicos de una clase:

   a) Se pueden utilizar libremente

   b) Solo se pueden utilizar dentro de la clase que los define

   c) Solo se pueden utilizar dentro de la clase que los define, aquellas clases que la extiendan y cualquier clase definida en el mismo paquete

7. Para utilizar componentes que están en otro paquete diferente se debe añadir una declaración de importación con la sintaxis:

   a) include nombre-del-paquete

   b) import nombre-del-paquete

   c) package nombre-del-paquete

8. ¿Para qué se usan las clases asociadas a los tipos primitivos?

   a) Para facilitar la programación en Java. Estas clases proporcionan métodos útiles para convertir cadenas de texto a otros tipos, para imprimir los números con diversos formatos y para describir los tipos simples

   b) Para definir nuevos tipos simples

   c) Para convertir cadenas de texto a otros tipos

9. La clase `String` se usa para manejar cadenas de caracteres.

   a) Falso

   b) Verdadero

10. El operador + está sobrecargado y puede utilizarse para concatenar cadenas de caracteres.

    a) Falso

    b) Verdadero

Ejercicios

1. Indique los cuatro niveles de derechos de acceso a los elementos de una clase Java.

2. Declare la clase `Producto` con los atributos código, descripcion, marca, precio, unidades en existencia, unidades vendidas. Los atributos código, descripción y marca son de tipo `String`. El atributo precio es de tipo `double` y los atributos unidades en existencia y unidades vendidas son de tipo `int`.

3. Explique qué es un paquete de Java.

4. Indique los elementos de la clase `Cliente` y el tipo de derecho de acceso de cada uno de ellos.

```
public class Cliente {
  private String nif;
  private String nombre;
  private String apellidos;
  private String tipoTarifa;

  public Cliente(String nif, String nombre,
                 String apellidos,String tipoTarifa) {
     this.nif = nif;
     this.nombre = nombre;
     this.apellidos = apellidos;
     this.tipoTarifa = tipoTarifa;
  }

  public String getNIF() {
     return this.nif;
  }
```

```
    public String getNombre() {
       return this.nombre;
    }

    public String getApellidos() {
       return this.apellidos;
    }

    private String getTipoTarifa() {
       return this.tipoTarifa;
    }

    public double getDescuento() {
       if (this.getTipoTarifa() == "Plata")
          return 10.0;
       else
          if (this.getTipoTarifa() == "Oro")
             return 15.0;
          else
             return 0.0;
    }

}
```

5. Utilice el constructor de la clase `Integer` para instanciar el objeto `numero` de tipo `int` con el valor inicial “2020”.

   Utilice el constructor de la clase `String` para instanciar el objeto `texto` de tipo `String` con el valor inicial “Introducción a Java”. Utilice el método `length()` para obtener la longitud del objeto texto y mostrarla por la consola.

6. Estructuras de control

Test de evaluación

1. La estructura `if` es una estructura de selección única porque ejecuta un bloque de sentencias solo cuando se cumple la condición del `if`. Si la condición es verdadera se ejecuta el bloque de sentencias. Si la condición es falsa, el flujo del programa continúa en la sentencia inmediatamente posterior al `if`.

   a) Falso

   b) Verdadero

2. La estructura `if-else` es una estructura de selección doble porque selecciona entre dos bloques de sentencias mutuamente excluyentes. Si se cumple la condición, se ejecuta el bloque de sentencias asociado al `if`. Si la condición no se cumple, entonces se ejecuta el bloque de sentencias asociado al `else`.

   a) Falso

   b) Verdadero

3. La estructura `switch` es una estructura de selección múltiple que permite seleccionar un bloque de sentencias entre varios casos. Es equivalente a una estructura de selección de `if-else` anidados y siempre que se puede utilizar un `if-else` anidado se puede aplicar un `switch`.

   a) Falso

   b) Verdadero

4. La expresión de un `switch` puede devolver un número entero (`int`), un número real (`double`), un carácter (`char`) o una cadena de carateres (`String`).

   a) Falso

   b) Verdadero

5. ¿Es posible interrumpir la ejecución de una estructura de selección `switch`?

   a) Sí, se utiliza la sentencia `continue` que provoca la finalización del `switch`. El flujo del programa continúa en la sentencia inmediatamente posterior al `switch`

   b) Sí, se utiliza la sentencia `break` que provoca la finalización del `switch`. El flujo del programa continúa en la sentencia inmediatamente posterior al `switch`

   c) No, no es posible interrumpir la ejecución de un `switch`

6. El operador condicional `(?:)` es el único operador de Java que utiliza tres operandos. El primer operando es una condición lógica, el segundo es el valor que toma la expresión cuando la condición es `true` y el tercero es el valor que toma la expresión cuando la condición es `false`.

