Vocabulari

- **Nom de la classe:** Vocabulari
- **Descripció:** Classe que conté el llistat de totes les paraules que els documents tenen.
- **Cardinalitat:** 1 instància
- **Descripció dels atributs:**
    - **arrel:** Arbre que conté totes les paraules dels documents.
- **Descripció dels mètodes:**
    - **Vocabulari()** Constructora per defecte de Vocabulari.
    - **decrementarOcurrencia(Paraula p)** Decrementa en una unitat el nombre d'ocurrencies de la paraula p.
    - **esborrarParaula(Paraula p)** Esborra una paraula del diccionari.
    - **Paraula inserirObtenirParaula(String s)** Obtenir la classe Paraula que correspon a la seqüència s, la crea si no existeix aquesta.
    - **Paraula obtenirParaula(String s)** Obtenir la classe Paraula que correspon a la seqüència s.
- **Descripció de les relacions:**
    - Relació d'associació amb la classe "Paraula": indica quines paraules formen part del vocabulari.

Paraula

- **Nom de la classe:** Paraula
- **Descripció:** Paraula que pot ser continguda a algun document.
- **Cardinalitat:** 0 o més instàncies
- **Descripció dels atributs:**
    - **index:** Identificador de la paraula.
    - **mot:** La pròpia paraula en format String.
    - **ocurrència:** Número de vegades que la paraula apareix entre tots els continguts.
    - **static proxim_index:** Pròxim identificador a asignar.
- **Descripció dels mètodes:**
    - **Paraula(String p)** Constructora per defecte de paraula.
    - **decrementarOcurrencia()** Decrementa en una unitat el nombre d'ocurrencies.
    - **int getId()** Retorna l'índex de la paraula.
    - **int getOcurrencia()** Retorna el nombre d'ocurrencies total.
    - **String getParaula()** Retorna la paraula representada per la classe.
    - **incrementarOcurrencia()** Incrementa en una unitat el nombre d'ocurrencies.
- **Descripció de les relacions:**
    - Relació d'associació amb la classe "Vocabulari": indica el vocabulari on aquesta paraula està inclosa.

- Relació d'associació amb la classe "Contingut": indica els continguts on apareix la paraula.

Contingut

- **Nom de la classe:** Contingut
- **Descripció:** Classe que conté el contingut d'un document.
- **Cardinalitat:** 1 instància per a cada document
- **Descripció dels atributs:**
  - **plaintext:** Emmagatxema tot el text del contingut en un sol String.
  - **phrases:** Array de Frases que formen el contingut.
  - **words:** HashMap que mapeja cada índex de paraula amb el seu nombre d'ocurrències al contingut.
  - **n_paraules:** Nombre de paraules total al contingut.
- **Descripció dels mètodes:**
  - **Contingut (String plaintext, Frase [ ] phrases):**
    Donat el plaintext i l'array de Frases, construeix el Contingut
  - **double getTFofWord (int index)**
    Donat l'índex d'una paraula, retorna el Term Frequency d'aquesta al contingut (o -1 si no existeix la paraula al contingut).
  - **MAP<Integer, Double> getTF ()**
    Retorna un HashMap que mapeja cada índex de paraula amb el seu TF al contingut.
  - **boolean conteSequencia (String seq)**
    Retorna *true* si, i només si, la sequüència *seq* es troba en el text del contingut.
- **Descripció de les relacions:**

ConsultaSemblant

- **Nom de la classe:** ConsultaSemblant
- **Descripció:** Classe que implementa la consulta per semblança.
- **Cardinalitat:** 0 instàncies. La classe només té un mètode static.
- **Descripció dels atributs o relacions: -**
- **Descripció dels Mètodes:**
  - **LIST< PAIR < Double, Document > > executeQuery**
    (Llibreria lib, Document doc, int k, int mode)
    Únic mètode static de la classe ConsultaSemblant que implementa l'execució de la Query donada una llibreria (és a dir, un conjunt de documents), un document *doc*, un enter *k*, i un mètode (0 ó 1) per a assignar pesos a les paraules.
    El mètode retorna una llista de parells, on cada parell tenim un document i el seu cosinus amb el document *doc*. La llista són els *k*

documents més semblants a *doc*, on definim el concepte “semblança” com el cosinus entre els angles de dos vectors de paraules i pesos TF-IDF (mode 0) ó pesos Nombre d’Ocurrències (mode 1).

