**Programación Ejercicios tema 2 Página 1 de 2**

# Programación. Ejercicios Tema 2

1. ¿Por qué utilizamos identificadores significativos en nuestro código?

Para que sea más intuitivo.

1. ¿Qué se almacena en una variable de un tipo primitivo?

El dato que se le asigna.

1. ¿Es un algoritmo lo mismo que un programa?

No, un algoritmo es: Una secuencia de instrucciones que representan un modelo de solución para una serie de problemas.

Y un programa es: Una secuencia de instrucciones y comandos escritas en código para realizar una tarea concreta en un ordenador.

1. ¿Es un carácter un tipo primitivo?

Que es un carácter de tipo primitivo:

Es un tipo de dato utilizado para representar un único dato.

**Res2:**

char en java, lo es.

1. Escribe una declaración de constante que asigne el nombre ZED al valor ‘Z’

final String Z = “ZED”;

1. Declara una variable de tipo carácter denominada letra y una variable de tipo cadena denominada calle

char letra;

String calle;

1. Asigna el valor “Elm” a la variable de tipo cadena denominada calle.

calle = “Elm”;

1. Añade el valor “ Street” al valor de calle.

calle = calle + “Street”;

1. El siguiente código es incorrecto; indica por qué y corrígelo

Falta una barra oblicua en cada comentario:

int miNumero;/**/** Número de ocasiones en que

/**/** he realizado este ejercicio

1. Indica si son o no válidos los siguientes identificadores.
   1. Elemento#1 → Invalido
   2. Dato → Valido
   3. Y → Valido, pero no lo recomendaría
   4. 3Conjunto → Invalido
   5. DIA\_DE\_PAGA → Valido pero solo para constantes
   6. bin-2 → Invalido
   7. num5 → Valido
   8. M 2 → Invalido
2. Haz coincidir cada uno de los términos siguientes con las definiciones correctas. Sólo una definición es correcta

|  |
| --- |
| 1. Programa → n) 2. Algoritmo → m) 3. Compilador → h) 4. Identificador → f) 5. V. Variable → a) 6. Constante → g) 7. Memoria → l) |

* 1. Nombre simbólico compuesto de letras, dígitos, subrayados y símbolos de dólar, pero que no comienza con un dígito
  2. Un dispositivo de entrada/salida
  3. El tipo dedicado a planificar el programa
  4. Reglas gramaticales
  5. Cuando se pone en ejecución un programa compilado
  6. El significado de algo
  7. Lugar de la memoria en la que se almacena un objeto cuyo valor no puede ser modificado
  8. Programa que traduce un programa en lenguaje de alto nivel a código máquina o bytecode
  9. Una secuencia de instrucciones encerradas entre llaves

**Programación Ejercicios tema 2 Página 2 de 2**

* 1. Lugar de la memoria en la que se almacena un objeto cuyo valor puede ser modificado
  2. Un programa que traduce un programa en ensamblador a código máquina
  3. La parte del computador que puede almacenar código y datos
  4. Instrucciones para resolver un problema en un tiempo finito utilizando una cantidad finita de datos
  5. Especificaciones de los tipos de datos y las instrucciones utilizadas por un ordenador para resolver un problema.

1. Si las variables de tipo cadena s1 y s2 contienen, respectivamente “pájaro” y “azul”, cuál es el resultado de las siguientes expresiones.
   1. “s1 = “ + s1 + “s2 = “ + s2 → s1 = pajaros2 = azul
   2. “Resultado:” + s1 + s2 → Resultado:pájaroazul
   3. “Resultado: ” + s1 + s2 → Resultado: pájaroazul
   4. “Resultado: ” + s1 + ‘ ‘ +s2 → Resultado: pájaro azul
2. ¿Cuántos caracteres podemos almacenar en una variable de tipo carácter?

Solo uno.

1. ¿Cuántos caracteres hay en la cadena vacía?

Ninguno.

1. ¿Puede una variable de tipo cadena ser asignada a una variable de tipo carácter?

No, porque son tipos diferentes.

1. ¿Cuál es la diferencia entre el literal “ordenador” y el identificador ordenador?

Que el literal en este caso es una cadena y el identificador es la palabra que ser refiere a una variable.

1. Dados los siguientes valores para las variables booleanas xV, yV y zV, evaluar las expresiones booleanas y responder si el resultado es verdadero o falso, teniendo en cuenta que:

**xV = true yV = flase zV = true**

* 1. (xV && yV) || (xV && zV) = true
  2. (xV || ! yV) && (! xV || zV) = true
  3. xV || yV && zV = true
  4. ! (xV || yV) && zV = false

1. Dados los siguientes valores para las variables $i, $j, $k y $l, añadir los paréntesis que sean necesarios a las expresiones de abajo para que se evalúen todas dando valor verdadero.

Tomamos como referencia que

**$i = 10 $j = 19 $k = true $l = false**

* 1. $i == $j || $k = true
  2. $i >= $j || $i <= $j AND $k = true
  3. ! $k || $k = true
  4. ! ($l && $l) = ~~false,~~ true

1. Las siguientes expresiones tienen sentido pero no son válidas de acuerdo con las reglas de Java. Reescribirlas como expresiones booleanas válidas.
   1. $x < $y <= $z ➩ if ($x < $y && $y <= $z)
   2. $x, $y y $z son mayores que 0 ➩ if ($x < $y && $y <= $z && $y <= $z)
   3. $x es diferente a $y y a $z ➩ if ($x < $y && $y <= $z)