Familia Profesional **Informática y Telecomunicaciones**		Nombre del Ciclo Formativo Título de **Técnico Superior en Desarrollo de Aplicaciones Web**			
Centro Educativo **IES Campanillas (sede CITIC)**		Módulo Profesional **Programación** Código: **0485** N.º de créditos ECTS: **14**		Profesor **Luis José Sánchez González**	
Curso lectivo **2015 / 2016**	Grupo **1º DAW**	Tipo de documento **Examen**	Trimestre **Primero**	Fecha **3 de diciembre de 2015**	Modelo **A**

INSTRUCCIONES

➔ El alumno debe entregar una carpeta con las soluciones al examen cuyo nombre debe estar formado por "Ex" seguido del número de lista, seguido de las iniciales. Por ejemplo, Facundo Romuedo Piladro que es el número 8 de la lista entregaría una carpeta con nombre **Ex08frp**.
➔ Los ficheros o carpetas correspondientes a las soluciones se deben nombrar igual que la carpeta junto con el número del ejercicio, por ejemplo **Ex08frp1.java, Ex08frp2.java, etc**.
➔ En los comentarios de cada programa **se debe indicar el nombre completo**, la fecha y - si procede - el turno.
➔ Únicamente se necesita entregar el código fuente en java, **no se deben entregar los archivos con la extensión .class**.

EJERCICIOS

1. Realiza un programa que calcule la nota que hace falta sacar en el segundo examen de la asignatura **Programación** para obtener la media deseada. Hay que tener en cuenta que la nota del primer examen cuenta el 40% y la del segundo examen un 60%.
   Ejemplo 1:

```
Introduce la nota del primer examen: 7
¿Qué nota quieres sacar en el trimestre? 8.5
Para tener un 8.5 en el trimestre necesitas sacar un 9.5 en el segundo examen.
```

   Ejemplo 2:

```
Introduce la nota del primer examen: 8
¿Qué nota quieres sacar en el trimestre? 7
Para tener un 7 en el trimestre necesitas sacar un 6.33 en el segundo examen.
```

2. Escribe un programa que rellene un array de 100 elementos con números enteros comprendidos entre 0 y 500 (ambos incluidos). A continuación el programa mostrará el array y preguntará si el usuario quiere destacar el máximo o el mínimo. Seguidamente se volverá a mostrar el array escribiendo el número destacado entre dobles asteriscos.
   Ejemplo:

```
459 204 20 250 178 90 353 32 229 357 224 454 260 310 140 249 332 426 423 413 96 447
465 298 459 411 118 480 302 417 42 82 126 82 474 362 76 190 104 21 257 88 21 251 6 383
47 78 392 394 244 494 87 253 376 379 98 364 237 13 299 228 409 402 225 426 267 330 243
209 426 435 309 356 173 130 416 15 477 34 28 377 193 481 368 466 262 422 275 384 399
397 87 218 84 312 480 207 68 108
¿Qué quiere destacar? (1 – mínimo, 2 – máximo): 1
459 204 20 250 178 90 353 32 229 357 224 454 260 310 140 249 332 426 423 413 96 447
465 298 459 411 118 480 302 417 42 82 126 82 474 362 76 190 104 21 257 88 21 251 **6**
383 47 78 392 394 244 494 87 253 376 379 98 364 237 13 299 228 409 402 225 426 267 330
243 209 426 435 309 356 173 130 416 15 477 34 28 377 193 481 368 466 262 422 275 384
399 397 87 218 84 312 480 207 68 108
```

(el examen continúa en la siguiente página)

3. Escribe un programa que pida 8 palabras y las almacene en un array. A continuación, las palabras correspondientes a colores se deben almacenar al comienzo y las que no son colores a continuación. Puedes utilizar tantos arrays auxiliares como quieras. Los colores que conoce el programa deben estar en otro array y son los siguientes: verde, rojo, azul, amarillo, naranja, rosa, negro, blanco y morado.
Ejemplo:

Array original:

0	1	2	3	4	5	6	7
casa	azul	verde	ordenador	morado	bombilla	bicicleta	rosa

Array resultado:

0	1	2	3	4	5	6	7
azul	verde	morado	rosa	casa	ordenador	bombilla	bicicleta

4. Un restaurante nos ha encargado una aplicación para colocar a los clientes en sus mesas. En una mesa se pueden sentar de 0 (mesa vacía) a 4 comensales (mesa llena). Cuando llega un cliente se le pregunta cuántos son. De momento el programa no está preparado para colocar a grupos mayores a 4, por tanto, si un cliente dice por ejemplo que son un grupo de 6, el programa dará el mensaje **"Lo siento, no admitimos grupos de 6, haga grupos de 4 personas como máximo e intente de nuevo"**. Para el grupo que llega, se busca siempre la primera mesa libre (con 0 personas). Si no quedan mesas libres, se busca donde haya un hueco para todo el grupo, por ejemplo si el grupo es de dos personas, se podrá colocar donde haya una o dos personas. Inicialmente, las mesas se cargan con valores aleatorios entre 0 y 4. Cada vez que se sientan nuevos clientes se debe mostrar el estado de las mesas. Los grupos no se pueden romper aunque haya huecos sueltos suficientes. El funcionamiento del programa se ilustra a continuación:
Ejemplo:

Mesa nº	1	2	3	4	5	6	7	8	9	10
Ocupación	3	2	0	2	4	1	0	2	1	1

¿Cuántos son? (Introduzca -1 para salir del programa): 2
Por favor, siéntense en la mesa número 3.

Mesa nº	1	2	3	4	5	6	7	8	9	10
Ocupación	3	2	2	2	4	1	0	2	1	1

¿Cuántos son? (Introduzca -1 para salir del programa): 4
Por favor, siéntense en la mesa número 7.

Mesa nº	1	2	3	4	5	6	7	8	9	10
Ocupación	3	2	2	2	4	1	4	2	1	1

¿Cuántos son? (Introduzca -1 para salir del programa): 3
Tendrán que compartir mesa. Por favor, siéntense en la mesa número 6.

Mesa nº	1	2	3	4	5	6	7	8	9	10
Ocupación	3	2	2	2	4	4	4	2	1	1

¿Cuántos son? (Introduzca -1 para salir del programa): 4
Lo siento, en estos momentos no queda sitio.

Mesa nº	1	2	3	4	5	6	7	8	9	10
Ocupación	3	2	2	2	4	4	4	2	1	1

¿Cuántos son? (Introduzca -1 para salir del programa): -1
Gracias. Hasta pronto.

Nota: No es necesario comprobar en ningún ejercicio los datos introducidos por el usuario. Suponemos que el usuario introduce los datos del tipo correcto y en el rango correcto.