PROJECTE DE PROGRAMACIÓ

CURS 19-20

ASPECTES SOBRE DESENVOLUPAMENT I AVALUACIÓ DEL PROJECTE

TREBALL EN GRUP

- Aquesta activitat s'ha de desenvolupar obligatòriament en grups de **2 o 3 alumnes**, de manera que els alumnes d'un mateix grup **heu d'assistir al mateix horari de laboratori.**
- Els grups els trieu els alumnes, però si cal hi intervindrà el professor (en particular per garantir que cap alumne quedi sol).
- Si durant el desenvolupament del projecte algun alumne abandona, haurà de comunicar-ho al seu professor de pràctiques **el més aviat possible**. Llavors el professor reajustarà el grup de la forma que cregui més adequada (si això succeeix en un grup de 3, el més probable és que passi de forma immediata a ser considerat grup de 2)
    - *Heu de procurar no abandonar el projecte a la fase final del curs, ja que aquestes situacions perjudiquen clarament la feina dels vostres companys de grup i també el seguiment per part del professor. Si voleu deixar el projecte, feu-ho quan encara hi hagi temps per a que els vostres companys es reorganitzin!*

SEGUIMENT

- El seguiment del desenvolupament del projecte per part del professor es durà a terme a classe de laboratori. **Es requereix, per tant, l'assistència de tots els membres del grup**.
- El professor de laboratori avaluarà cada setmana el grau de seguiment de la feina del projecte a cada grup, i podrà penalitzar algun o tots els alumnes del grup cas de considerar que no estan treballant de forma correcta. **La penalització màxima setmanal per alumne serà de mig punt** (que es reflectiran a la part corresponent a desenvolupament, vegeu el següent apartat).
- Si, puntualment, algun membre del grup no pot assistir a classe, haurà de comunicar-ho al professor per correu electrònic, tot indicant la causa de l'absència. El professor determinarà llavors si es requereix alguna acció específica al respecte, com ara una trobada al despatx per comprovar el seguiment de l'alumne absent a classe. Cas d'incompliment d'aquesta norma, l'alumne pot ser penalitzat d'acord amb el que s'ha comentat al punt anterior.

LLIURAMENTS

PRIMER LLIURAMENT

DATES DE LLIURAMENT: dimecres 11 o 18 de març (grup dimecres) / **divendres 13 o 20 de març** (grups divendres).

***Atenció**: el fet de realitzar el lliurament el primer o el segon dels dies estipulats no comportarà cap penalització pel que fa a la nota. Tanmateix, els grups que lliureu el primer dia rebreu abans la correcció i el feedback del vostre professor, i per tant estareu en condicions d'avançar feina més aviat.*

Es lliurarà al **professor**, durant la classe de pràctiques, un document en format paper (un document per grup), al qual hi haurà:

- **Definició dels tipus abstractes de dades i mòduls funcionals** que formaran l'aplicació, incloent-hi:
  - una descripció general (abstracta) de cadascun d'ells,
  - la capçalera de les seves operacions publiques,
  - i l'especificació pre/post de cadascuna d'elles.

  Es prohibeix explícitament incloure cap detall d'implementació (representació de dades o codi) en aquesta documentació. Fer-ho implicarà una penalització a la nota (fixeu-vos que el que us demanem són tipus abstractes de dades).

- **Diagrama de mòduls**, on constin els diferents tipus abstractes de dades i mòduls funcionals, la signatura de les operacions, i les seves relacions de **d'ús (- - >)** i **subtipatge (—|>)**. Noteu que no demanem relacions d'implementació, perquè tots els tipus en aquest primer lliurament han de ser abstractes.

  Nota: Un mòdul A usa un altre mòdul B (A - - > B) sempre que apareix un element de tipus B com a paràmetre o com a resultat d'alguna operació d'A. Les relacions d'agregació i composició d'UML en són casos particulars, que podeu utilitzar si les coneixeu i ho desitgeu.

En trobareu un **exemple** de tot plegat a l'annex d'aquest mateix document.

LLIURAMENT FINAL

DATA LÍMIT: dimarts 19 de maig

Haureu de lliurar (al **Moodle**):

- Tots els fitxers font (.java) de l'aplicació.
- El joc de proves utilitzat
- Els documents descrits més endavant
- Un fitxer .jar corresponent a l'aplicació que permeti executar-la amb facilitat.

AVALUACIÓ I QUALIFICACIÓ: PESOS

Hi haurà una nota del projecte (entre 0 i 10) per a cada alumne, que es calcularà de la següent forma:

A) **DESENVOLUPAMENT (3 punts), dels quals:**

   A1) Primer lliurament (1 punt)

   A2) Procés de desenvolupament (2 punts)

B) **LLIURAMENT FINAL (7 punts)**

Noteu que la nota de l'apartat de desenvolupament pot ser diferent per a cada membre d'un grup. També pot ser diferent la nota del resultat final, depenent dels mòduls desenvolupats per cadascú.

