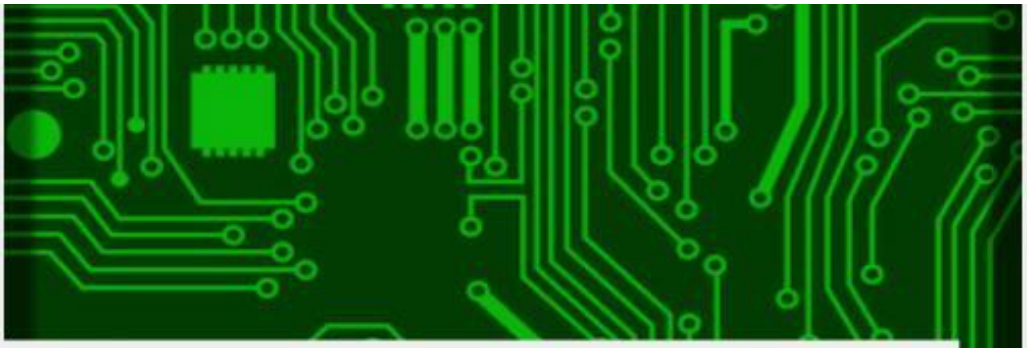
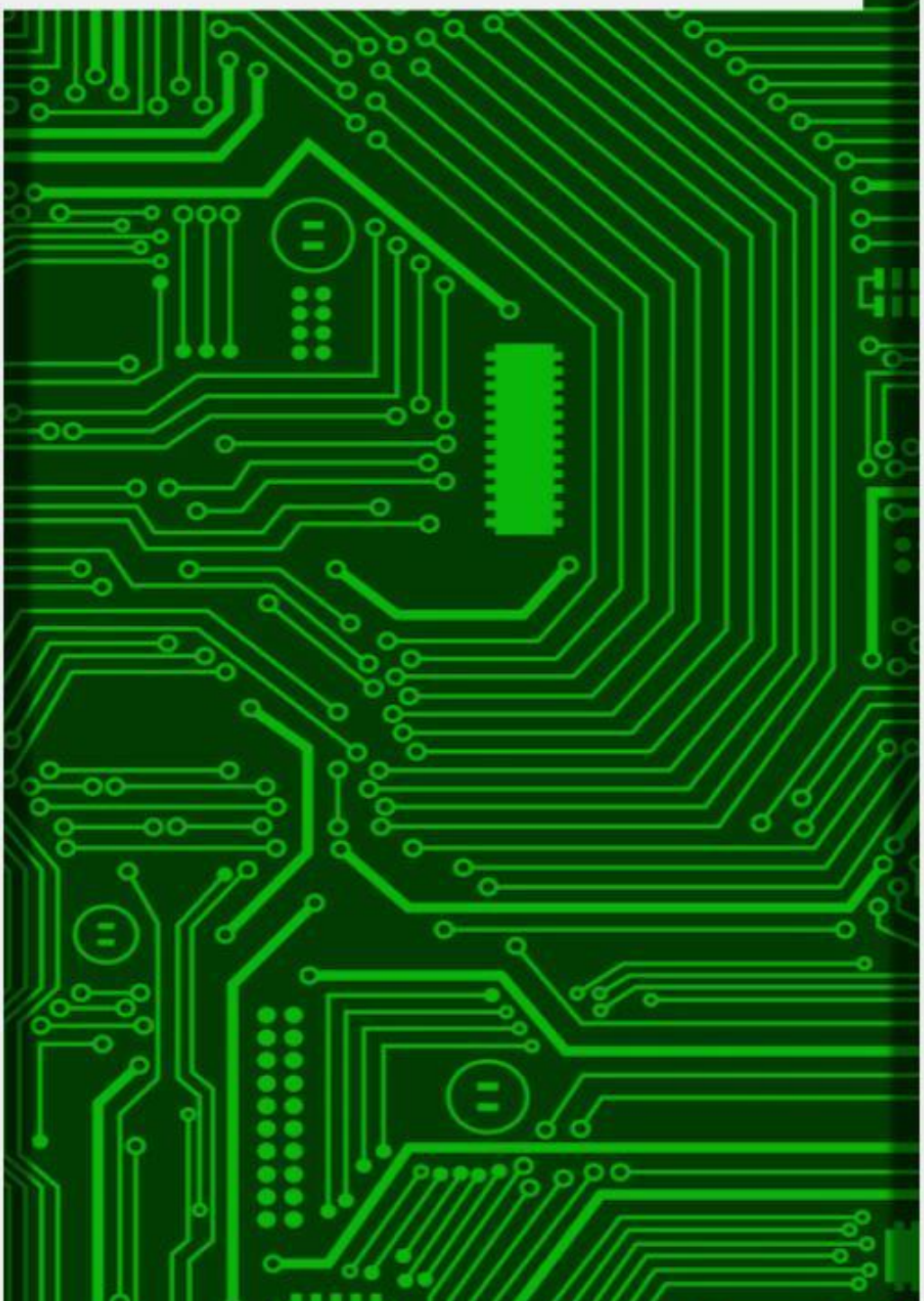