CtrlDomini

- **Nom de la classe:** CtrlDomini
- **Descripció:** Classe que conté mètodes relacionats amb la gestió de de les classes de la capa de Domini.
- **Cardinalitat:** 1 instància
- **Descripció dels atributs:**
    - **Vocabulari vocab:** El vocabulari de paraules creades
    - **Llibreria lib:** El conjunt de documents de l’aplicació
    - **ExpressioBooleanaCtrl EBC**: Controlador EBs
    - **CtrlPersistència DISK:** Controlador de Persistència
    - **CtrlPresentacio UI:** Referència al Controlador de Presentació (necessari per als pop-ups i la navegació inversa)
    - **(4X) Comparator<Document> documentComparator**
      Són quatre comparadors de Documents que serveixen per a ordenar-los segons dates, preferits, autors i títols.
- **Descripció dels mètodes:**
    - **boolean docExists(String titol, String autor)**
      Retorna cert si existeix el Document (titol, autor).
    - **LIST<Document> sortDocuments(LIST<Document>, int type)**
      Mètode que ordena i retorna una ArrayList de Document segons un criteri, especificat per l’enter *type*. (0 = per data de creació; 1 = preferits primer; 2 = autor alfabèticament; 3 = títol alfabèticament).
    - **LIST<Document> sortDocuments(SET<Document>, int type)**
      Mateix mètode que l’anterior, però admet com a paràmetre un conjunt de documents en comptes d’una llista.
    - **CtrlDomini ()**
      Constructora per defecte. Crea noves instàncies de Vocabulari, Llibreria, ExpressióBooleanaCtrl i CtrlPersistència.
    - **LIST<String> decomposeWords(String frase)**
      Mètode que descomposa una String en paraules. És a dir, separa cadenes de lletres per espais ó signes especials i llavors insereix aquestes paraules a una llista a retornar.
    - **PAIR<Document, Boolean> getDocument (nomAutor, nomTitol)**
      Mètode que retorna el Document identificat per l’String nomAutor i nomTitol. Si existeix el booleà del Pair que retorna serà *true*, altrament *fals*. En cas que sigui *fals* el document retornar serà *null*.
    - **void crearDocument (Str titol, Str autor, LIST<String> content, String ptxt_cont, DATE dia, boolean isFav)**

Crea una instància de Document amb el títol, autor, contingut, data i booleà preferit dels paràmetres.

- **Frase[ ] generatePhrases (String plaintext)**
  Donat un text, el separa en diferents frases (separació per signes de puntuació).
- **Contingut generateContent (String plaintext)**
  Donat un contingut en String, retorna un nou objecte Contingut (necessari per a crear noves referències al obrir Documents o consultar-ne)
- **boolean getFavourite (String title, String autor)**
  Mètode que retorna el booleà preferit del document (title, autor)
- **LocalDate getData (String title, String autor)**
  Mètode que retorna la data del document (title, autor)
- **String getContingut (String title, String autor)**
  Mètode que retorna l'String del contingut del document (title, autor)
- **String preview (String title, String autor)**
  Mètode que retorna l'String formatejada que visualment representa el document (title, autor). *(format title ~ autor)*
- **SET<String> donaAutors (String prefix)**
  Mètode que retorna el conjunt dels Autors en Strings que comencen per la String *prefix*.
- **void modificarData (String title, String autor, DATE newdate)**
  Mètode per a modificar la data del document (title, autor)
- **void modificarAutor (String title, String oldauthor, Str newauthor)**
  Mètode per a modificar l'autoria del document (title, autor)
- **void modificarTitol (String oldtitle, String author, Str newtitle)**
  Mètode per a modificar el títol del document (title, autor)
- **void modificarContingut (title, autor, LIST<String> cont, Str pltxt)**
  Mètode per a modifiar el contingut del document (title, autor) amb el plaintext *pltxt* i les frases a la llista *cont*
- **void eliminarDocument (String titol, String autor)**
  Elimina el document (titol, autor) de la llibreria
- **LIST<String> consultaData (DATE ant, DATE post, int criteria)**
  Mètode per a consultar els documents en el rang de dates.
- **LIST<String> consultaSeq (String seq, int criteria)**
  Mètode per a consultar els documents que tenen la seqüència *seq*.
- **LIST<String> getAllDocs()**
  Mètode per a consultar tots els documents existents
- **LIST<String> consultaTit (String author, int criteria)**
  Mètode per a consultar els documents de l'autor *author*.
- **LIST<String> consultaSemb (titol, autor, int n, int mode)**
  Mètode per a consultar els *n* documents més semblants a (titol, autor)
- **LIST<String> consultaRell (String words, int k, int modeCons)**