AVALUACIÓ I QUALIFICACIÓ: ASPECTES QUE ES VALORARAN

DEL PRIMER LLIURAMENT

Es valorarà l'adequació de:

- els mòduls triats
- la descripció dels mòduls
- les relacions entre els mòduls
- les operacions de cada mòdul
- l'especificació pre/post de les operacions
- el diagrama de mòduls

DEL PROCÉS DE DESENVOLUPAMENT

Es valoraran els següents aspectes:

- Assistència a les sessions de seguiment
- Interacció amb el professor (preferentment a classe, al despatx si és necessari)
- Actitud personal
- Planificació i compliment de la feina programada cada setmana
- Treball en grup. Aquí ens referim a temes com ara la divisió de feina entre els membres del grup (cada membre s'haurà d'ocupar d'un grup de mòduls, i el professor haurà de ser informat al respecte), la comunicació entre els membres del grup, les reunions setmanals (a classe i fora de classe), la dinàmica de treball i l'actitud de cada alumne dins el grup.

DEL LLIURAMENT FINAL

Es valoraran els següents punts:

FUNCIONAMENT

El funcionament és, òbviament, un capítol important que inclou diferents aspectes, com ara:

- Adequació als requeriments inicials
- Absència de bugs
- Interfície d'usuari "amigable", tant en mode text com en mode gràfic
- Tractament de situacions anòmales

DISSENY MODULAR

Un dels aspectes clau del desenvolupament d'un projecte de programació d'aquestes característiques és el disseny modular, és a dir, el fet de decidir quins mòduls hi haurà al nostre disseny (i quines operacions públiques hi haurà a cada mòdul) i també les relacions entre aquests mòduls (subtipatge, ús). Realitzar una bona divisió modular és fonamental, i presenta molts avantatges, com ara la claredat i llegibilitat, la facilitat per detectar i arreglar errors, la possibilitat de reutilitzar algun mòdul, i la facilitat en modificar /ampliar funcionalitats de l'aplicació

Remarquem que els mòduls corresponen a les classes (instanciables o abstractes), interfícies i mòduls funcionals.

Heu de tenir en compte, per tal de realitzar un bon disseny, alguns principis fonamentals:

- Heu de triar els mòduls de forma adient, de manera que es faciliti la seva reutilització. Cada mòdul ha de tenir una funcionalitat concreta i clara. Recordeu que un mòdul ha de ser responsabilitat única d'un membre del grup.
- En particular, l'entrada/sortida de l'aplicació ha de ser independent del nucli de l'aplicació, és a dir, cal concentrar l'entrada/sortida en mòduls dedicats a aquesta funcionalitat.
- Cal també triar de forma adient quins mòduls estaran relacionats entre ells, i també quin tipus de relació s'hi estableix (subtipatge, ús), tot evitant relacions innecessàries, és a dir, procurant maximitzar la independència entre mòduls.
- Un altre punt crític és decidir quines operacions públiques s'inclouen a cada mòdul. Cal pensar en no incloure com a públiques operacions que puguin ser privades.
- Un mòdul mai accedirà a la part privada d'un altre mòdul (encapsulament).
- Cal evitar tant com sigui possible els mòduls excessivament grans.
- Cal evitar tant com sigui possible les repeticions de codi (principi "*don't write twice*").
- L'ús excessiu de *càstings* i/o de l'operador *instanceof* solen ser símptomes d'un disseny poc acurat.

D'altra banda, cal tenir clar que un bon disseny modular rarament s'aconsegueix "a la primera". Cal anar rectificant decisions preses anteriorment. De tota manera, quant millor ens pensem el disseny menys haurem de "tornar enrere" més endavant, i per tant ens estalviarem feina.

ESTRUCTURES DE DADES I ALGORÍSMICA

Dins aquest apartat, valorarem:

- Algorismes utilitzats
- Tria de les estructures de dades adients per a cada classe, fent un ús adequat del que ens ofereix l'API del Java. Heu de triar estructures que afavoreixin solucions eficients.
- Eficiència asimptòtica.

CLAREDAT/LLEGIBILITAT DEL CODI

Un altre tema que cal tenir en compte és el fet que el codi sigui el més llegible possible. Cal que eviteu, doncs:

- Mètodes gaire llargs
- Excés de paràmetres
- Noms identificadors (de mètodes, de mòduls, etc.) poc descriptius
- Codi mal alineat
- Ús d'estructures de codi innecessàriament complicades
- Codi repetitiu
- Variables globals
- ...

DOCUMENTACIÓ + JOC DE PROVES

Heu d'oferir una documentació acurada de l'aplicació resultant. Aquesta documentació ha d'incloure, necessàriament, els següents documents, en format pdf (excepte la documentació generada per Doxygen, que estarà en format html):

DOCUMENT 1: Disseny modular

- **Diagrama dels mòduls utilitzats i les seves relacions** (el podeu elaborar amb l'eina que preferiu).
- **Documentació html, elaborada amb Doxygen**, que inclogui, per a cada mòdul de l'aplicació:
  - Nom i descripció general del mòdul.
  - Descripció dels atributs.
  - Especificació pre/post de cada operació (pública i privada).

