Laboratori Programació I, 2020-2021

Exercicis P3: Taules/Arrays i fitxers

Grau d'Enginyeria Informàtica

UNIVERSITAT DE BARCELONA

Aclaracions:

- Feu ús d'arrays quan realment sigui necessari. Si el problema es pot resoldre sense arrays, feu-lo sense arrays.
- En tots els problemes, cal que identifiqueu totes les seqüències existents i amb quin esquema algorísmic cal tractar-les (recorregut o cerca). La identificació es fa en un comentari abans de la composició iterativa corresponent.

```
/* Identificacio de la sequencia: …
 * Primer(): …
 * Seguent():…
 * FiSeq():…
 * Identificacio de l'esquema: Cerca o Recorregut
   (Quan es cerca, s'afegeix Condició de cerca: …)
*/
```

- Heu de resoldre TOTS els exercicis fent servir mètodes que es criden des del mètode `main()`.
- Recordeu fer servir JUnit per testejar els mètodes.

1. Feu un programa que donats un nombre natural `n` i dues seqüències de com a màxim de `n` nombres enters acabades en 0, indiqui si les dues contenen els mateixos números encara que potser en ordre diferent i amb repeticions. El programa mostra "Si" si son els mateix nombres, "No" si no ho són. (Conjunts.java)

   Entrada:
   10
   5 1 2 5 7 1 8 12 0
   7 12 8 5 1 8 2 0
   Sortida:
   Si

   Entrada:
   3
   5 1 2 0
   7 1 0

   Sortida:
   No

2. Feu un programa que indiqui si dues paraules entrades per l'usuari són o no iguals. No es pot utilitzar el mètode `equals()` de la classe String. (ParaulesIguals.java)

3. Creeu un programa que al iniciar-se crea un array amb els dies de la setmana ("Dilluns", "Dimarts", "Dimecres", etc..). A continuació es demanarà per consola un número del 1 al 7 i es mostrarà el dia que correspon a aquell número (Exemple: Num: 3 -> "Dimecres"). (DiesSetmana.java)

4. Feu un programa que llegeixi un enter `n` i tot seguit llegeixi una seqüència de n enters. El programa ha de retornar els nombres que estan repetits i el nombre de cops que es repeteixen. Si no hi ha nombres repetits, ha de sortir el missatge "No hi ha repetits" (Repetits.java)

   Entrada:
   5
   2 3 2 4 3

   Sortida:
   2 apareix 2 cops
   3 apareix 2 cops

   Entrada:
   4
   1 2 3 4

   Sortida:
   No hi ha nombres repetits

5. Feu un programa que donats un nombre natural `n` i una seqüència de `n` paraules en minúscules, indiqui quina és la paraula més freqüent. (ParaulaFrequent.java)
   Entrada:
   17
   pa amb oli i pa amb xocolata i pa amb vi i pa amb tomaquet i pa

   Sortida:
   pa

6. Feu el programa anterior però amb les dades (`n` i seqüència de `n` paraules) emmagatzemades en un fitxer "paraules.txt", que heu de crear abans en un editor de text pla. (ParaulaFrequentFitxer.java)

7. Feu un programa que multipliqui dues matrius. L'entrada comença amb les dimensions de les matrius: primer i segon enter són les dimensions de la primera matriu i tercer i quart enters les de la segona. A continuació es donen els valors. (MultMatrius.java)
   Entrada:
   3 2 2 4
   Introdueixi els valors de la matriu 3x2:
   1 0
   0.5 1
   1.5 -1

   Introdueixi els valors de la matriu 2x4:
   1 0 – 1.5 1
   0 2 0 -1

   Sortida:
   La matriu resultant és:
   1.0 0.0 -1.5 1.0
   0.5 2.0 -0.75 -0.5
   1.5 -2.0 -2.25 2.5

8. Feu un programa que indiqui si la matriu que se li passa és la matriu identitat. Es diu que una matriu és la identitat quan tots els seus elements són 0 a excepció de la diagonal principal, que es troba plena d'uns. Per a que una matriu sigui la identitat, aquesta ha de ser quadrada, és a dir, té el mateix nombre de files i de columnes (MatriuIdentitat.java).

