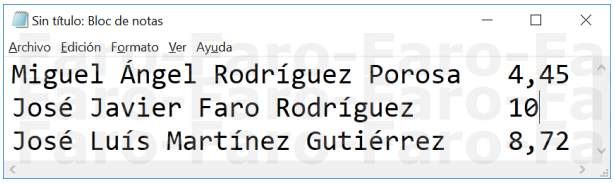
MÒDUL M03

Es demana el codi font en **java** dels programes que s'indiquen a continuació:

Lectura de fitxers de text en java

1º) – Tenim un fitxer de text pla anomenat **notes.txt** que presenta la sintaxis que s'indica a continuació (és un exemple, pot tenir un contingut diferent i més línies), on tenim el nom d'un alumne (de 1 a 4 paraules separades per espais) i la seva nota numèrica amb un màxim de dos decimals, on el separador decimal (que pot no existir) és la coma "**,**".



Per analitzar aquest fitxer, que llegirem línia a línia des del fitxer original, es demanen tres programes (tenen molts aspectes en comú, i es pot reutilitzar la major part del codi):

- **java Mesnota nomfitxer.ext nota (Exemple: java Mesnota notes.txt 6)**

  Escriu a la pantalla tots els alumnes que figuren en aquest fitxer que tenen una nota superior al valor indicat al segon paràmetre (**6** a l'exemple).

- **java Mitja nomfitxer (Exemple: java Mitja notes.txt)**

  Indica la màxima, i mínima nota presents al fitxer i també la mitja de totes les notes.

- **java NotaAlumne nomfitxer.ext nom_alumne**

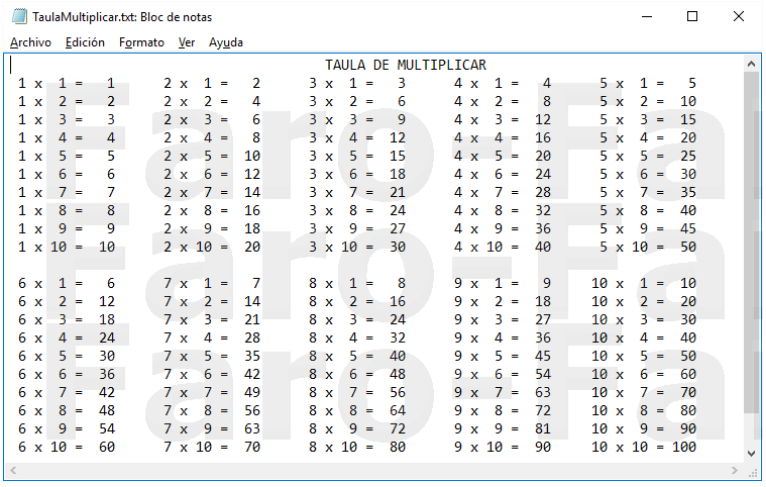
  Retorna la nota que té l'alumne especificat. El nom de l'alumne es podrà donar parcialment, en minúscules o majúscules, i els accents no son significatius (es busca sense accents). Si el nom de l'alumne conté espais, s'introduirà entre cometes dobles. Per exemple, si cridem al programa com:

  java NotaAlumne exemple.txt JOSE, per al fitxer de l'exemple que s'ha vist al començament de l'exercici, escriurà a la consola les dues coincidències:

  José Javier Faro Rodríguez 10
  José Luís Martínez Gutiérrez 8,72

Escriptura de fitxers de text en java

2º) – Crear un programa de nom **Taula.java**, al que cridarem sense paràmetres, i al mateix directori del programa, crearà un fitxer de nom **TaulaMultiplicar.txt** amb el contingut que s'indica:



3º) – Fer un programa que crearà el fitxer **numeros.txt** al mateix directori on s'executa. Aquest fitxer, contindrà **100** valors enters positius generats aleatòriament, el valor dels quals pot anar de **0** a **1000**. Els números s'escriuran al fitxer en **10** línies (**10** números per línia) i formatats en un camp de **5** (precedits d'espais). Recordeu que per escriure un retorn de carro hem de posar el caràcter de control **"\n"**, que segons el **S.O.**, a l'hora de ser traslladat al fitxer, es substituirà pels codis **13,10** (**CR-LF**) al **Windows** o únicament pel codi **10** (**LF**) al **Linux**. Podeu anomenar al programa **Num_100.java**.