NOTA: *per realitzar aquesta documentació html us heu de basar a l'exemple que es comenta a classe de pràctiques.*

DOCUMENT 2: **El joc de proves**

És fonamental que tingueu present que un bon desenvolupament d'un projecte no pot obviar la fase de proves. A més de les proves realitzades al final, quan ja doneu l'aplicació per acabada, convé també que proveu els mòduls (si més no els més importants) de forma individual, per a la qual cosa no cal tenir acabada l'aplicació. D'aquesta forma serà més fàcil la detecció i correcció d'errors.

Haureu de lliurar un document il·lustrant el joc de proves, incloent-hi una descripció de totes les proves realitzades (fitxers d'entrada, resultats obtinguts, ...). Estaria bé que hi féssiu sortir també alguna captura de pantalla.

DOCUMENT 3: Petit manual d'usuari

CONSIDERACIONS FINALS

- Us recordem que aquesta assignatura té 5 crèdits ECTS, dels quals 3 corresponen a treball no presencial. Això correspon, aproximadament, **a unes 5h per setmana de dedicació al projecte** (durant les aproximadament 10 setmanes que aquest projecte comprèn).

- En aquest sentit és important que cada grup vagi marcant un **ritme de treball adient setmana rere setmana, amb la supervisió del professor**. Això implica marcar-se uns objectius (almenys a una setmana vista) i, naturalment, procurar d'acomplir-los. Com ja s'ha comentat, aquest aspecte tindrà repercussió a la nota del projecte (a la part corresponent al desenvolupament).

- Podeu usar les eines que cregueu més adequades a l'hora de realitzar les implementacions (IDE, gestors de versions, eines per documentar/dissenyar diagrames, etc.). Pel que fa al Java, recordeu que el codi ha de ser compatible amb la versió 8. Podeu utilitzar els recursos que ofereix l'API de Java, però en principi **no podeu usar recursos externs, de fora de l'API** (tret que el professor indiqui el contrari).

ANNEX: EXEMPLE DE DOCUMENTACIÓ PER AL PRIMER LLIURAMENT

Tipus Laberint

Descripcio general: Laberint format per sales, cadascuna d'elles amb una porta d'entrada i dues portes de sortida.

Operacions

```
Laberint()
Pre:  ---
Post: Crea un laberint buit.

Laberint(Sala entrada, Laberint esquerre, Laberint dret)
Pre:  entrada != null
Post: Crea un laberint a partir d'una sala i dos laberints.
      La sala  entrada  s'estableix com a sala d'entrada al laberint.
      Les portes esquerra i dreta de sortida d'aquesta sala es
      connecten amb el laberint  esquerre  i  dret  respectivament.

Sala salaEntrada()
Pre:  ---
Post: Retorna la sala d'entrada d'aquest laberint.

Laberint laberintEsquerre()
Pre:  ---
Post: Retorna el laberint que hi ha a partir de la porta de sortida
      esquerra de la sala d'entrada d'aquest laberint.

Laberint laberintDret()
Pre:  ---
Post: Retorna el laberint que hi ha a partir de la porta de sortida
      dreta de la sala d'entrada d'aquest laberint.

boolean buit()
Pre:  ---
Post: Diu si aquest laberint es buit (sense cap sala)
```

Tipus Sala

Descripcio general: Una habitacio amb objectes

Operacions

```
Sala(int num, boolean oberta, Caixa c, boolean tresor)
Pre:  ---
Post: Crea una sala amb numero identificador  num, i caixa
      amb objectes  c. Els booleans  oberta  i  tresor  indiquen
      si la porta d'entrada esta oberta, i si la sala conte un
      tresor, respectivament.

int numero()
Pre:  ---
Post: Retorna el numero de sala.
```

boolean oberta()
Pre: ---
Post: Diu si la porta d'entrada esta oberta.

Caixa caixa()
Pre: ---
Post: Retorna la caixa dins la sala.

boolean tresor()
Pre: ---
Post: Diu si la sala conte un tresor.

Tipus Persona

Descripcio general: Persona que visita un laberint, gastant una unitat de vida a cada moviment (canvi de sala).

Operacions

Ruta explorar(Laberint l)
Pre: ---
Post: Explora el laberint l. Si troba un tresor, aleshores retorna la ruta seguida fins arribar a la sala del tresor.
En cas contrari retorna una ruta buida.

Tipus IndianaJones refina Persona

Descripcio general: Personatge de ficcio, que explora laberints seguint una estrategia optima consistent en agafar els objectes que troba pel cami que li poden aportar major benefici, i no repetir mai camins ja intentats.

Tipus Ruta

Descripcio general: Sequencia de sales

Operacions

Ruta()
Pre: ---
Post: Crea una ruta buida.

void afegirSala(Sala s)
Pre: ---
Post: Afegeix la sala s al final de la ruta.

Modul funcional Dibuix

```
void dibuixar(Laberint l)
Pre:  ---
Post: Dibuixa el laberint  l  en una finestra.
```

DIAGRAMA DE MÒDULS CORRESPONENT