DAM 1

Exercici 52_02. Validació de programes

Soufian El Hajouji



ÍNDEX

Pregunta 1. Els meus errors

Aquesta pregunta fa referència als continguts Els errors.

Si has arribat a aquestes alçades del curs, probablement ja fa un cert temps que desenvolupes i segur que en tot aquest temps has comés errors. Probablement molts. No cal que t'avergonyeixis, ens passa a tothom!

En aquesta pregunta, et demano que pensis en un error de programació concret que t'hagi passat. Intenta trobar un que et costés prou de trobar o/i de corregir. Descriu en detall:

l'error: en què consistia

la detecció: com te'n vas adonar

el procés de trobar-lo. ex. com vas realitzar les proves

la correcció: en què va consistir la correcció

ah! intenta incloure com et vas sentir a cadascuna de les quatre fases anteriors! (encara que tothom pot imaginar-se la darrera si va costar-te molt, oi?)

L'error consistia en utilitzar l'operador == per comparar dos Strings en lloc de fer servir el mètode equals(), el que provocava que la comparació no funcionés correctament.

Vaig detectar l'error mentre revisava el codi i vaig veure que el mateix valor de String es comparava amb el mateix valor de String utilitzant == en lloc de equals().

Per a trobar l'error, vaig revisar el codi línia per línia per identificar on es produïa la error.

Per corregir l'error, vaig canviar totes les comparacions que feien servir l'operador == per comparar Strings a través del mètode equals(). Això va solucionar el problema i les comparacions van funcionar correctament.

Durant el procés de detecció, vaig sentir una mica de frustració pel temps que em va portar descobrir l'error, però també em vaig sentir satisfet en detectar i resoldre el problema. Així mateix, durant el procés de correcció, em vaig sentir satisfet i satisfet de solucionar l'error i millorar el codi.

Pregunta 2. El cicle de TDD

Aquesta pregunta fa referència als continguts Les proves.

Descriu amb les teves paraules les cinc passes que conformen el cicle de Test Driven Development segons aquest article de la Wikipedia sobre el TDD.

Compara aquest cicle amb la teva manera de programar actual. Quina et sembla més adequada segons criteris com velocitat de desenvolupament, qualitat de codi generat, mantenibilitat, etc.