Mètode per a consultar els *k* documents més rellevants amb la query *words* i el mode de consulta *modeCons.*
    - **LIST<String> consultaPref (int criteria)**
      Mètode per a consultar els documents que són preferits.
    - **void novaEB(String nom, String cos)**
      Crea una nova Expressió Booleana amb el nom i cos donats.
    - **void canviarEB(String nom, String noucos)**
      Modifica el cos de l'Expressió Booleana "nom".
    - **void eliminarEB(String nom)**
      Elimina l'Expressió Booleana "nom".
    - **int numberOfEBs()** Retorna el nombre total d'EBs.
    - **boolean existsEB(String nom)** Retorna cert si l'EB "nom" existeix.
    - **LIST<String> consultaEB (Str cos, Str nom, int mode, int criteri)**
      Executa una consulta per Expressió Booleana. El mode indica si l'expressió existeix (nom) o bé si volem fer-ne una sense guardar-la (cos). Els documents s'ordenen pel *criteri*.
    - **void exportarDocument (titol, autor, int ext, String filename)**
      Exporta el document (titol, autor) amb l'extensió *ext* (0 = .txt; 1 = .xml; 2 = .yay) i el nom de fitxer *filename*,
    - **void importFile (String path)**
      Mètode per importar un fitxer ubicat al path absolut *path* (UNIX)
    - **void importSaved ( )**
      Importa tots els documents emmagatzemats a DATA/ d'una sessió anterior.
    - **void togglePreferit (String titol, String autor)**
      Fa un toggle del booleà preferit del document (titol, autor)

Expressio Booleana

- **Nom de la classe:** ExpressioBooleana
- **Descripció:** Expressió formada per combinacions de instancies d'operadors, paraules o expressions més breus. Les expressions requereixen una paraula o una altra expressió.
- **Cardinalitat:** 0 o més instàncies
- **Descripció dels atributs:**
    - **String cos:** Guarda l'expressió booleana tal qual arriba a l'instancia de la classe.
    - **String nom:** Guarda el nom associat a l'expressió booleana. Pot ser null.
    - **Node root:** Node arrel de l'arbre d'expressió generat per la classe ExpressionTree
    - **SET<Document> resultat:** Conjunt de instancies de la classe Document que només conté els documents que compleixen l'expressió booleana.

- **Descripció dels mètodes:**
    - **SET<Document> getResultat(Llibreria l):** Retorna els Documents que compleixen l'expressió booleana de Cos.
    - **ExpressioBooleana():** Creadora per defecte.
    - **ExpressioBooleana(String cos):** Creadora de la classe donada l'expressió.
    - **ExpressioBooleana(String nom, String cos):** Creadora de la classe donats el nom i l'expressió.
    - **ExpressioBooleana ExpressioBooleana (String cos):** Torna una instancia d'ExpressioBooleana donada una expressió.
    - **Set<Docment> ConsultaBooleana (Llibreria l, Node consulta):** Torna un conjunt de documents que compleixen l'expressió booleana donats una llibreria i un node arrel.
    - **Boolean ConteParaula (String query, Document doc):** Torna cert si el document doc conté la paraula query
    - **Set<Document>ConsultaBooleanaRec (Node consulta, Set<Document> tots):** Torna recursivament un conjunt de documents que contenen els documents que compleixen la condició del node donats un node i un conjunt de documents.

