

# Segona pràctica obligatòria

## Programació 1 – Grau en Enginyeria Informàtica

Curs 2021-2022

### Exercici 1 (5 punts) – Missatges d'SPAM

#### Introducció

Des de fa temps, els missatges de correu no desitjat (SPAM) són un problema per als que utilitzem habitualment el correu electrònic. El que tractarem de fer en aquest exercici és implementar un detector de missatges d'SPAM senzill, que reconegui i compti certes paraules en un text donat i que finalment, amb els resultats obtinguts, identifiqui el missatge com a SPAM o com a correu ordinari.

#### Enunciat

La pràctica consistirà en 3 fases:

1. **Obtenir les paraules a identificar:** Una de les tècniques que fan servir els programes anti-SPAM, és analitzar el text per buscar una sèrie de paraules que es fan servir sovint en aquest tipus de correu, i d'aquesta forma poder decidir si el correu s'ajusta a les característiques de l'SPAM. En aquesta fase, el que farem serà obtenir les paraules que voldrem identificar en el text introduïdes des de l'entrada estàndard (teclat o fitxer en cas de fer un redireccionament). El format a seguir per a l'entrada d'aquestes paraules serà el següent:
  - Les paraules vindran separades per un o més espais o canvis de línia.
  - La seqüència de paraules a identificar s'acabarà amb un punt.
2. Identificar les paraules en el text: Una vegada sabem les paraules que volem buscar en el text, passarem a l'obtenció del text que, a l'igual que les paraules a identificar, ens vindrà introduït per l'entrada estàndard. L'únic requisit a seguir per a l'entrada d'aquest text és que ha d'acabar amb 2 punts simples, separats per espais o no. No es permet emmagatzemar tot el text en memòria, per tant, el que es pretén és llegir el text paraula a paraula i tractar-les a mesura que les anem llegint.
3. Decidir si el missatge obtingut **es SPAM o no:** Finalment, una vegada obtinguts els resultats de buscar les paraules dins el text, el que hem de fer es decidir si el missatge llegit és o no és SPAM. Per prendre aquesta decisió, ho farem de la següent forma: Si apareixen 2 o més vegades el 50% o més de les paraules buscades en el text, llavors considerarem que el missatge rebut és SPAM, altrament considerarem que el missatge és de correu ordinari.

#### Consideracions

- Com a molt tindrem 20 paraules a identificar.
- Cada paraula tindrà com a molt 30 caràcters.
- El programa no ha de diferenciar entre majúscules i minúscules, és a dir, per al nostre programa és el mateix “HoLa” que “hola”.
- Junt amb l'enunciat de la pràctica trobareu varis fitxers de prova (ex1-jp?.txt) que us poden donar una idea de com us poden venir les dades redireccionant l'entrada des d'un fitxer.
- Els resultats que haurem d'obtenir del nostre programa són els següents: (i) El número de vegades que apareix cada paraula (ii) El rati de paraules que apareixen 2 o més cops en el text (iii) El resultat de si el missatge es SPAM o no. Podeu veure varis exemples de com ha de ser la sortida del vostre programa dins el fitxer “exemple-sortida.txt” que trobareu junt

amb l'enunciat.

- Als jocs de proves (ex1-jp?.txt), en alguns casos, al tractar-se de casos reals, s'ha procurat separar les paraules de símbols com parèntesis, cometes, etc... amb espais per a facilitar una mica la lectura de les paraules. Per tant, només cal tenir en compte que una paraula esta formada únicament per caràcters alfabètics i s'acaba quant es troba un caràcter no alfabètic.

## Exercici 2 (5 punts) – Comprovació de Sudokus

### Introducció

Un Sudoku és un trencaclosques de col·locació que requereix paciència i una certa habilitat lògica. El joc està format per una graella de 9x9 cel·les subdividida en 9 subgraelles de 3x3 anomenades regions. Donats uns quants números inicials, l'objectiu és col·locar un número de l'1 al 9 en cada cel·la de tal manera que mai coincideixin dos números iguals en cada línia horitzontal, vertical o en cada regió. Podeu trobar més informació a <http://ca.wikipedia.org/wiki/Sudoku>. A continuació podeu veure un Sudoku resolt:

---	---	---	---	---	---	---	---	---
1	2	3	7	8	9	4	5	6
---	---	---	---	---	---	---	---	---
4	5	6	1	2	3	7	8	9
---	---	---	---	---	---	---	---	---
7	8	9	4	5	6	1	2	3
---	---	---	---	---	---	---	---	---
2	3	1	8	9	7	5	6	4
---	---	---	---	---	---	---	---	---
5	6	4	2	3	1	8	9	7
---	---	---	---	---	---	---	---	---
8	9	7	5	6	4	2	3	1
---	---	---	---	---	---	---	---	---
3	1	2	9	7	8	6	4	5
---	---	---	---	---	---	---	---	---
6	4	5	3	1	2	9	7	8
---	---	---	---	---	---	---	---	---
9	7	8	6	4	5	3	1	2
---	---	---	---	---	---	---	---	---

### Enunciat

L'exercici consistirà en 3 fases:

1. Obtenir el Sudoku: El nostre programa rebrà des de l'entrada estàndard (teclat o fitxer en cas de fer un redireccionament) la seqüència de números que formaran el nostre Sudoku. Els números seran enters de l'1 al 9, separats per un o més espais, tabuladors o canvis de línia. Rebrem un mínim de 81 números per poder completar el nostre Sudoku. El primer número que rebrem serà el número de més a l'esquerra de la primera fila, la de dalt. El següent número serà el que va immediatament a la dreta de l'anterior i així successivament fins al novè número que formarà la primera fila. Els següents nou números formaran la segona fila i així successivament fins a formar el Sudoku complert quant haguem llegit els 81 números que el formen. Una vegada llegit tot el Sudoku, el mostrarem per pantalla tal com surt en l'exemple anterior.
2. Comprovar que el Sudoku és correcte: El segon pas consistirà en comprovar que el Sudoku és correcte, és a dir, que cada una de les seves files conté tots els números de l'1 al 9 sense

repeticions, el mateix per cada una de les seves columnes i per cada una de les seves regions de 3x3. Quant trobeu un error, heu de dir en quina fila (de la 1 a la 9), columna (de la 1 a la 9) o regió (de la superior esquerra 1,1 a la inferior dreta 3,3) l'heu trobat. Haureu d'aturar-vos al trobar el primer error, amb un sol error el Sudoku ja és incorrecte.

3. Mostrar el resultat: Si en el pas anterior heu trobat algun error, haureu de dir que el Sudoku és incorrecte. En cas contrari, haureu de dir que el Sudoku és correcte.

Exemple de Sudoku incorrecte:

	---		---		---		---		---		---		---		---		---		---
	1		2		3		7		8		9		4		5		6		
	---		---		---		---		---		---		---		---		---		
	4		5		6		1		2		3		7		8		9		
	---		---		---		---		---		---		---		---		---		
	7		8		9		4		5		6		1		2		3		
	---		---		---		---		---		---		---		---		---		
	2		3		1		8		9		7		5		6		4		
	---		---		---		---		---		---		---		---		---		
	5		6		4		2		3		1		9		8		7		
	---		---		---		---		---		---		---		---		---		
	8		9		7		5		6		4		2		3		1		
	---		---		---		---		---		---		---		---		---		
	3		1		2		9		7		8		6		4		5		
	---		---		---		---		---		---		---		---		---		
	6		4		5		3		1		2		9		7		8		
	---		---		---		---		---		---		---		---		---		
	9		7		8		6		4		5		3		1		2		
	---		---		---		---		---		---		---		---		---		

El Sudoku te un error a la columna 7.  
La solucio al Sudoku es INCORRECTA!

## Jocs de proves

Junt amb l'enunciat es deixaran una sèrie de fitxers per a que pugueu provar el funcionament dels exercicis i que es faran servir per a l'avaluació. Podeu utilitzar-los fent un redireccionament de l'entrada estàndard de la següent forma:

```
$ ./ex1 < ex1-jp0.txt
```

Hi ha varis jocs de proves (exX-jpX.txt) i els resultats que hauria de donar per cada un d'ells (exX-jpX.txt.out).

El programa ha d'estar implementat com si les dades fossin introduïdes per teclat. Per a provar els jocs de proves no cal fer cap modificació en el vostre programa, sempre i quant estigui ben implementat.