   - L'entrada comença amb un nombre que indica el número de files de la matriu quadrada.
   - A continuació es demanen els elements de la matriu
   - Es demana aquesta informació a l'usuari fins que l'entrada sigui una matriu amb 0 files
   - Un exemple d'entrada i sortida seria el següent:

     ```
     3
     1 0 0
     0 1 0
     0 0 1
     2
     0 1
     1 0
     5
     1 0 0 0 0
     0 5 0 0 0
     0 0 1 0 0
     0 0 0 1 0
     0 0 0 0 1
     0
     ```

   - La sortida de l'exemple serà:

     ```
     SI
     NO
     NO
     ```

9. Feu un programa que, donats per teclat un nombre natural `n` i una llista de `n` estudiants, cadascun identificat pel seu nom (una paraula), cognoms (dues paraules) i DNI (una paraula), indiqui si hi ha 2 estudiants que tenen algun cognom en comú i imprimeixi les dades dels primers dos que trobi. (MateixCognom.java)

   Entrada:
   4
   Manuel Prats Lopez 87652211R
   Enric Vidal Garcia 55552222T
   Maria Lopez Roura 12345678R
   Jordi Carreras Prats 87654321S

   Sortida:
   Manuel Prats Lopez 87652211R
   Maria Lopez Roura 12345678R

10. Feu el programa anterior però aquesta vegada els dades estan emmagatzemats a un fitxer "estudiants.txt" que heu de crear abans a un editor de text pla. (MateixCognomFitxer.java)

11. Feu un programa que donats per teclat dos nombres naturals n i k i una seqüència de n estudiants, cadascun especificat amb el seu nom (una paraula) i nota (un nombre real no negatiu), imprimeixi els noms i cognoms dels k estudiants amb millor nota. (MillorsEstudiants.java)

    Entrada:
    4 2
    Joan 7.8
    Maria 9.5
    Carles 7
    Marta 8.6

    Sortida:
    Els 2 estudiants amb millor nota:
    Maria 9.5
    Marta 8.6

12. Feu el programa anterior però aquesta vegada els dades estan emmagatzemats a un fitxer "millorsEstudiants.txt" que heu de crear abans a un editor de text pla. (MillorsEstudiantsFitxer.java)

13. Feu un programa per a jugar a una versió simple del Yahtzee. El programa ha de simular la tirada de cinc daus i imprimeix "Yahtzee" si els cinc daus són iguals; en cas contrari, s'hauria d'imprimir "Torna a intentar-ho". Teniu en compte que la tirada de un dau dona un nombre entre 1 i 6. (Yahtzee.java)

    Implementa els següents mètodes:

    - `static void tiraDados(int [] daus)`

      Aquest mètode omple l'array `daus` amb els valors dels daus d'una tirada.

    - `static boolean esYahtzee(int [] daus)`

      Aquest mètode retorna t`rue` si s'ha aconseguit yatzee amb els valors del paràmetre `daus, false` en cas contrari.

14. Feu un programa per a jugar al **buscamines**. L'usuari entrarà un nombre enter per terminal i aleshores el programa crearà un tauler $n \times n$ (de tipus `char`) amb n bombes col·locades aleatòriament. L'usuari ha de, com a mínim, poder efectuar mitjançant un menú, les següents operacions durant una partida:
    1. Fer una aposta d'on es troba una bomba. Per fer-ho, es demanarà per pantalla dos nombres enters: fila i columna. Ara a la cel·la seleccionada apareixerà una 'B'.
    2. Descobrir el contingut d'una cel·la on,
        i. Si seleccionem una cel·la amb una bomba, perdrem la partida i finalitzarem el programa.
        ii. Si no conté cap bomba, el programa calcularà quantes bombes hi ha en les 8 cel·les immediatament contigües (les 4 a en vertical i

horitzontal, i les 4 en diagonal). Aquesta cel·la apareixerà amb el nombre de bombes calculat.

