CICLO FORMATIVO			ASIGNATURA	
Desarrollo de Aplicaciones Web			PROGRAMACIÓN	
FECHA/MODELO	**PROFESOR/A**	**CURSO**	***A rellenar por el profesor/a***	
FEB/2019	Manuel Vázquez Enríquez	1º	Calificación del examen Test 10% & Práctico 90%	

Nota:

- Leer con cuidado que se solicita en el enunciado y utilizar la "Salida de referencia" como método de compresión del algoritmo solicitado y de validación del creado.
- Crear un paquete llamado "examenFEB" y para cada uno de los ejercicios debéis crear una clase con el nombre que se indica al inicio del ejercicio en negrita.
- Antes de la finalización del examen debéis comprimir el paquete "examenFEB" y enviarlo por correo al docente a la dirección: **mvazquez@profesores.imf.com**. No se evaluará nada subido una vez haya terminado la hora de finalización del examen. **Verificar con el docente que está enviado correctamente.**

1. Ej1Copisteria. (1.0pto) Se nos pide realizar el programa de facturación de una copistería. Crea un algoritmo que:

- Solicite al usuario, por teclado, el número de fotocopias que desea fotocopiar.
- Validar que el usuario introduce un número positivo de fotocopias. En caso de introducir un número negativo saldrá el siguiente mensaje por pantalla: "Numero de fotocopias incorrecto." y finaliza el algoritmo.

En caso de que el número de fotocopias sea válido, la copistería ofrece un precio al superar un numero de fotocopias. Así pues:

- La fotocopia costará 0.05 cada fotocopia.
- Pero, si el pedido consta de más de 500 fotocopias cada fotocopia costará 0.03

Se pide mostrar por pantalla el resultado a pagar por el cliente según el número de fotocopias a imprimir.

Salida de referencia:

```
Ingrese número de fotocopias:125
La factura es de: 6.25
```

2. **Ej2ConvertirDias. (1.0pto)**. Realizar un algoritmo que solicite un día de la semana siendo los valores válidos ["Lunes"," Martes","Miercoles","Jueves","Viernes","Sabado","Domingo"] y convertirlo en un número entre 1 y 7. Ojo! Si el usuario introduce una cadena de texto no contemplada no valida el programa mostrará un mensaje de **"Entrada no válida".**

Salida de referencia:

```
Introduzca un día de la semana ["Lunes","Martes","Miercoles","Jueves","Viernes","Sabado","Domingo"] : Martes
2
```

3. Ej3GuionDocente (2.0ptos). Un alumno desea saber cuál será su calificación final en la materia de Programación. Dicha calificación se compone de los siguientes porcentajes:

- 60% del promedio de **tres boletines.**
- 40% de la calificación del **examen final**.

Crear un algoritmo que solicite estas cuatro notas: los tres boletines y la nota del examen final.

- **Ingreso de notas**: Recoger las 4 notas y validar que todas se encuentran entre valores de 0 y 10, sino mostrar por pantalla "Datos introducidos son erróneos" y finalizaría el algoritmo.
- **Calcular la nota de los boletines**: Las notas de los 3 boletines tienen el mismo peso. Calcular la **media**.
- **Calcular nota final**: Se aplicará para calcular la nota final las ponderaciones indicadas anteriormente tanto para el promedio de los boletines como del examen final.
- Mostrar la nota del alumno en número y en letra.

```
Introduzca nota boletin 1: 9
Introduzca nota boletin 2: 9
Introduzca nota boletin 3: 10
Introduzca nota examen: 7
Media de la nota de los boletines: 9
Nota final: 7.80000001
Nota final: Notable
```

NOTAS	
NUMERO	**TEXTO**
0.0 - 4.999999	Suspenso
5.0 - 5.999999	Suficiente
6.0 - 6.999999	Bien
7.0 - 8.999999	Notable
9.0 - 10.0	Sobresaliente

4. Ej4Login(2.0ptos). Crear un sistema de acceso en el cual el algoritmo solicite un usuario y contraseña hasta 5 veces, en caso de fallar mostrará el número de intentos disponibles y se bloqueará en caso de que el usuario haya fallado más de 5 veces seguidas la contraseña.