4º) – El mateix programa de l'exercici anterior, però ara es dirà **FilesNum.java** i rebrà tres paràmetres numèrics indicant el valor màxim dels números a generar (primer paràmetre), el número de fileres de números que volem (segon paràmetre), i el número de valors que desitgem en cada filera (tercer paràmetre). Si cridem al programa com: **java FilesNum 1000 10 10**, el resultat serà el mateix que a l'exercici anterior. En aquest cas, la longitud del camp per escriure els números es calcularà amb la longitud de la cadena que conté el primer paràmetre **+ 1**. Així, si el primer valor és **1000000** per exemple, els valors s'escriuran en un camp de **8**.

Lectura-Escriptura de fitxers de text en java

5º) - Al fitxer **NomsMarvel.txt** tenim una llista de **superherois** i **dolents** de **Marvel**. El que hem de fer és cridar al nostre programa **java SeparaHerois NomsMarvel.txt** que a partir d'aquest fitxer, generarà al directori per defecte els fitxers **superherois.txt** i **dolents.txt**, on ha separat els noms del fitxer original en **superherois** i **dolents**. La forma de decidir si un nom pertany a un superheroi o a un dolent és la següent, si el com conté una **s** (majúscula o minúscula), considerarem que és un **superheroi**, si no la conté, serà un **dolent**. El programa comprovarà que cap dels fitxers de destí existeixi, i si fos el cas, informarà a l'usuari que esborri prèviament el o els fitxers i terminarà l'execució sense haver modificat els arxius existents.

6º) – Un anagrama és una paraula o frase que resulta de la transposició de lletres d'una altra paraula o frase, per exemple:

```
amor - roma - omar - mora - ramo - armo
monja - jamón - mojan
```

Donat un arxiu, que conté una llista de paraules en castellà (codificada en **UFT-8**, on hi ha una paraula per línia, i on el caràcter de final de línia és el codi **10 (LF)**, per tant es tracta d'un fitxer **Linux**), trobar totes les paraules que son anagrames entre si i escriure-les al mateix fitxer (el contingut previ es perdrà, de forma que el preservarem en un fitxer auxiliar de **backup** amb extensió **.bak**). La forma de procedir serà posar totes les paraules que son anagrames a la mateixa línia, separades per un espai. Es vol també que el fitxer resultant, estigui codificat en **cp1252** (es a dir, la codificació que fa servir **Windows**), i que el final de cada línia s'indiqui amb els codis **13 (CR) – 10 (LF)**, que és els que fa servir **Windows** per separar paràgrafs.

El fitxer original ocupa **775 KBytes**, i té **5903** grups d'anagrames d'un total de **80379** paraules. Per tant, el fitxer resultant, tindrà **5903** línies, on a cada línia han de figurar tots els anagrames de la mateixa paraula. Per facilitar la feina, podeu llegir el fitxer d'origen sencer a la memòria (en un vector de paraules per exemple).

El mètode de resolució que us proposo és llegir totes les paraules del fitxer en un vector **vo** (vector original). Crear una altre **vc** (vector còpia) amb totes les paraules d'aquest, però on a totes les paraules els hem tret els accents, les hem passat a minúscules i hem ordenat totes les seves lletres. A continuació, farem una recerca al vector **vc** (començant per la primera paraula) on per a cada paraula no descartada, buscarem al vector totes les paraules iguals entre si (a la de referència, inicialment a la primera del vector). Un cop trobades les paraules iguals a la primera, les escriurem al fitxer de destí, però des del vector original **vo** (les paraules es trobaran a la mateixa posició que al vector **vc**). Cada paraula escrita, es marcarà com descartada al vector **vc** (es pot assignar a la cadena buida per exemple). Un cop trobades totes les paraules iguals a la primera, procedirem amb la següent paraula no descartada (diferent de cadena buida). El programa l'haureu d'anomenar **Anagrama.java**, i li passareu com a únic paràmetre el nom del fitxer (Ex. de crida: **java Anagrama anagrames_wordlist.txt**).