ExpressioBooleanaCtrl

- **Nom de la classe:** ExpressioBooleanaCtrl
- **Descripció:** Classe controlador que conté mètodes per a la creació, ús, eliminació i cerca d'expressions booleanes
- **Cardinalitat:** 1
- **Descripció dels atributs:**
    - **MAP<String, ExpressioBooleana> SetDeExpressions:** És un HashMap que conté totes les expressions booleanes i fa servir el seu nom com a identificador.

- **Descripció dels metodes:**
    - **int getNEBS():** Retorna el nombre d'EBs guardades.
    - **boolean existsEB(nom):** Retorna cert si existeix l'EB "nom".
    - **ExpressioBooleanaCtrl():** Creadora per defecte.
    - **Set<String> GetNomExpressions():** Torna un conjunt amb tots els identificadors de SetDeExpressions.
    - **ExpressioBooleana GetExpressioBooleana (String nom):** Torna l'instancia d'ExpressioBooleana corresponent al parametre.

    - **ExpressioBooleana ExpressioBooleanaTemporal (String cos):** Torna una nova instància d'ExpressioBooleana sense nom donat el seu cos.
    - **DeleteExpressioBooleana(String nom):** Elimina l'expressió booleana del conjunt donat el seu nom.
    - **SetExpressioBooleana(String nom, String cos):** Substitueix una expressió booleana donats el seu nom i el nou cos. La crea si aquesta no existia previament.
- **Descripció de les relacions:**
    - Relació amb la classe ExpressioBooleana: Hi ha diverses instancies d'ExpressioBooleana totes associades a la mateixa instancia d'ExpressioBooleanaCtrl

ConsultaAutors

- **Nom de la classe:** ConsultaAutors
- **Descripció:** Representa el tipus de consulta per Autor, descrita a l'apartat 2 de l'Enunciat.
- **Cardinalitat:** 0 instàncies
- **Descripció dels mètodes**
    - **Set<Frase> donaAutors (String prefix, TST <Frase, MAP < String, Document> >)**
      Retorna un set on estan els noms dels autors que compleixen el prefix, passat per paràmetre, en forma Frase. El set resultant és el resultat d'una crida a la funció *obtenirAutors(String=prefix, int=0)* de la classe *TernaryTree*.

- **Descripció de les Relacions: -**

ConsultaTítol

- **Nom de la classe:** ConsultaTítol
- **Descripció:** Representa el tipus de consulta per Títol, descrita a l'apartat 1 de l'Enunciat.
- **Cardinalitat:** 0 instàncies
- **Descripció dels Atributs:**
- **Descripció dels mètodes:**
    - **public static SET <Document> getDocAutor(Frase autor, TST < PAIR < Frase, MAP< String, Document>>>)**
      Retorna el conjunt de documents corresponent a l'autor, donat l'autor i el Ternary Search Tree que, per a cada autor, manté un mapa dels títols.
- **Descripció de les Relacions: -**

Frase

- **Nom de la classe:** Frase
- **Descripció:** Classe que representa una frase i guarda d'aquesta les seves paraules.
- **Cardinalitat:** 0 o més instàncies.
- **Descripció dels atributs:**
    - **Paraula[] Oracio:** Conjunt d'instàncies de la classe Paraula que formen la Frase.
    - **int n_paraules:** Número de paraules de la frase.
    - **String text:** Contingut de la frase original en format String.
- **Descripció dels mètodes**
    - **Frase(ArrayList<Paraula> words, String frase)**
      Constructora d'una frase donada una ArrayList de les paraules que formen part de la frase i un string (per l'atribut text).
    - **Frase(Paraula[] words, String frase)**
      Constructora d'una frase similar a l'anterior però ara les paraules venen donades per Paraula[].
    - **Frase(String frase)**
      Constructora d'una frase donat només un string. Mitjançant les funcions String to paraula, decompose i isPuntuacio s'aconsegueixen les paraules.
    - **boolean conteSequencia(String[] Paraules)**
      Retorna *true* si es troben les paraules concatenades (és a dir una darrere de l'altre en ordre i sense interrupcions), passades per atribut, en la frase.
    - **boolean conteCaracters(String lletres)**
      Retorna *true* si la frase conté els caràcters que formen el String *lletres* en ordre i concatenats.
    - **boolean conteParaula(String paraula)**
      Retorna *true* si la frase conté la paraula passada per paràmetre.
    - **HashMap<Integer, Integer> donaWords()**
      Retorna un HashMap on, per cada paraula diferent de la Frase, la *key* és el *Id* de la paraula i el *value* és el número d'aparicions d'aquesta en la frase.
    - **boolean isPuntuacio(char c)**
      Retorna *true* si el caràcter c és igual a una sèrie de caràcters que ens interesa identificar (els quals en aparèixer marquen el final d'una paraula).
    - **ArrayList<String> decompose(String frase)**
      Retorna, donat una Frase en format String, un ArrayList de les paraules d'aquesta frase. Es va creant un String llegint els caràcters que són lletres de *frase* que serà una paraula a insertar; quan arribem