   a) Falso

   b) Verdadero

7. De forma general, las estructuras de repetición se componen de:

   a) Dos partes: la condición y el bloque de sentencias

   b) Cuatro partes: la inicialización, la condición, el bloque de sentencias y la actualización

   c) Cinco partes: la inicialización, la condición, el bloque de sentencias, la actualización y el criterio de finalización

8. Seleccione la opción que describe correctamente las diferencias entre las estructuras de repetición `while`, `do-while` y `for`.

    a) La estructura de repetición `while` repite el bloque de sentencias mientras la condición es verdadera. La estructura `do-while` ejecuta el bloque de sentencias cero o más veces, comprueba la condición y repite el bloque de sentencias mientras la condición es verdadera. La estructura `for` repite el bloque de sentencias mientras la condición es verdadera

    b) La estructura de repetición `while` repite el bloque de sentencias mientras la condición es verdadera. La estructura `do-while` ejecuta el bloque de sentencias al menos una vez. Después comprueba la condición y repite el bloque de sentencias mientras la condición es verdadera. La estructura `for` repite el bloque de sentencias mientras la condición es verdadera

    c) La estructura de repetición `while` repite el bloque de sentencias al menos una vez. La estructura `do-while` ejecuta el bloque de sentencias cero o más veces, comprueba la condición y repite el bloque de sentencias mientras la condición es verdadera. La estructura `for` repite el bloque de sentencias mientras la condición es verdadera

9. La sentencia `break` se utiliza para interrumpir la ejecución de una estructura de repetición o de un `switch`. Cuando se ejecuta el `break`, el flujo del programa continúa en la sentencia inmediatamente posterior a la estructura de repetición o del `switch`.

    a) Falso

    b) Verdadero

10. La sentencia `continue` únicamente puede aparecer en una estructura de repetición. Cuando se ejecuta un `continue`, se deja de ejecutar el resto del bucle para volver al inicio de éste.

    a) Falso

    b) Verdadero

Ejercicios

1. Explique qué hace el siguiente programa e indique la salida por la consola para valores de la nota 4, 5, 6, 7, 8, 9 y 10.

```
public class Resultados {
  public static void main(String[] args) {
    int nota = 7;

    if (nota >= 5) {
      System.out.print("El resultado es aprobado ");

      switch (nota) {
       case 5:
       case 6:break;
       case 7:
       case 8:System.out.println("con Notable");
              break;
       case 9:System.out.println("con Sobresaliente");
       case 10:System.out.println("con Matrícula");
     }
    }
    else
      System.out.println("El resultado es suspenso");
  }
}
```

2. Explique qué hace el siguiente programa e indique la salida por la consola para valores de la nota 4, 5, 6, 7, 8, 9 y 10.

```
public class Resultados {
  public static void main(String[] args) {
    int nota = 7;

    System.out.println((nota>=5) ? (nota<8) ?
           "Entrevistar" : "Contratar" : "Rechazar");
  }
}
```

3. Explique qué hace el siguiente programa e indique la salida por la consola. ¿Cuántas iteraciones del `for` se realizan?

```
public class Numeros {
  public static void main(String[] args) {
    for (int i=1; i<=1000; i++) {
      if (i % 2 == 0)
         continue;

      if (i % 3 == 0)
         continue;

      System.out.println("Números: " + i);

       if (i >= 15)
          break;
    }
  }
}
```

4. Desarrolle un programa Java para calcular el producto de dos números ‘n’ y ‘m’ con sumas utilizando una estructura de repetición `for`.

5. Utilice la estructura de repetición `while` para desarrollar un programa Java para calcular la potencia de un número utilizando productos. Para calcular la función potencia de un número entero positivo utilizando productos. La potencia se calcula como el producto de la base repetido tantas veces como el valor del exponente.

   `potencia = base x base x base x base x... x base`

7. Estructuras de almacenamiento

Test de evaluación

1. Un array permite almacenar muchos objetos de la misma clase e identificarlos con distinto nombre.

   a) Falso

   b) Verdadero

2. El tipo base de un array es el tipo que se declara para todos sus elementos. El tipo base puede ser un tipo primitivo de Java, un objeto o una clase definida.

   a) Falso

   b) Verdadero

3. El valor inicial de un array es `null` y antes de hacer referencia a los elementos del array es necesario instanciarlo indicando el número de elementos que va a almacenar.

   a) Falso

   b) Verdadero

4. Para hacer referencia a un elemento de un array es necesario indicar la posición que ocupa en la estructura de almacenamiento. El primer elemento de un array se almacena en la posición 1 y el último elemento en la posición N, donde N es el tamaño del array.