3. Imprimir per pantalla l'estat actual del joc,
    i. S'imprimirà un '?' si no sabem que hi ha en una cel·la.
    ii. S'imprimirà una 'B' si hem seleccionat aquella cel·la com a cel·la amb bomba.
    iii. S'imprimirà un enter de 0 a 8 si hem descobert aquella cel·la (i no conté cap bomba) indicant el nombre de bombes que té al seu costat.
4. Resoldre la partida. Això és, comprovar si hem col·locat les 'B' sobre les cel·les amb bomba i no ens hem deixat cap.
    i. Si s'ha realitzat tot correctament apareixerà un missatge indicant que hem guanyat la partida.
    ii. Si ens hem deixat alguna bomba sense marcar, apareixerà un missatge indicant que hem perdut la partida, i ens mostrarà on estaven les bombes.

    En ambdós casos, un cop realitzada aquesta opció, finalitzarem el programa correctament.
5. Sortir de la partida sense resoldre-la.

Per a realitzar aquest exercici, partireu del fitxer **Buscamines.java** que trobareu al Caseine on trobareu implementada la part d'impressió per pantalla. Haureu d'implementar els següents mètodes:

- `static char[][] createLayout(int n, char c)`

  Aquest mètode retornarà un array bidimensional de tipus `char`, de dimensió `nxn` on a cada cel·la hi col·locarà el caràcter `c` passat per paràmetre.

- `static void placeBombs(char[][] layout, int n, int numBombs):`

  Aquest mètode pren un array bidimensional (que representa el tauler), la dimensió `n` de l'array i el nombre de bombes que hem de col·locar en l'array, `numBombs`, i retorna un array bidimensional `numBombs` bombes col·locades aleatòriament. Tingueu en *compte* que cada cel·la de l'array pot tenir, com a molt, una bomba.

  La crida següent genera un nombre aleatori entre `min` i `max` (ambdós inclosos):

```
import java.util.concurrent.ThreadLocalRandom;

int nombreA;
nombreA = ThreadLocalRandom.current().nextInt(min, max + 1);
```

- `static void betBomb(char[][] layout, int n, Scanner sc)`

  Aquest mètode pren un array bidimensional, la dimensió `n` de l'array `nxn` i un objecte Scanner, `sc`, com a entrada. Aleshores demana a l'usuari dos índexs per a marcar una cel·la com a cel·la amb bomba. Tingueu en compte que no podeu tenir més apostes de bomba que número de bombes i que heu d'assegurar que la consulta a l'array bidimensional es fa dintre dels límits definits (de 0 a n-1 en

ambdues dimensions). Useu el mètode `checkIndexes()` , definit avall, per a fer això.

- `static int discoverCellAndCheckNeighborhood(char[][] layout, char[][] bombs, int n, Scanner sc)`

  Aquest mètode pren un array bidimensional, `layout,` la dimensió `n` i un objecte Scanner, `sc`. Demanarà a l'usuari dos índexs per a descobrir una cel·la. Si la cel·la seleccionada conté una bomba, doneu un missatge indicant que heu perdut i finalitzeu el programa. Si no conté una bomba, compteu el nombre de bombes que hi ha les 8 cel·les immediatament contigües (les 4 a en vertical i horitzontal, i les 4 en diagonal) i col·locareu aquest nombre a la cel·la seleccionada. Tingeu en compte que heu d'assegurar que els índexs demanats es troben dintre dels límits de l'array bidimensional. Useu el mètode `checkIndexes()` , definit avall, per a fer això.

- `static void resolve(char[][] layout, char[][] bombs, int n)`

  Aquest mètode pren dos arrays bidimensionals, `layout i bombs`, i la dimensió `n` dels arrays `nxn` com a entrada. Comprovarà si en el primer array bidimensional s'han marcat correctament totes les caselles com a bomba, comparant-ho amb el segon array bidimensional (que conté les bombes).