a un signe de puntuació (isPuntuacio) sabem que la paraula ha acabat i podem inserir el String a l'ArrayList.
    - **ArrayList<Paraula> stringToParaules(String frase)**
      Donat un String obté primer un ArrayList<String> de les paraules gràcies a decompose i després retorna un ArrayList<Paraula> on per cada string de l'ArrayList retornat per decompose es crea una instància de la classe Paraula i s'afegeix al ArrayList resultat.

- **Descripció de les relacions:**
    - Relació d'associació amb la classe "Paraula": indica les paraules per les quals la frase està formada.
    - Una instància de la classe Frase pot ser títol o autor de diversos documents.
    - Un contingut està format de diverses frases.

Document

- **Nom de la classe:** Document
- **Descripció:** Document donat d'alta.
- **Cardinalitat:** 0 o més instàncies.
- **Descripció dels atributs:**
    - **LocalDate date:** Indica la data de creació del document.
    - **boolean isFav:** Indica si el document ha estat marcat com a preferit
    - **Frase title:** Instància de la classe Frase que fa referència al títol del document.
    - **Frase author:** Instància de la classe Frase que fa referència a l'autor del document.
    - **Contingut cont:** Instància de la classe Contingut que té tot el text del contingut del document. Si no es necessita el document, llavors aquest camp serà null (serà importat de DATA/)
- **Descripció dels Mètodes:**
    - **Document (autor, titol, isFav, path, date, cont)**
      Constructora d'un Document que inicialitza els atributs.
    - **double getTFofWord(int index)**
      Donat l'índex d'una paraula, retorna el Term Frequency d'aquesta al contingut del document (o -1 si no existeix la paraula al contingut).
    - **MAP<Integer, Double> getTF ()**
      Retorna un HashMap que mapeja cada índex de paraula amb el seu TF al contingut.
    - **void oblidaContingut ( )**
      Oblida el contingut i fa un set del camp *cont* a null
    - **boolean conteSequencia (String seq)**

Retorna *true* si, i només si, la sequüència *seq* es troba en el text del contingut, en la String de l'Autor, o en el títol.

- **Descripció de les relacions:**
    - Document té un únic autor i títol; tots dos són instàncies de la classe Frase.
    - Document té un Contingut

ConsultaData

- **Nom de la classe:** ConsultaData
- **Descripció:** Representa el tipus de consulta per Data, una cerca extra basada en la consulta de Documents segons la seva data (atribut de la classe Document).
- **Cardinalitat:** 0 instàncies
- **Descripció dels atributs: -**
- **Descripció dels Mètodes:**
    - **public static LIST < Document> consulta (LIST < Docuement> docs, DATE anterior, DATE posterior)**
      Retorna els documents amb data <= posterior i amb data >= anterior. Es fa mitjançant dues cerques lineals, una començant pel final per acotar superiorment i una començant per l'inici per acotar inferiorment *docs*, on s'aconsegueixen dos iteradors que marquen els indexos on l'interval de documents requerit comença i acaba. Es fa un recorregut lineal i s'insereix cada document desde l'index inicial fins al final al ArrayList resultat i es retorna. En el cas de ser un interval sense documents es retorna un ArrayList empty.

