Col·lecció de Problemes C - Taules, Tuples i Mètodes

Laboratori, Programació I, 2012

En tots els problemes només feu servir taules si és necessari. A més, aquestes haurien de ser de la mida mínima necessària.

1. (`Estadistiques.java`) Feu un programa que, donats un nombre natural n i una seqüència de n nombres reals, calculi la seva esperança i la seva variança. (L'esperança $\mathbb{E}[X]$ és la mitjana de la seqüència i una fórmula per a la variança és $\text{Var}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$).

   Entrada:
   ```
   10
   5 -1.2 7.5 0 2.35 6 5 1.7 2 2
   ```

   Sortida:
   ```
   E[X] = 3.0349999999999997, Var[X] = 6.799025.
   ```

2. (`Mitjana.java`) Feu un programa que, donats un nombre natural n i una seqüència de n notes (nombres reals no negatius), calculi el nombre de notes per sota la nota mitjana.

   Entrada:
   ```
   8
   8.9 7.5 0 9.8 7.85 8 8.1 10
   ```

   Sortida:
   ```
   2
   ```

3. (`Conjunts.java`) Feu un programa que, donats un nombre natural n i dues seqüències de com a màxim n nombres enters acabades en 0, indiqui si les dues contenen els mateixos números encara que potser en ordre diferent i amb repeticions.

   Entrada:
   ```
   10
   5 1 2 5 7 1 8 12 0
   7 12 12 5 1 8 2 0
   ```

   Sortida:
   ```
   Sí
   ```

4. (`MultiConjunts.java`) Feu un programa que, donats un nombre natural n i dues seqüències de com a màxim n nombres enters acabades en 0, indiqui si les dues contenen els mateixos números i amb les mateixes freqüències.

   Entrada:
   ```
   10
   5 1 2 5 7 1 8 12 0
   7 12 12 5 1 8 2 0
   ```
   Sortida:
   ```
   No
   ```

   Entrada:
   ```
   10
   5 1 2 5 7 1 8 12 0
   7 12 5 5 1 8 1 2 0
   ```
   Sortida:
   ```
   Sí
   ```

5. (`ParaulaFrequent.java`) Feu un programa que donats un nombre natural n i una seqüència de n paraules en minúscules, indiqui quina és la paraula més freqüent.

   Entrada:
   ```
   17
   pa amb oli i pa amb xocolata i pa amb vi i pa amb tomaquet i pa
   ```
   Sortida:
   ```
   pa
   ```

6. (`ComptaLletres.java`) Feu un programa que llegeixi un text format per paraules escrites en minúscules (lletres entre 'a' i 'z', sense puntuació, ni accents) fins que trobi la paraula "fi" i indiqui quina lletra és la més freqüent (la paraula "fi" no compta). Us pot servir el fet que els codis de les lletres segueixen l'ordre alfabètic.

   Entrada:
   ```
   the quick brown fox jumps over the lazy dog fi
   ```
   Sortida:
   ```
   o
   ```

7. (`Cognoms.java`) Feu un programa que, donats 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 dos estudiants que tenen algun cognom en comú i imprimeixi les dades dels primers dos que trobi.

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

   Sortida:
   ```
   Manuel Prats Lopez 87652211R
   Mireia Lopez Roura 12345678R
   ```

8. (`CercleDominant.java`) Feu un programa que, donats un nombre natural n i una seqüència de n cercles, cadascun especificat per les coordenades del seu centre i el seu radi, indiqui si algun dels cercles conté tots els altres al seu interior. La classe `Cercle` hauria de tenir un mètode `boolean esDominant(Cercle c)` que, donat un altre cercle, indiqui si aquest està a l'interior.

   Entrada:
   ```
   4
   1 1 2.5   -0.5 0 1   0 0 5.5   -1 -1 0.5
   ```

   Sortida:
   ```
   Sí: 0.0 0.0 5.5
   ```

9. (`IntervalDominant.java`) Feu un programa que, donats un nombre natural n i una seqüència de n intervals tancats, indiqui si algun dels intervals conté tots els altres.

   Entrada:
   ```
   4
   -1 2   -1.5 0   -2 2   0 1
   ```

   Sortida:
   ```
   Sí: [-2.0, 2.0]
   ```

10. (`MultMatriuVector.java`) Feu un programa per multiplicar una matriu per un vector. L'entrada comença amb les dimensions de la matriu.

    Entrada:
    ```
    2 3
    1    0 -1
    0.5  1  0
    1
    2.5
    0
    ```

    Sortida:

```
1.0
3.0
```

11. (`MultMatrius.java`) Feu un programa per multiplicar dues matrius. L'entrada comença amb les tres dimensions de les matrius.