- `static int [] checkIndexes(int i, int j, int n, Scanner sc)`

  Aquest mètode pren dos índexs `i, j,` un enter `n` i un objecte Scanner, `sc`, com a entrada. Aleshores anirà demanant índexs a l'usuari mentre els dos índexs i, j no estiguin en el rang [0, n-1]. Si estiguessin en el rang, retorna un array null, sinó retorna un array de dues posicions on s'emmagatzemen els nous valors de i i de j.

A part heu d'omplir els corresponents cases al `switch` del `main()`.

15. Feu un programa per a jugar al **penjat** (juego del ahorcado) per a dos jugadors. L'objectiu és tractar d'endevinar el mot que ha pensat l'altre jugador (l'ordinador). Com a pista es mostra la paraula en construcció. Per exemple, si ha esbrinat dos lletres, llavors la paraula s'imprimeix com "ME??"el nombre de lletres. El jugador va dient lletres i si són en el mot secret, s'escriuen al seu lloc i si no s'incrementa el nombre d'intents fallits (PENJAT=6).
    Serà una versió simplificada que no dibuixarà el penjat però que donarà un màxim d'intents al jugador per encertar la paraula. El jugador guanya la partida si endevina la paraula sense arribar a "penjar-se" i perd si arriba al màxim d'intents.

    Feu ús dels següents tipus de dades:
    - `String [] llistaParaules`, una llista de paraules des d'on l'ordinador selecciona de forma aleatòria a cada partida la paraula a encertar.
    - `boolean [] lletresEncertades`, per a guardar les posicions de les lletres encertades pel jugador.
    - `String paraulaEndevinar`, és la paraula a endevinar, que l'agafarem del vector `llistaParaules`.

Implementeu els següents mètodes:

- `static void initLlistaDeParaules(String [] llistaParaules)`

  Aquest mètode inicialitza el vector de paraules del joc. La mida del vector `llistaParaules` és una constant (NUM_PARAULES=10) i les paraules són decisió vostra.

- `static String novaPartida(String [] llistaParaules)`

  Aquest mètode selecciona des de `llistaParaules` la paraula a esbrinar en una partida. La selección es fa de forma aleatòria. El mètode retorna la paraula a esbrinar.

- `static void initLletrasEncertades(boolean [] lletresEncertades, String paraulaEsbrinar)`

  Aquest mètode inicialitza el vector `lletresEncertades` posant el valor de cada posició a false. El tamany del vector depèn de la paraula a esbrinar, que és un altre paràmetre del mètode, `paraulaEsbrinar`.

- `static boolean lletraEncertada (String paraulaEsbrinar, int pos, char c, boolean [] lletresEncertades)`

  Aquest mètode retorna true si la lletra `c` a la posició `pos` es correspon a la lletra de la mateixa posició a la paraula a esbrinar, `paraulaEsbrinar`. Aquest mètode també actualitza el vector `lletresEncertades`, assignant el valor de true a la posició que toqui, si ho ha encertat el jugador.

- `static public String mostraParaula()`

  Aquest mètode construeix la paraula durant el joc per a donar-li feedback a l'usuari. Aquesta paraula tindrà les lletres esbrinades i el símbol "?" per les posicions que no s'han esbrinat encara. Ho podeu fer amb un `StringBuffer`.

- `static public boolean jocAcabat()`

  Aquest mètode retorna cert si el joc s'ha acabat.