- **Descripció de les relacions:-**

ConsultaPreferit

- **Nom de la classe:** ConsultaPreferit
- **Descripció:** Consulta documents marcats com a preferits.
- **Cardinalitat:** 0 instàncies.
- **Descripció dels atributs:**
- **Descripció dels mètodes:**
    - **SET <Document> getDocPreferit (SET < Document> docs)**
      Retorna el conjunt de documents preferits.
- **Descripció de les relacions:** -

ConsultaRellevancia

- **Nom de la classe:** ConsultaRellevancia
- **Descripció:** Representa el tipus de consulta per Títol, descrita a l'apartat 4.3 de l'Enunciat.
- **Cardinalitat:** 0 instàncies
- **Descripció dels atributs: -**
- **Descripció dels Mètodes:**

  public static Llibreria ConsultaPerRellevancia(int k_docs, Paraula[] words, String frase, int mode, Llibreria documents)
  Mètode estàtic on es retorna una Consulta per Rellevància amb *k*= k_docs, *query*= words; docs s'obtè depenent del mètode (seleccionat per *mode*) que volguem utilitzar (recordem que aquesta consulta ofereix dues formes d'obtenir els k documents més rellevants a la query). El primer mètode suma les aparicions de les paraules de la query en cada document i guarda, en docs, els k documents amb la suma més gran (el primer el que té la suma màxima). El segón mètode es basa en crear un document trivial, on títol, autor i contingut és igual a la query (pasada per paràmetre i en format String en *frase*), i fer una cerca per Semblança utilitzant aquest document com el de referència, amb *k*= k_docs i Llibreria com a conjunt de documents a partir dels quals volem obtenir els k més rellevants.

Llibreria

- **Nom de la classe:** Llibreria
- **Descripció:** La llibreria representa un conjunt de documents.
- **Cardinalitat:** 1 principal (més si es retorna una Llibreria en alguna consulta)
- **Descripció dels atributs:**
  - **LIST < PAIR < Document, MAP < Integer, Double > > > docs0**
    Llista de documents, on cada document té un HashMap que serveix com a representació del vector sparsed de paraules amb pesos TF-IDF
  - **LIST < PAIR < Document, MAP < Integer, Double > > > docs1**
    Llista de documents, on cada document té un HashMap que serveix com a representació del vector sparsed de paraules amb pesos el nombre d'ocurrències de cada paraula al document.
  - **MAP < Integer, Integer > word_ocurrences**
    HashMap que assigna cada índex de paraula amb el seu nombre d'ocurrències totals a tota la llibreria. Necessari pel càlcul de l'IDF eficient.
  - **HashMap<Document, HashMap<Document, Double>> precalculat0**: Matriu de TF-IDF precalculat.
  - **HashMap<Document, HashMap<Document, Double>> precalculat1**: Matriu d'ocurrències precalculat.

    - **HashMap<Document, Integer> docMapper:** Mapa que ens mapeja cada document amb el seu índex als vectors tfidf.
    - **int nDocs:** Nombre de documents en total a la llibreria
    - **TernaryTree<Pair<Frase, HashMap<String, Document>>> autor_documents:** Arbre d'autors i de documents. Cada autor té el seu map de documents, amb el títol com a clau.
    - **ArrayList<Document> documents_per_data:** Llistat de documents ordenats per data.
- **Descripció dels Mètodes:**
    - **Llibreria ()**
      Constructora d'una llibreria, inicialment buida.
    - **double computeCosinus(Document d1, Document d1, int mode)**
      Donats dos documents, calcula el cosinus entre els dos vectors de paraules que representen cada document. Els pesos de les paraules són o bé TF-IDF (si *mode = 0*) o el nombre d'ocurrències (mode = 1).
    - **void addDocument(Document d)**
      Afegeix el document *d* a la llibreria i actualitza tots els índex IDF i ocurrències de les paraules.
    - **void afegir_document_ordenat(Document D)**: Afageix el document a la posició correcta de documents_per_data.
    - **void deleteDocument(Document d)**
      Elimina el document *d* de la llibreria i actualitza tots els índexs IDF i ocurrències de les paraules.
    - **PAIR < Document, Boolean > getDocument (String aut, String tit)**
      Mètode que retorna el Document identificat per l'String *aut* i *tit*. Si existeix el booleà del Pair que retorna serà *true*, altrament *fals*. En cas que sigui *fals* el document retornar serà *null*.
    - **Llibreria getPreferits()**: Retorna una Llibreria dels documents preferits
    - **Document getlessim(int i):**
      Retorna l'i-éssim document afegit a la Llibreria.
    - **SET < Document > getSetDocuments ( )**
      Retorna tots els documents de la llibreria en un conjunt.
    - **ArrayList<String> toStringArray()** Mètode que retorna la Llibreria representada con una arrayList de Documents (en String)
    - **String toString()** Retorna el document com una seqüència de paraules.
    - **TernaryTree<Pair<Frase, HashMap<String, Document>>> getArbre():** Retorna l'arbre dels autors.
    - **ArrayList<Document> getDocArray()**: Retorna el llistat de documents ordenats per dates.