- Solicitar usuario y contraseña, hasta que este acierte o falle 5 veces (se bloquee el sistema). La combinación correcta de acceso es: [User: IMF, Contraseña:12345].
    - En caso de acierto termina el algoritmo y se muestra el siguiente mensaje: "Acceso permitido".
    - En caso de fallo, se mostrará el número de intentos disponibles y solicitará nuevamente el usuario y la contraseña.
    - En caso de fallar más de 5 veces, termina el algoritmo y se muestra el siguiente mensaje: "Acceso bloqueado".

Salida de referencia:

```
Introduzca usuario: iff
Introduzca contraseña: 12345
Contraseña erronea
Numero de intentos disponibles: 4
Introduzca usuario: IMF
Introduzca contraseña: 123333
Contraseña erronea
Numero de intentos disponibles: 3
Introduzca usuario: IMF
Introduzca contraseña: 12345
Acceso permitido
```

5. EjFunciones (1.5 punto). Crear un algoritmo en el cuál solicitamos 3 números enteros entre -500 y 500 (ambos incluidos). Y se solicita que se programe una serie de funciones:

- **Función mostrar datos: imprimir (int,int,int) : void**. Se muestra los datos introducidos con el siguiente **formato: "[** *num1* **,** *num2* **,** *num3* **]"** sustituyendo num1, num2, num3 por los valores introducidos por el usuario.
- **Función validar: validar(int,int,int) : boolean**. Dado todos los números introducidos indicar si podemos proseguir o no con el ejercicio. Retornará únicamente true si todos los números están dentro del rango permitido, por el contrario, retorna un falso y mostrará por pantalla "Formato incorrecto de alguno de los datos introducidos".
- **Función positivos o negativos:** Dados dos números nos indica por pantalla si ambos son números son positivos o si ambos números son negativos.
    - **positivosNegativos(int,int): void**
- **Función par o impar**: Nos retorna con un boolean si el valor introducido es un número par o impar.
    - **parImpar (int) : boolean**

6. Ej6Array2D.(1punto) Crear una matriz de dos dimensiones con un tamaño de 5x5 de números enteros. Debemos crear un algoritmo que insertar números en la diagonal (según la imagen). ***¡OJO! Se valorará el uso de bucles y no la inserción manual de datos en la matriz.***

4	0	0	0	0
0	3	0	0	0
0	0	2	0	0
0	0	0	1	0
0	0	0	0	0

7. EjFuncionesArrays.(1.5puntos) Crear algoritmo que solicite un tamaño N al usuario. Crear un vector de N posiciones de números enteros. Realizar las siguientes funciones:

- **Función Mostrar.** Función que muestra todas las posiciones de un vector (ojo sin utilizar la clase Arrays) sino que se debe recorrer cada una de las posiciones del vector y mostrarlo por pantalla.
    - **mostrar(int[]) : void**

- **Función Inicializar.** Función que dado un vector de entrada pone todos los números con valores -1. Es decir, si le damos un vector [0,1,0,0] el resultado de esta función debe ser [-1,-1,-1,-1]
    - **inicializar**(int[]) : void

- **Función Introducir Dato.** Función que, dado un vector, una posición y un valor a introducir. Realiza la inserción de dicho valor en la posición indicada.
  ¡OJO! Antes de inserta dato se debe validar lo siguiente:
  1. Que no se pida insertar un dato en una posición fuera del vector.
  2. Que no se pueda insertar un dato en una posición ya ocupada.

  Si tiene lugar alguna de las situaciones anteriores. No se tratará de insertar nada en el vector y se devolverá un falso. En caso de poder insertar un dato, este será insertado y la función devolverá true.

  - **insertar(int[], int, int) : boolean**