    Entrada:
    ```
    3 2 4
    1    0
    0.5  1
    1.5 -1
    1 0 -1.5 1
    0 2  0  -1
    ```

    Sortida:
    ```
    1.0 0.0 -1.5 1.0
    0.5 2.0 -0.75 -0.5
    1.5 -2.0 -2.25 2.5
    ```

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

    Entrada:
    ```
    4 2
    Salvador Dali Domenech 7.8
    Merce Rodoreda Gurgui 9.5
    Carles Puyol Saforcada 7
    Shakira Mebarak Ripoll 8.6
    ```

    Sortida:
    ```
    Merce Rodoreda Gurgui 9.5
    Shakira Mebarak Ripoll 8.6
    ```

13. (`FraseMonovocalica.java`) Creeu un programa que llegeixi una vocal i una frase en minúscules i acabada en el sentinella "fi" i converteixi la frase en mono vocàlica convertint cada vocal de la frase en la vocal donada. Per fer-ho caldrà crear un mètode `esVocal`.

    Entrada:
    ```
    o
    una mosca volava per el llum i el llum es va apagar fi
    ```

    Sortida:
    ```
    ono mosco volovo por ol llom o ol llom os vo opogor
    ```

14. (`MenuBicicletes.java`) Creeu un programa per gestionar la informació de com a molt 25 bicicletes. Cada bicicleta té un conjunt de dades: model, pes, si té suspensió o no, i preu. Es vol programar una solució que permeti:

    - *Afegir bicicleta* → Demanarà el model, el pes, si té suspensió i el preu de la bicicleta.
    - *Bicicleta de cost preu per kg minim* → Imprimirà la informació de la bicicleta que tingui mínim cost per kilogram.
    - *Cerca de la bicicleta inferior a un cert pes* → Imprimirà una bicicleta de pes inferior a un pes entrat per l'usuari, si és que n'hi ha alguna.
    - *Sortir* → El programa acaba.

15. (`Edats.java`) Creeu un programa que mostri el següent menú per pantalla en iniciar-se:

    - *Afegir edat* → Demanarà un nom (p.e: "Javi"), i a continuació una edat (p.e: 29). Creeu un objecte `Persona` per guardar aquestes dades.
    - *Veure edat* → El programa demanarà un nom i mostrarà per pantalla l'edat d'aquella persona. En cas que no es trobi ho mostrarà per pantalla.
    - *Veure persones amb edat determinada* → El programa demanarà una edat i a continuació mostrarà per pantalla totes les persones amb l'edat introduïda.
    - *Veure majors de* → El programa demanarà una edat i a continuació mostrarà per pantalla totes les persones amb edat major a la introduïda.
    - *Sortir* → El programa acaba.

    Implementeu els mètodes d'objecte `afegir(String nom, int edat)`, `int veure(String nom)`, `veurePersonesAmbEdat(int edat)`, `veureMajorsDe(int edat)`.

16. (`GestioCinema.java`) Feu un programa que gestioni les butaques de la sala d'un cinema. Inicialment el programa demanarà el número de files de la sala i el número de butaques per cada fila. A continuació posarà totes les butaques inicialment com a lliures i l'usuari podrà:

    - *Ocupar butaca* → El programa demanarà fila i butaca i la posarà com a ocupada. Si ja estava ocupada, donarà el missatge a l'usuari.
    - *Lliurar butaca* → El programa demanarà fila i butaca i la posarà com a lliure. Si ja estava lliure, donarà el missatge a l'usuari.
    - *Lliurar totes les butaques* → El programa posarà totes les butaques com a lliures.
    - *Sortir* → El programa acaba.

    Per facilitar la gestió del programa, podeu implementar els mètodes `ocuparButaca(int fila, int butaca)`, `lliurarButaca(int fila, int butaca)`, `lliurarTotesButaques()`.

17. (`Set.java`) El joc *Set* conté 81 cartes amb figures. Les cartes tenen 1, 2, o 3 figures de forma de rombe, d'òval o d'ona, de color vermell, verd o blau i amb textura sòlida, a ratlles o buida. Per tant cada carta té 4 característiques: color, número, forma i textura. Un *set* és un conjunt de tres cartes amb la propietat que, de cada característica, les cartes són o bé totes diferents o bé totes iguals. Feu un programa que, donat un conjunt de cartes, trobi algun set dins d'aquest conjunt si n'hi ha (i diu que no n'hi ha, si no). Per llegir les cartes, suposem que cada carta està descrita amb quatre xifres, una per cada característica, i aquestes poden tenir el valor 1, 2 o 3 (per exemple 2121 vol dir 2 rombes verds amb textura sòlida). Implementeu un mètode `boolean esCarta(int carta)` que, donat un nombre enter, indiqui si aquest correspon a una carta del joc Set, i un mètode `boolean esSet(int carta1, int carta2, int carta3)` que, donat 3 nombres enters, indiqui si aquests formen un set.

