Indicaciones para resolver el Boletín 3 de problemas.

1. Pedir un número y leerlo desde el teclado. Hacer un bucle while con la condición de que el número sea distinto de cero, y dentro del bucle, hacer lo que se pide para cada número: mostrar si es par, mostrar si es positivo y mostrar su cuadrado. Antes de finalizar el bucle pedir el siguiente número y leerlo.

   Otra opción es usar un bucle do – while con la misma condición. En ese caso, sólo habría que pedir el número y leerlo al principio del bucle, pero el resto de las operaciones habría que ponerlas en una estructura condicional, para que se ejecuten si el número introducido es distinto de cero.

2. Igual que en ejercicio anterior se puede hacer con un while o con un do – while. La condición en este caso es que la edad sea mayor o igual que cero. En este ejercicio hay que llevar la cuenta de varios datos acumulados durante el proceso. Para ello se usan acumuladores, como por ejemplo, sumaEdades, cuentaAlumnos o cuentaAlumnosMayores. Estas variables se inicializarán a cero antes de comenzar el bucle, y dentro del bucle se le sumará a sumaEdades, la edad del alumno en curso y se incrementará cuentaAlumnos. También se incrementará cuentaAlumnosMayores si procede. Después de finalizar el bucle, cuando ya se disponga de todos los datos, se calculará la media de las edades como sumaEdades / cuentaAlumnos, y finalmente se mostrarán todos los datos que nos piden.

3. Calcular els número secreto usando la función Math.random() de la API estándar Java. Dicha función devuelve un número aleatorio de tipo double entre 0 y 9,99999… Para obtener un número aleatorio entero entre 1 y 100 podemos hacer 1 + (int)(Math.random() * 100). Almacenamos el número en una variable entera. Preguntar al usuario un número y almacenarlo en una variable (por ejemplo, intento) y realizar un bucle while que dependerá, como en los casos anteriores, del número introducido, pero en esta ocasión, la condición para continuar iterando será que intento sea distinto del número secreto (no se ha acertad) e intento distinto de -1. Dentro del bucle compararemos el intento con el número secreto y le diremos al usuario si el número es mayor o menor, para darle una pista. Una vez finalizado el bucle, si intento = -1, es porque el usuario se ha rendido y le mostraremos un mensaje al respecto. Si no, es que se ha encontrado el número secreto. En este caso le daremos la enhorabuena.

4. Pedir un número entero y leerlo en una variable, por ejemplo, n. Hacer un bucle for que recorra los valores desde 1 hasta n con un contador, por ejemplo i:

```
for (int i = 1;i <=n; i++) {
     mostrar(i);
}
```

   Al principio del bucle se declara y se inicializa la variable i con el valor 1. Antes de comenzar cada ciclo se evalúa la condición del bucle y si es true se ejecuta el bloque de instrucciones que contiene. Al final de cada ciclo se ejecuta la actualización. En este caso la condición es que i sea menor o igual que n. De esta forma se irán recorriendo con el contador i todos los enteros desde 1 mientras i <= n, es decir, hasta n (incluído). En el momento en que i sea mayor que n, ya no se ejecutará el bloque y el programa continuará después de la estructura for.

5. Pedir un número y leerlo en la variable minimo. Pedir otro y leerlo en la variable maximo. Realizar un bucle while donde se pedirá un número. La condición será que el número esté fuera del rango indicado por minimo y maximo. Dentro del bucle se volverá a pedir el número. Una vez que el bucle termine ya tendremos un número dentro del rango. También se puede hacer con do – while, como en los dos primeros ejercicios.

6. Declarar un contador, por ejemplo, i e inicializarlo a 1. Realizar un bucle while que muestre 7 * i y posteriormente incremente i. La condición del while es que el correspondiente múltiplo de 7 sea menor que 100, es decir 7 * i < 100. También se puede hacer un bucle for con la condición indicada.