   a) Falso

   b) Verdadero

5. La propiedad de un array que permite saber el número de elementos que tiene es:

   a) size

   b) elements

   c) length

6. La declaración `Cliente[] clientes = new Cliente[5]` corresponde a:

   a) Un array de tipo base `Cliente`, de una dimensión y tamaño 5, con identificador `clientes`

   b) Un array de tipo base `Cliente`, de una dimensión, con identificador `clientes`, que aún no ha sido instanciado

   c) Un array de tipo base `Cliente`, de una dimensión y tamaño 5, que puede almacenar objetos en las posiciones: `clientes[1]`, `clientes[2]`, `clientes[3]`, `clientes[4]`, `clientes[5]`

7. La declaración `int[][][] numeros = new int[2][3][4]` corresponde a:

   a) Un array de tipo base `int` de tres dimensiones, de tamaño 4x3x2, con identificador `numeros`

   b) Un array de tipo base `int`, de tres dimensiones, con identificador `numeros`, que aún no ha sido instanciado

   c) Un array de tipo base `int` de tres dimensiones, de tamaño 2x3x4, con identificador `numeros`

8. Un `for` “para todo” es una estructura de repetición que permite recorrer todos los elementos de un array. La variable de control del `for` toma el valor de todos los elementos del array indicado.

   a) Falso

   b) Verdadero

9. La búsqueda binaria es un algoritmo de búsqueda que se aplica a un conjunto de datos ordenado. El conjunto de búsqueda se delimita por dos posiciones: el límite inferior y el límite superior. El algoritmo empieza la búsqueda por el elemento que está almacenado en la mitad del conjunto de búsqueda. Si el elemento almacenado en la mitad del conjunto es mayor que el valor que se busca, entonces continúa la búsqueda en la primera mitad. Si el elemento almacenado en la mitad del conjunto es menor que el valor que se busca, entonces continúa la búsqueda en la segunda mitad. Si el elemento almacenado en la mitad del conjunto es igual que el valor que se busca, finaliza el proceso. En cada comparación, el algoritmo reduce el conjunto de búsqueda a la mitad. Si durante las sucesivas reducciones del conjunto de búsqueda el límite inferior es mayor que el límite superior, entonces el valor que se busca no está en el array y finaliza el proceso.

    a) Falso

    b) Verdadero

10. La clase `Arrays` de Java ofrece métodos que permiten realizar operaciones de ordenación y búsqueda en objetos de tipo array.

    a) Falso

    b) Verdadero

Ejercicios

1. Utilice un `for` “para todo” para recorrer el siguiente array y mostrar sus elementos en la consola.

```
String[] meses = {"Enero","Febrero","Marzo","Abril",
                  "Mayo","Junio","Julio","Agosto",
                  "Septiembre","Octubre","Noviembre",
                  "Diciembre"};.
```

2. Declare el array `numeros` de tipo `int` de dos dimensiones de tamaño 3x3. Inicialice el contenido del array de manera que el elemento `numeros[i][j]` almacene el valor `i+j`. Muestre el contenido de los elementos del array separados con un tabulador.

```
0      1   2
1      2   3
2      3   4.
```

3. Explique brevemente el algoritmo de búsqueda binaria. ¿Cuál es la condición que se cumple cuando el valor que se busca no existe en el conjunto de búsqueda?

   Para el conjunto de búsqueda definido por el array `numeros`, realice la búsqueda de los números 8, 15 y 19. Muestre los valores de los límites del conjunto de búsqueda de cada iteración.

```
int[] numeros={1,2,4,5,6,7,8,9,10,11,12,15,16,17,18};
```

4. Declare un array de objetos de la clase `Cliente` de tamaño 10 con identificador `misClientes`. Almacene en la posición cero del array un cliente con NIF 43658713X, nombre Juan y apellidos Fernández López. La clase Cliente se ha declarado:

```
public class Cliente {
   private String nif;
   private String nombre;
   private String apellidos;

   // se omiten los métodos 'get' y 'set' de la clase

   public Cliente(String nif, String nombre,
                  String apellidos) {
      this.nif = nif;
      this.nombre = nombre;
      this.apellidos = apellidos;
   }
}
```

5. Defina la clase `ConcesionarioVehiculos` con los atributos cif, nombre, domicilio y página web. Esta clase debe utilizar dos arrays redimensionables para registrar los vehículos en venta y sus clientes. La clase `Cliente` debe almacenar los atributos nif, nombre y apellidos. La clase `Vehiculo` debe almacenar los atributos matricula, marca, modelo, color y precio. Defina los métodos:

   `public void registrarCliente(Cliente c)`. Registra un cliente en la lista de clientes del concesionario.