    Entrada:
    ```
    12
    1221 3123 3312 1111 3222 2222 1213 3322 1121 2121 3133 1312
    ```

    Sortida:
    ```
    Cartes número 4, 7 i 12 formen un set.
    ```

18. (`Quadrats.java`) Feu un programa que, donats un nombre natural n i una seqüència de n quadrats, indiqui si hi ha algun quadrat que intersequi tots els altres. Els quadrats tenen els costats paral·lels als eixos de referència i estan especificats per les coordenades del vèrtex de més avall i més a l'esquerra i la mida del costat.

    Entrada:
    ```
    4
    1 1 2.5   -0.5 0 1   0 0 5.5   -1 -1 3.5
    ```

    Sortida:
    ```
    Sí: 0.0 0.0 5.5
    ```

19. (`Cercles.java`) Feu un programa que, donats un nombre natural n i una seqüència de n cercles, cadascun donat per les coordenades del seu centre i el seu radi, indiqui quins cercles s'intersequen. Implementeu un mètode `sIntersequen` que calculi si dos cercles s'intersequen.

    Entrada:
    ```
    4
    1 1.5 2.5   1 1 0.5   2.5 1.5 1.5
    ```

    Sortida:
    ```
    1 i 3
    2 i 3
    ```

20. (`MitjanaEnters.java`) Feu un programa que, donada una seqüència de notes enteres entre 0 i 10 acabada amb -1, calculi el nombre de notes per sota la nota mitjana.

    Entrada:
    ```
    10 9 10 7 6 0 9 9 10 4 5 8 -1
    ```

    Sortida:
    ```
    5
    ```

21. (`NotaMediana.java`) Feu un programa que, donada una seqüència de notes enteres entre 0 i 10, acabada amb -1, calculi la nota mediana.

    Entrada:
    ```
    9 9 0 10 0 1 1 2 6 -1
    ```

    Sortida:
    ```
    2.0
    ```

    Entrada:
    ```
    9 9 0 10 0 1 1 2 5 6 -1
    ```

    Sortida:
    ```
    3.5
    ```

22. (`Anagrames.java`) Feu un programa que, donada una seqüència de com a molt 100 frases acabada en "fi", indiqui quins parells de frases són anagrames. Implementeu un mètode `esAnagrama` que calculi si dues frases són anagrames.

    Entrada:
    ```
    tom marvolo riddle
    madonna louise ciccone
    one cool dance musician
    i am lord voldemort
    fi
    ```

    Sortida:
    ```
    tom marvolo riddle = i am lord voldemort
    madonna louise ciccone = one cool dance musician
    ```

23. (`Postfix.java`) Feu un programa per calcular expressions en notació Polaca Inversa o postfix, considerant només les operacions de suma '+' i producte '*'. Quan utilitzem aquest tipus de notació per descriure expressions matemàtiques, escrivim els operands primer, i l'operació matemàtica a continuació. Per exemple, $a + b$ s'escriuria $a\ b\ +$. Amb aquest tipus de notació, ens evitem la utilització de parèntesis, ja que queda perfectament delimitat. Per exemple, $(a+b)*(c+d)$ ho podem escriure com $a\ b + c\ d + *$. Una manera de resoldre operacions amb aquesta notació, és la utilització d'una pila. L'algorisme és el següent:

a) Si arriba un nombre, el posem a la pila.

b) Si arriba un operador, traiem dos elements de la pila, apliquem l'operador sobre ells i posem el resultat a la pila.

c) Si s'acaba l'expressió, i era correcta, a la pila només quedarà un valor, el qual serà el resultat.

Imagineu que volem calcular 1 5 4 3 + ∗ +, els passos que s'anirien succeint, i l'estat de la pila en cada moment es mostren a continuació:

a) Entra 1 → Apilem a la pila → [1]

b) Entra 5 → Apilem a la pila → [1 5]

c) Entra 4 → Apilem a la pila → [1 5 4]

d) Entra 3 → Apilem a la pila → [1 5 4 3]

e) Entra operador + → Desempilem a la pila els dos últims elements i els substituïm pel resultat de l'operació → [1 5 7]

f) Entra operador * → Desempilem a la pila els dos últims elements i els substituïm pel resultat de l'operació → [1 35]

g) Entra operador + → Desempilem a la pila els dos últims elements i els substituïm pel resultat de l'operació → [36]

Si no ens venen més dades, el resultat és l'element que ha quedat a la pila. Podeu suposar que la mida màxima de la pila és de 100 operacions, que l'expressió és legal i que acaba amb el símbol 'q'. Els nombres són enters.

24. (`Peons.java`) Feu un programa que, donada una seqüència de posicions de peons blancs i negres en un tauler d'escacs, totes diferents, i acabada amb la paraula "fi", indiqui si existeix algun peó que amenaci algun altre.

    Entrada:
    ```
    wA5 bF2 bH7 wB3 wA6 bB4 wB3 wB5 bC6 bB6 bH8 fi
    ```

    Sortida:
    ```
    Position B3 is already occupied.
    Black at C6 threatens white at B5.
    ```