## Avaluació

- La pràctica s'ha d'implementar en el llenguatge ANSI C++ i s'ha d'executar correctament en una plataforma Linux. Per garantir que satisfà l'estàndard ANSI compileu afegint l'opció `-ansi`. És a dir, la comanda de compilació ha de ser:  

```
$ g++ programa.cpp -o executable -ansi
```
- S'han d'implementar els exercicis utilitzant les tècniques de disseny descendent vistes a classe.
- La pràctica es puntua sobre 10 punts i el seu pes a l'avaluació final és del 25%.

- La data límit per al lliurament de la pràctica és el **14 de Gener**.
- La pràctica es lliurarà via el Campus Virtual (CV), dins l'apartat Activitats.
- Es recomana posar comentaris a les parts importants del programa dins els fitxers .cpp dels exercicis que ajudin a interpretar l'algorisme implementat.
- La pràctica s'ha de resoldre individualment o en grups de màxim 2 persones.
- Com a comentaris de l'activitat heu d'indicar si la pràctica s'ha realitzat de forma individual o en grup. A més heu d'indicar els membres que formen part del grup. Finalment, comenteu breument l'estratègia emprada per resoldre cada problema.

## Aspectes a tenir en compte

Alguns aspectes que heu de tenir en compte a l'hora de realitzar les vostres pràctiques i que puntuaran negativament si no els teniu en compte:

- Nitidesa en el codi (tabulació correcta, no fer càlculs innecessaris, no repetir codi, utilització correcta dels recursos de la màquina...).
- Utilitzar estructures algorísmiques adients per als problemes a resoldre.
- Utilitzeu noms de variables entenedors.
- Llegiu atentament l'enunciat i no implementeu funcionalitats diferents a les que us demana.
- Queda prohibit utilitzar la instrucció de salt goto, o instruccions per alterar el funcionament normal d'un bucle com continue o break (l'únic cas en que es pot fer servir el break és en el cas que facis anar la instrucció switch).
- No es poden utilitzar llibreries no estàndard com la conio.h.
- En cas de realitzar pràctiques de forma “col·laborativa” entre diferents grups, esmentar-ho en el moment de l'entrega o en els comentaris, encara que finalment s'entreguin les pràctiques per separat o individualment.
- Per al primer exercici, no es permès emmagatzemar tot el text en memòria i després tractar-lo, ja que el text pot ser infinit. L'incompliment d'aquest requeriment significarà un 0 a l'exercici.
- Si es detecta que la pràctica és copiada, la nota és un 0. Tant pel que còpia com pel que deixa copiar.

## Mètode de correcció

Per a la validació del funcionament de la pràctica s'utilitzarà el Makefile i els corresponents jocs de proves. Tant els jocs de proves com el fitxer Makefile que s'utilitzarà per a la correcció els podreu trobar a la carpeta de la pràctica a l'apartat de Recursos del Campus Virtual. A l'hora de presentar la pràctica a través del CV, haureu d'enviar els següents fitxers:

- Makefile: El mateix fitxer “Makefile” que trobareu al CV.
  - ex?-jp?.txt: Els fitxers de proves que trobareu al CV.
  - ex?.cpp: Els fitxers on implementareu els vostres programes, un per cada exercici.
- Important que tinguin el mateix nom que us proporciono per a que tot funcioni correctament, ex1.cpp i ex2.cpp.**

Per poder enviar tots els fitxers sense problemes els podeu comprimir utilitzant la comanda 'tar' de la següent forma:

Suposant que teniu els exercicis resolts dins la carpeta 'prac2' podeu fer:

```
$ cd prac2
```

```
$ tar cvfz prac2.tgz *
```

El fitxer resultant és el fitxer prac2.tgz que conté tots els fitxers de la carpeta 'prac2'.

Comandes que se seguiran per la correcció:

1. `$ make clean`  
Per eliminar els possibles fitxers binaris que hi pugui haver.
2. `$ make all`  
Per compilar els exercicis seguint l'estàndard ANSI.
3. `$ make test`  
Per provar el correcte funcionament dels programes d'acord amb l'enunciat.

Podeu veure un exemple d'una sortida correcta després d'executar aquestes 3 comandes dins el fitxer “exemple-sortida.txt” que trobareu junt amb l'enunciat.