16. Feu un programa per a jugar a un joc de negocis. El joc consta de 10 dies. El jugador comença amb una quantitat fixada de diners (DINERS_INICI). Cada dia, el jugador pot vendre o comprar pomes i peres. Hi ha un preu inicial (PREU_BASE) per les pomes i el mateix per les peres, cada dia aquest preu puja o baixa (aleatòriament) una quantitat fixa (VARIACIO). Un jugador no pot comprar si no té diners, i no pot vendre un producte que no té. L'objectiu es guanyar el màxim de diners possible. Mitjançant un menú es pot consultar el preu dels productes, els diners que té el jugador, i es pot comprar o vendre. (JocNegocis.java)
    Mètodes a implementar (penseu vosaltres mateixes els paràmetres i tipus de retorn):
    `calcularPreu(), vendrePomes(), vendrePeres(), compraPomes(), compraPeres(), menu().`

17. Feu un programa per a jugar al **tres en ratlla**. En aquest joc juguen dos jugadors, en el nostre cas una persona que s'enfrontarà a la màquina. El joc es juga a un tauler de 3x3. Les fitxes dels jugadors són diferents, per tal de distingir-les. A cada moviment, els jugadors posicionen una fitxa a una posició no ocupada del tauler. El joc consisteix en fer una fila, columna o diagonal amb totes les fitxes del mateix tipus (jugador / màquina). El joc finalitza quan un dels jugadors ha assolit l'objectiu, o bé quan no hi ha més moviments possibles – és a dir, el tauler està ple (TresEnRatlla.java).

    Penseu vosaltres mateixos els mètodes a implementar (nom, paràmetres i tipus de retorn).

18. Donat un array de nombres reals, escriu un mètode que ordeni els elements del vector de tal forma que els nombres parells apareguin abans que els nombres senars. A més a més, els nombres parells han d'estar ordenats de forma ascendent, mentre que els nombres senars han d'estar ordenats de forma descendent. Per exemple, l'array {1, 2, 3, 4, 5, 6} quedarà com {2, 4, 6, 5, 3, 1} (VectorOrdenatEspecial.java).

19. El propietari d'un bar ens ha demanat un programa per investigar en quins productes guanya més diners i en quins menys (GuanysBar.java). A més a més, també li agradaria saber si els diners superen la mitjana. Per a fer això, ha establert vàries categories:

Codi	Categoria
E	Esmorçars
D	Dinars
B	Berenars
S	Sopars
C	Copes

    - El propietari descriu cada venda dins d'una d'aquestes categories. Quan té un moment passa les dades de totes les vendes a l'ordinador i li agradaria que li retorni els següents valors: la categoria que més diners ha recaudat, la que menys, i si els diners aconseguits amb els dinars (D) supera a la mitjana.

    - El propietari introduirà les vendes realitzades i per cadascuna d'elles indicarà la categoria (E, D, B, S, C) i un valor real. Per finalitzar l'entrada de dades, el propietari introdueix una categoria inexistent (N) amb un valor 0 (és a dir, N 0).

    - El programa mostrarà per pantalla una línia que contindrà tres valors separats per una #. Els dos primers indicaran el nom de les categories que han suposat més i menys beneficis respectivament (tingueu en compte que si una categoria no ha venut res, el seu benefici és 0); les categories indicaran els seus noms, ESMORÇARS, DINARS, BERENARS, SOPARS, o COPES. El tercer valor de la línia indicarà "SI" si mitjana gastada pels clients en els dinars supera a la mitjana de les vendes del dia, i "NO" en cas contrari.

    - En el cas que hi hagi vàries categories que hagin aconseguit el màxim i el mínim de vendes, s'especificarà "EMPAT".

      Exemples d'entrada:

E 2.80
C 48.0
D 8.0
N 0

E 15.33
D 60.0
B 12.00
S 25.0
N 0

La sortida per les dues entrades anteriors seria:

COPES#EMPAT#NO
DINARS#COPES#SI

La primera línia de sortida indica que el màxim que s'ha venut ha sigut en COPES (48.0) i al mínim hi ha un empat perquè no hi ha vendes en dues categories (B i S) per tant hi ha EMPAT. Amb NO s'indica que la venda en dinars (D) no supera la mitjana de vendes.

La segona línia de sortida indica que el màxim que s'ha venut ha sigut en DINARS (60.0) i al mínim tenim COPES (0) perquè no hi ha vendes d'aquesta categoria. Amb SI s'indica que la venda en dinars (D) si supera la mitjana de vendes.

