Respuestas semana 1

Góngora Martínez Adán Antonio

1.- ==, ! Son los únicos operadores que se pueden usar con variables booleanas, debido a la naturaleza de las mismas, pues solo pueden tener valores de verdadero o falso.

2.- Int, long y double son los tipos de datos que permitirían a el programa compilar, pues se necesita de un espacio en memoria lo suficientemente grande como para almacenar un valor entero, que es el que se devuelve al sumar un short y un byte. Float también puede almacenarlo aunque no este entre las respuestas.

3.- Hacer un cast en la variable “ear” a un int, hacer un cast a la operación “2*ear”, cambiar el tipo de dato de “hearing” de int a long. Esto por que la variable hearing esta declarada como un int, por lo que en principio se debe hacer un casteo ya que el espacio que ocupa un long es mucho mayor que un entero.

4.-Ninguna de las anteriores. El programa no va a compilar porque el primer parámetro introducido va a ser casteado a long y un long no se puede convertir a float.

5.- 4,5,1. Primero, el programa muestra pregunta si a es mayor que 2, lo cual es falso ya que a vale 2, y por lo tanto este retorna el valor de b (4) y que después de mostrarlo, le suma 1. En segundo lugar, pregunta si a no es igual que c, como el resultado es falso, regresa el valor de b, que previamente se había incrementado a 5, y después de mostrarlo lo vuelve a incrementar. Finalmente, el programa hace una doble comparación, preguntando primero si a es mayor que b, y como esto es falso, pregunta si b es menor que c, falso de nuevo, por lo cual se regresa el valor 1.

6.-Primero, el programa realiza la suma del valor de ticketsSold + 1 + el valor de ticketsTaken(al cual después de realizar la operación se le aumentará 1), como estos valores son respectivamente 3+1+1, el resultado es 5 y este se almacena en ticketsSold. Despues el programa multiplica el valor de ticketsTaken por 2, siendo esta operación un 2*2 y lo almacena en la misma variable. Y finalmente, el programa suma el valor de ticketsSold (5) con un 1 casteado a un long, lo cual genera que java convierta ese esultado en un int, para poder almacenarlo. Por lo tanto, ticketsSold es 6 y ticketsTaken es 4.

7.- Primero, el programa calcula cual es la humedad mediante la operación -4+4*3, que (mediante las leyes de los signos) da como resultado 8. Ahora el programa hace una comparación, como temperatura es igual a 4, entra a la segunda comparación, y como humedad es 8, este imprime “Just right”

8.- break RABBIT y continue BUNNY. La lógica para ambas respuestas es la misma, cuando la suma de col y row sea un número par, regresa al for BUNNY para no aumentar el varlo de count.

De esta manera podemos ver que las variables toman los estos respectivos valores con cada ciclo:

row =1, col=0, count=1.

row =1, col=1, count=1.

row =2, col=0, count=1.

row =3, col=0, count=2.

row =3, col=1, count=2.

9.- No compila por que la variable keepGoing debe de ir entre paréntesis por sintaxis.

10.- Compila, pero el código nunca termina de ejecutarse por que el ciclo do while de que empieza en la línea 13 se ejecuta infinitamente.

11.- El código no compila por que no se puede transformar un int en un String.

12.- abbaccca. Por que primero el programa le concatena al StringBuilder un “aaa”, después le inserta un “bb” después del primer carácter, dando como resultado un “abbaa”, y finalmente, le inserta después del cuarto carácter un “ccc”.

13.- No se puede comparar de esa manera (“s1==s3”) una variable de referencia String con una StringBuilder

14.- Primero crea una variable de referencia que apunta al objeto “roar” dentro de la String Pool, despues crea una variable de referencia que apunta al objeto StringBuilder “roar” fuera de la String Pool. Después se crea un objeto de la clase Lion, el cual concatena al String un “!!!”, lo que significa que crea un objeto de tipo String nuevo que contiene “roar!!!”, y después se concatena al StringBuilder un “!!!”, pero como los objetos de la clase StringBuilder son mutables, en lugar de crear un nuevo objeto, se cambia el valor de existente a “roar!!!”.

Por eso, cuando se imprimen las variables, debería aparecer “roar roar!!!”

15.- La palabra clave var no se introdujo hasta Java 10.

Pero en el caso de que se reemplace var con el StringBuilder, la línea faltante debería ser “puzzle.reverse();” que es un metodo de la clase que permite retornar los caracteres en el orden inverso.