UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Politècnica Superior d'Enginyeria
de Vilanova i la Geltrú

PROGRAMACIÓ 1 **PRÀCTICA**
CURS: 2021/2022-Q2

1 Codificació Huffman

La codificació Huffman és una codificació utilitzada per a la **compressió de dades** desenvolupada per David A. Huffman l'any 1952 i descrita en *A Method for the Construction of Minimum-Redundancy Codes*[1].

La sortida de l'algorisme de Huffman pot veure's com una taula de codis de longitud variable usada per codificar un conjunt de símbols, com poden ser els caràcters d'un fitxer de text. La longitud de cada codi depèn de la freqüència relativa d'aparició de cada símbol en un text: **com més freqüent sigui un símbol en un text, més curt serà el seu codi associat**. A més, un codi de Huffman és un codi lliure de prefixos (també anomenat codi prefix), és a dir, cap codi forma la primera part d'un altre codi. Això permet que els missatges codificats siguin no ambigus.

La compressió és més gran quan la varietat de símbols diferents que apareixen en el text és menor. Per exemple, si un text està format únicament per números o per lletres majúscules, s'aconsegueix una major compressió.

Un cop codificat un text, per recuperar el text original cal conèixer el codi assignat a cada caràcter. Si aquesta informació s'omet i el receptor del text la coneix, podrà recuperar el text original. D'aquesta manera, és possible usar la codificació Huffman per encriptar dades.

2 L'algorisme

Huffman va descriure un algorisme per obtenir codis de Huffman a partir d'un conjunt de símbols i les seves freqüències associades. A continuació, es descriuen els passos en què es basa l'algorisme:

1. Fer un recompte de les vegades que apareix cadascun dels símbols en el fitxer a comprimir i crear una taula (per exemple) que contingui la informació de cada símbol juntament amb la seva freqüència associada i la seva codificació Huffman (inicialment, buida).
2. Convertir cada element de la taula en un arbre binari de manera que l'element sigui el valor de l'arrel i els seus dos fills estiguin buits.

[1] http://compression.ru/download/articles/huff/huffman_1952_minimum-redundancy-codes.pdf

3. Inserir tots els arbres en una cua amb prioritat on la prioritat és la freqüència del valor a l'arrel de cada arbre, tenint en compte que han de quedar ordenats de **menor a major freqüència**.
4. Fusionar tots els arbres de la cua en un únic arbre. Per fer-ho, es repeteix el següent procediment mentre la cua d'arbres contingui més d'un arbre:
   (a) Formar un nou arbre amb els dos primers arbres de la cua (els dos que tenen menor freqüència), de manera que cada un dels arbres originals sigui un fill del nou arbre. El primer arbre extret serà el fill esquerre i el segon el fill dret. Aquests dos arbres s'eliminen de la cua.
   (b) Etiquetar l'arrel del nou arbre amb un element tal que la seva freqüència sigui la suma de les freqüències de les arrels dels subarbres que s'uneixen. El símbol de l'element de l'arrel del nou arbre no és rellevant.
   (c) Inserir el nou arbre en la cua amb prioritat.
5. Per assignar el codi de Huffman a cada símbol només hem de seguir el camí adequat dins l'arbre final. Si s'agafa una branca zero, que considerarem que és l'esquerra, s'afegeix un 0 al codi; si s'agafa una branca u, la dreta, s'afegeix un 1.
6. Codificar el text del fitxer usant la codificació Huffman obtinguda.

3 Exemple

Suposem que tenim un fitxer de text format per 100 000 caràcters que volem transmetre. El text del fitxer només conté 6 caràcters diferents que apareixen amb freqüències diverses.

A la Taula 1 podem veure els símbols (caràcters) a transmetre, les seves freqüències relatives, el seu codi en una codificació binària i el seu codi en una possible codificació Huffman segons els valors de les seves freqüències.

Símbol	Freqüència relativa	Codi binari	Codi de Huffman
a	0.10	000	1011
b	0.20	001	00
c	0.30	010	11
d	0.25	011	01
e	0.10	100	100
f	0.05	101	1010

Taula 1: Exemple de codificació binària i codificació Huffman per un conjunt de símbols.

Podem veure que, amb la codificació binària, tots els símbols reben codis del mateix nombre de bits (*fixed-length code*), mentre que amb la codificació Huffman cada símbol té associat

un codi amb un nombre de bits diferent (*variable-length code*): els símbols més freqüents tenen codis de dos bits, mentre que els menys freqüents tenen codis de tres o quatre bits.

Pel nostre fitxer de 100 000 caràcters, el codi de longitud fixa (3 bits) necessitaria 300 000 bits, mentre que el de longitud variable només en necessitaria 240 000.