20. El joc "Hundir la Flota" o "Batalla naval" és un clàssic joc infantil de llapis i paper. Cada jugador disposa de dos taulers, formats per una quadricula de cel·les. Cada jugador col·loca la seva flota de vaixells. Cada vaixell ocupa un nombre variable de cel·les, i poden col·locar-se en vertical o horitzontal. Els vaixells no poden tocar-se. Anem a fer una variant del joc per detectar si la flota està correctament posicionada (FlotaCorrecteBatallaNaval.java).

- Com a màxim es podran posar 10 vaixells en un tauler.
- La mida màxima del tauler serà de 20 i sempre serà quadrat.
- En el tauler, una cel·la amb un valor de 0 indica aigua, una cel·la amb un valor 1 indicarà part d'un vaixell.
- **Entrada de dades:** A l'inici de l'entrada es demanen les dades del nombre de vaixells, després la mida dels vaixells, després la mida del tauler i finalment les dades de cadascuna de les files (0's o 1's). La primera posició del tauler és la (0,0).
- **Sortida de dades:** A la sortida, s'ha d'indicar si el tauler és correcte amb un "El tauler SI és correcte" o amb un "El tauler NO és correcte". Un tauler és correcte si conté tots els vaixells del joc descrits i amb la mida indicada. Els vaixells poden col·locar-se en vertical o horitzontal, i no poden tocar-se (ni tan sols en la seva diagonal). A més a més, a la sortida s'indica quines són les posicions ocupades pels vaixells. El joc continua fins que l'usuari indica una entrada amb 0 vaixells.

– Exemples d'entrada i sortida:

Al primer exemple d'entrada hi ha dos vaixells (2) de mida (2 3), la mida del tauler és 6 (6x6) i a continuació ens donen els valors al tauler. Aquest és un exemple de tauler correcte (els vaixells, indicats en negreta, no es toquen) per tant la sortida és "El tauler SI és correcte" i a continuació es mostren les posicions dels vaixells.

Al segon exemple d'entrada hi ha dos vaixells (2) de mida (2 3), la mida del tauler és 5 (5x5) i a continuació ens donen els valors al tauler. Aquest és un exemple de tauler NO correcte (els vaixells, indicats en negreta, es toquen a la diagonal) per tant la sortida és "El tauler NO és correcte" i a continuació es mostren les posicions dels vaixells.

Entrada	**Sortida**
2 2 3 6 0 **1 1** 0 0 0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 0 0 **1** 0 0 0 0 0 **1** 0 0 0 0 0 0 0 0	El tauler SI és correcte (0,1)(0,2) (2,3) (3,3) (4,3)
2 2 3 5 0 **1 1 1** 0 0 0 0 0 **1** 0 0 0 0 **1** 0 0 0 0 0 0 0 0 0 0	El tauler NO es correcte (0,1)(0,2)(0,3) (1,4) (2,4)
0	S'acaba el joc

21. Feu el programa anterior però aquesta vegada els dades estan emmagatzemats a un fitxer "taulersHundirFlota.txt" que heu de crear abans a un editor de text pla. (FlotaCorrecteBatallaNavalFitxer.java)

22. Feu un programa comprovar si dues seqüències d'ADN són idèntiques. Una cadena d'ADN es representa com una seqüència circular de bases (Adenina, Timina, Citosina i Guananina) que és única per a cada ésser viu. Per exemple:

A	T	G
T		C
A	T	G

Aquesta cadena es pot representar com un vector de caràcters recorrent-la en el sentit horari des de la part superior esquerra:

A	T	G	C	G	T	A	T

Escriu un programa que guardi un array que representi la seqüència d'ADN i inclogui un mètode booleà que ens retorni true si dues seqüències coincideixen (AdnIgual.java).

MOLT IMPORTANT: La seqüència d'ADN és cíclica, per tant pot començar en qualsevol posició. Per exemple, les dues seqüències que venen a continuació coincideixen:

A	T	G	C	G	T	A	T

A	T	A	T	G	C	G	T