- **Descripció de les relacions:**
    - Llibreria té referències als Documents que conté

ConsultaAvançada

- **Nom de la classe:** ConsultaAvancada
- **Descripció:** Consulta documents que conté la seqüència indicada.
- **Cardinalitat:** 0 instàncies.
- **Descripció dels atributs:**
- **Descripció dels mètodes:**
    - **public static SET<Document> obtenirDocuments(Llibreria l, String s)** Retorna el conjunt de documents dins de l que contenen la seqüència s.
- **Descripció de les relacions:** -

TernaryTree

- **Nom de la classe:** TernaryTree
- **Descripció:** Estructura de dades per contenir el diccionari de paraules.
- **Cardinalitat:** 1
- **Descripció dels atributs:**
    - **esquerra:** Node fill de l'esquerra.
    - **dreta:** Node fill de la dreta.
    - **centre:** Node fill del centre.
    - **paraula:** Paraula que el node fa referència.
    - **lletra:** Lletra que conté el node.
- **Descripció dels mètodes:**
    - **TernaryTree()** Constructora per defecte de TernaryTree.
    - **TernaryTree(char c)** Constructora per als nodes fills.
    - **esborrar(String s, int i, TernaryTree<T> esborrable, int dir)** Esborra el valor corresponent a s juntament amb els nodes innecessaris.
    - **T inserirObtenirParaula(String s, int i)** Obtenir el contingut que correspon a la seqüència s, el crea si no existeix aquest.
    - **T obtenir(String s, int i)** Funció que retorna el contingut que correspon a la seqüència s.
    - **void recorreArbre(Set<T> resultat)** Obté tots els valors de l'arbre.
    - **void obtenirValors(Set<T> resultat, String s, int i)** Obté tots els valors que contenen el prefix s.
    - **Set<T> obtenirPerPrefix(String s)** Funció que retorna tots els valors de l'abre que contenen la seqüència s com a prefix.
- **Descripció de les relacions:** -

Pair<L,R>

- **Nom de la classe:** Pair
- **Descripció:** Classe que implementa un Pair.
- **Cardinalitat:** 0 o més instàncies.
- **Descripció dels atributs:**

    - **L l:** Contingut de l'esquerra**.**
    - **R r:** Contingut de la dreta.
- **Descripció dels mètodes:**
    - **L getL()** retorna el contingut de l'esquerra.
    - **R getR()** retorna el contingut de la dreta.
    - **void setL()** assignar el contingut de l'esquerra.
    - **void setR()** assignar el contingut de la dreta.
- **Descripció de les relacions:** -

Node

- **Nom de la classe:** Node
- **Descripció:** Classe que implementa un node de l'expressió booleana.
- **Cardinalitat:** 0 o més instàncies.
- **Descripció dels atributs:**
    - **String data:** Contingut del node.
    - **Node left:** Node de l'esquerra**.**
    - **Node right:** Node de la dreta.
- **Descripció dels mètodes:**
    - **Node(String date)** Crea un node assignant el contingut.
- **Descripció de les relacions:** -