Si volguéssim transmetre un fitxer amb el contingut següent:

```
fecdccbabdcededcdece
```

- usant la codificació binària, obtindríem una sèrie de 60 bits, és a dir, 3 bits per símbol

  `101100010011010010001000001011010100011100011010011100010100`

- usant la codificació Huffman, s'hauria d'enviar una seqüència de 49 bits, és a dir, 2.45 bits per símbol.

  `1010100110111110010110001111000110001110110011100`

En aquest exemple, la mitjana de bits per símbol que seria d'esperar per aquesta codificació, en seqüències de valors més llargues, és de:

$$4*0.10 + 2*0.20 + 2*0.30 + 2*0.25 + 3*0.10 + 4*0.05 = 2.4 \text{ bits}$$

4 Es demana

Es demana que dissenyeu un *programa modular* que, donat un fitxer de text, el codifiqui fent servir la codificació Huffman seguint el procediment descrit en l'apartat 2.

4.1 Classes

Per resoldre aquesta pràctica ja se us dona l'especificació (incompleta) d'algunes de les classes. Caldrà que completeu les seves especificacions afegint el mètodes que us calguin en funció de la implementació triada pel tipus. Busqueu el text `COMPLETEU` en els fitxers `.hpp` corresponents. Després, implementeu els mètodes en els fitxers `.cpp`.

- `Entrada.hpp`: Ha de representar un símbol, la seva freqüència i la seva codificació Huffman.
- `TaulaFreq.hpp`: Ha de representar una estructura de dades on emmagatzemar un conjunt de símbols i la seva informació associada.
- `Huffman.hpp`: Ha de representar la informació necessària per obtenir una codificació de Huffman. Les operacions per generar l'arbre de codificació (pas 4) i per recórrer l'arbre de codificació (pas 5) han de ser **recursives**.

També se us dona l'especificació i la implementació de dues classes que haureu de **modificar** o **completar**. Busqueu el text `PROGRAM THIS METHOD` en els fitxers `.hpp` corresponents.

- `PriorityQueue.hpp`: Partint de la classe `Queue` amb què heu treballat, cal modificar el mètode `push` per tal que es comporti com una cua amb prioritat de manera que els elements que s'afegeixin a la cua ho facin per ordre de "menor" a "major" del seu valor, en lloc de per ordre d'arribada.
- `BinaryTree.hpp`: Cal afegir un mètode que us permeti comparar dos arbres binaris redefinint l'operador $<$.

5 Normativa, avaluació i lliuraments

5.1 Normativa

1. La pràctica és **obligatòria**. La no presentació de la pràctica resultarà en un **NO PRESENTAT** com a nota final de l'assignatura.
2. S'ha de realitzar en grups de **màxim dues persones**. Serà motiu de no acceptació del treball l'haver estat realitzat per més de dues persones. La formació de grups de pràctiques queda al vostre criteri.
3. La qualificació final de la pràctica té un pes del 30% sobre la nota total de l'assignatura. La nota de la pràctica està formada per dues parts: programa (1 lliurament) i prova de validació **individual** (PVI). La nota de la PVI serà un número real entre 0 i 1. La fórmula per calcular la nota final de la pràctica és la següent: nota pràctica = nota programa $*$ nota PVI.

   La no compareixença d'un/a estudiant a la PVI farà que la seva nota de la pràctica sigui NO PRESENTAT, encara que hagi fet la resta dels lliuraments.
4. Si dos o més treballs presentats són considerats **iguals**, no serà acceptat **cap** dels treballs.

5.2 Avaluació

Les pràctiques acceptades s'avaluaran mitjançant:

- La correctesa de l'especificació i/o comentaris del codi. Atenció especial amb:
  - les Pre/Post de les operacions més importants
  - les capçaleres dels mètodes pel que fa al bon ús de `const` (tant sobre mètodes com sobre paràmetres)
  - l'ús de tipus "pesants" en el pas de paràmetres per valor o de retorn de funcions.

- Els mètodes pertanyen a les classes adequades. Es fa una bona distinció entre mètodes privats i públics.
- L'execució del programa.

Una pràctica que no funcioni pot ser igualment acceptada i avaluada tot i que, si no passa cap joc de prova, la nota de la pràctica serà un 0.

5.3 Lliurament

1. El lliurament s'ha de fer únicament a través d'Atenea en la Tasca prevista per fer-ho.
2. Un lliurament consistirà en un únic fitxer comprimit, amb nom **Pr-dni1-dni2.zip** format pels DNI dels membres del grup, ordenats de menor a major. Aquest fitxer contindrà **únicament** els fitxers de codi necessaris per compilar la pràctica.
3. Només cal fer un enviament per grup.
4. Se us facilitaran un conjunt de jocs de prova públics.

5.3.1 Data

Aquesta pràctica, a partir de la seva data de publicació, tindrà un seguiment setmanal dins les hores de laboratori. És a dir, una part de cada sessió de laboratori es dedicarà a poder-la implementar i resoldre els dubtes que tingueu.

Data límit pel lliurament: 5 de juny de 2022 a les 23:59