   `public void imprimirClientes()`. Muestra la relación de clientes por la consola.

   `public void registrarVehiculo(Vehiculo v)`. Registra un vehículo en la lista de vehículos del concesionario.

`public void imprimirVehiculos()`. Muestra la relación de vehículos del concesionario.

`public void imprimirVehiculos(String marca)`. Muestra la relación de vehículos de la marca indicada.

Desarrolle un programa Java que defina una instancia de la clase `ConcesionarioVehiculos` con CIF "A-28-187189", nombre "cheap car", y página web "www.cheapcar.com". Defina varias instancias de la clase `Cliente` y `Vehículo`, registre los clientes y los vehículos en la empresa “cheap car”. Muestre la relación de clientes y el catálogo de vehículos de marca “VW”.

8. Entrada y salida

Test de evaluación

1. El esquema de entradas y salidas de Java, basadas en flujos, permite que las entradas sean independientes de la fuente de datos y que las salidas sean independientes del destino de los datos.

   a) Falso

   b) Verdadero

2. Un flujo en Java representa un canal de información del que se puede leer o escribir datos de forma secuencial.

   a) Falso

   b) Verdadero

3. Java ofrece flujos de entrada y salida para dos tipos de datos:

   a) Clases y bytes

   b) Caracteres y bytes

   c) Clases y objetos

4. Los flujos de salida de Java permiten escribir datos en ficheros de texto y en ficheros con formato binario.

   a) Falso

   b) Verdadero

5. Indique el nombre del flujo que se utiliza para leer datos del teclado.

   a) Keyboard.in

   b) System.in

   c) Scanner.in

6. La clase `Scanner` facilita la lectura del teclado. Indique cuál de las siguientes declaraciones de la instancia `entradaTeclado` de la clase `Scanner` es correcta.

   a) Scanner entradaTeclado = new Scanner(Keyboard.in)

   b) Scanner entradaTeclado = new Scanner()

   c) Scanner entradaTeclado = new Scanner(System.in)

7. Para leer y escribir un objeto en un fichero binario es necesario que la clase a la que pertenece el objeto sea “serializable”. Esto significa que los objetos de la clase se codifican dentro de los flujos de entrada y salida de Java.

   a) Falso

   b) Verdadero

8. El método `readObject()` de la clase `ObjectInputStream` lee un objeto almacenado en un fichero binario. Después de leer el objeto:

   a) Es opcional convertir el objeto a la clase a la que pertenece

   b) No es necesario convertir el objeto a la clase a la que pertenece

   c) Es necesario convertir el objeto a la clase a la que pertenece

9. La sentencias `try` y `catch` permiten atrapar los errores que se producen en tiempo de ejecución de un programa Java, denominados excepciones.

   a) Falso

   b) Verdadero

10. En una sentencia `try-catch-finally`:

    a) Los bloques `catch` se pueden repetir tantas veces como excepciones de distinto tipo se desee atrapar. El bloque `finally` debe aparecer al menos una vez y se ejecuta siempre

    b) Los bloques `catch` se pueden repetir tantas veces como excepciones de distinto tipo se desee atrapar. El bloque `finally` no es opcional y se ejecuta siempre

    c) Los bloques `catch` se pueden repetir tantas veces como excepciones de distinto tipo se desee atrapar. El bloque `finally` es opcional y solo puede aparecer una vez. Este bloque se ejecuta siempre

Ejercicios

1. Describa brevemente los métodos `next()`, `nextLine()` y `nextInt()` de la clase `Scanner`.

2. Desarrolle un programa Java que defina una instancia de la clase `Scanner` para leer del teclado. Utilice los métodos `next()`, `nextLine()` y `nextInt()` para leer el nombre completo de una persona, su nacionalidad y el año de su nacimiento, respectivamente.

3. Desarrolle un programa Java que defina una instancia de la clase `PrintWriter` y utilice el método `println()` para escribir en un fichero de texto el contenido del siguiente array:

```
String[] meses = {"Enero", "Febrero","Marzo","Abril",
                  "Mayo","Junio","Julio","Agosto",
                  "Septiembre","Octubre","Noviembre",
                  "Diciembre"};
```

4. Desarrolle un programa Java que defina una instancia de la clase `File` para abrir el fichero Meses.txt. Utilice una instancia de la clase Scanner para leer los datos almacenados y muestre su contenido por la consola.

5. ¿Qué es una excepción? Explique brevemente el uso de la sentencia `try-catch-finally` para atrapar las excepciones de Java relacionadas con la lectura y escritura de ficheros.