El cicle de Test Driven Development (TDD) es basa en cinc passes que s'han de repetir per a cada nova funcionalitat. Segons l'article de la Wikipedia, aquestes passes són les següents:

1. **Afegir una prova:** El desenvolupador comença per escriure una prova que passa només si es compleixen les especificacions de la nova funcionalitat. Això ajuda a definir les especificacions abans d'escriure el codi i centra l'atenció en les necessitats dels usuaris.
2. **Executar totes les proves:** La nova prova hauria de fallar per les raons esperades, demostrant que es necessita codi nou per a la funcionalitat desitjada i validant que el test funciona correctament.
3. **Escriure el codi més simple que compleix la prova:** Escriure codi inelegant o difícil d'entendre és acceptable, sempre i quan compleixi la prova. El codi es millorarà posteriorment.
4. **Fer passar totes les proves:** Si alguna prova falla, el nou codi s'ha de revisar fins que totes les proves passin. Això assegura que el nou codi compleix les especificacions i no trencarà les funcionalitats existents.
5. **Refactoritzar segons sigui necessari, fent servir les proves després de cada refactorització per assegurar que la funcionalitat es conserva:** El codi es refactoritza per millorar la seva llegibilitat i mantenibilitat. Les dades de prova codificades en dur s'han de suprimir. Executar les proves després de cada refactorització ajuda a assegurar que cap funcionalitat existent no s'hagi trencat.

Comparant això amb la meva manera de programar actual, trobo que el cicle TDD és més estructurat i rigorós. Em sembla que pot ser més efectiu per a projectes importants o complexes, ja que ajuda a evitar errors i manté la qualitat del codi. Tanmateix, per a petits projectes o projectes de prova, la meva manera de programar actual és més ràpida i menys laboriosa. En general, considero que la combinació de la meva experiència i el cicle TDD poden portar a un codi de qualitat i una millor eficiència de desenvolupament.

Pregunta 3. Qui fa què

Aquesta pregunta fa referència als continguts Classes de proves.
Per cadascun dels nivells de prova, indica quin és el perfil dels testejadors més adequats per realitzar-les. Justifica-ho.
Proposa qui hauria de dissenyar aquestes proves i quan ho hauria de fer (a quina fase del desenvolupament).

1. Proves de sistema: aquestes proves han de ser realitzades per un equip de testejadors que puguin veure el sistema des del punt de vista d'un usuari final. Això significa que els testejadors han de tenir un coneixement tècnic i també una comprensió profunda dels requisits del sistema i dels casos d'ús dels usuaris finals. Els testejadors han de ser capaços de dissenyar i executar les proves de sistema per verificar que el sistema compleix els requisits especificats.

2. Proves d'integració: en aquest cas, els testejadors han de tenir un coneixement tècnic profund del sistema en qüestió, ja que les proves d'integració són realitzades sobre diferents components que han de treballar junts. Els testejadors han de ser capaços de coordinar-se amb els desenvolupadors per aconseguir una integració correcta dels components i també han de ser capaços de detectar i solucionar errors que apareguin en aquesta fase.

3. Proves funcionals: aquestes proves han de ser realitzades per un equip de testejadors que puguin veure el sistema des del punt de vista d'un usuari final. Això significa que els testejadors han de tenir un coneixement tècnic i també una comprensió profunda dels requisits del sistema i dels casos d'ús dels usuaris finals. Les proves funcionals han de verificar que el sistema compleix les especificacions funcionals del sistema. Els testejadors han de ser capaços de dissenyar i executar les proves funcionals per validar que el sistema compleix els requisits especificats.

4. Proves unitàries: aquestes proves són realitzades pels mateixos desenvolupadors de software que han escrit el codi. Això és perquè les proves unitàries són petites proves que es realitzen sobre les funcions o mètodes individuals i, per tant, requereixen un coneixement profund del codi i la programació. Els desenvolupadors han de ser capaços de dissenyar i executar les proves, i corregir els errors que trobin.

Pel que fa al disseny d'aquestes proves, els desenvolupadors de software han de dissenyar i executar les proves unitàries. Els testejadors de qualitat o els desenvolupadors han de dissenyar i executar les proves d'integració, les proves funcionals i les proves de sistema. Finalment, l'usuari final o el client