7. Realizar un bucle que recorra los números entre 1 y 19 de dos en dos. Mostrar el número que se está recorriendo dentro del bucle. Otra opción: Recorrer los números desde 0 hasta 9 y mostrar 2 * i +1.

8. Pedir un número y leerlo en la variable entera n, por ejemplo. Declarar otra variable entera, por ejemplo factorial e inicializarla a 1. Realizar un bucle que recorra los números desde 1 hasta n. Dentro del bucle, multiplicar la variable factorial por el valor actual del contador del bucle. Una vez terminado el bucle mostrar la variable factorial.

9. Declarar una variable entera para identificar a cada árbol e inicializarla a cero. Declarar otra variable para llevar la cuenta de cuál es el más alto de los introducidos hasta el momento, por ejemplo, masAlto y donde tendremos la máxima altura registrada hasta el momento, por ejemplo, alturaMaxima y la inicializamos a cero. Crear un bucle para ir pidiendo la altura de cada árbol. La condición es que la altura introducida sea distinto de -1. Dentro del bucle compararemos la altura introducida con la alturaMaxima. Si la introducida es mayor, la almacenaremos en alturaMaxima y almacenaremos el contador de árbol actual en masAlto, para dejar registrado que el árbol más alto hasta el momento es el actual. Cuando finalice el bucle tendremos en masAlto el identificador del árbol más alto, y en alturaMaxima, su altura.

10. Pedir un número entre 1 y 10 y guardarlo en la variable n, por ejemplo. Mediante una estructura similar a la del ejercicio 5, se volverá a pedir el número tantas veces como sea necesario hasta que esté dentro del intervalo [1, 10]. Realizar un bucle que cuente desde 1 hasta 10 y mostrar en cada ciclo del bucle n multiplicado por el contador del bucle.

11. Realizar dos bucles anidados que cuenten de 1 a 10. En el bloque de instrucciones del bucle anidado mostrar el producto de los contadores de ambos. Para separar las tablas, dentro del bucle de fuera y antes del bucle anidado, mostrar “Tabla del “ y el contador del bucle externo.

12. Crear una variable boolean para registrar si se ha encontrado algún suspenso. Podemos llamarla por ejemplo haySuspensos y la inicializaremos a false, ya que antes de pedir los datos aún no tenemos ningún suspenso. Realizar un bucle que pida 5 calificaciones. Dentro del bucle comprobamos si la calificación actual es suspenso. En caso afirmativo ponemos haySuspensos a true. Al final del bucle, la variable haySuspensos nos indicará si se ha encontrado algún suspenso.

13. Parecido al anterior pero en este caso se pedirán 6 notas. Se utilizarán contadores para contar suspensos, condicionados y aprobados. Dentro del bucle se realizarán las comprobaciones necesarias para cada nota y se incrementará el contador que corresponda. Después del bucle se mostrarán los tres contadores.

14. Pedir n por teclado. Realizar un bucle para las filas que cuente desde n hasta 1. Dentro de este bucle realizar otro que cuente desde 1 hasta el valor actual del bucle de fuera. Dentro del bucle anidado mostrar “*” (utilizar System.out.print() para que no haga saltos de línea). Después del bucle anidado pero dentro del bucle externo hacer un salto de línea con System.out.println(“”);

15. Pedir n, inicializar un contador, por ejemplo, cantidadPrimos a cero y realizar un bucle que cuente desde p=1 hasta n para buscar los primos en ese intervalo. Dentro de éste, crear una variable booleana, esPrimo e inicializarla a true. Escribir un bucle anidado que cuente desde 2 hasta el valor actual de p -1 y compruebe para cada valor recorrido si es un divisor de p. En caso afirmativo, poner esPrimo a false, ya que si encontramos un divisor de p, p no es primo. Después del bucle anidado, si esPrimo es cierto, incrementamos cantidadPrimos. Después del bucle exterior ya podemos mostrar cantidadPrimos.