

EXAMEN 2^a AVALUACIÓ - PRÀCTICA

MODUL - PROGRAMACIÓ

CURS: 2023/2024

Autors: Nombre professors: Guillermo Garrido Portes

Salvador Rue Orquin

Correus electrònics : g.garridoportes@edu.gva.es

s.rueorquin@edu.gva.es

Llicència



CC BY-NC-SA 3.0 ES Reconeixement – No Comercial – Compartir Igual (by-nc-sa)

No es permet un ús comercial de l'obra original ni de les possibles obres derivades, la distribució de les quals s'ha de fer amb una llicència igual a la que regula l'obra original. Aquesta és una obra de Guillermo Garrido Portes i Salvador Rue.

1. INTRODUCCIÓ

L'examen consisteix en una ampliació de l'activitat avaluable "El poble dormit". Tant si l'heu realitzada com si no, podreu completar l'examen sense cap problema, ja que vos passem la llibreria amb les classes corresponents implementades i només heu de completar la classe principal implementant les classes externes que es demanen.

Per fer-ho s'han de seguir les instruccions exactament.

2. NOMENCLATURA

- Carrega el projecte **Examen65** que vos facilitem a Aules. No oblidis canviar el nom de l'autor a la classe principal.
- Les funcions, paquets i classes que es creen han de seguir la nomenclatura que s'indica en l'exercici.
- Les variacions en nomenclatura i estructura es penalitzaran.
- L'arxiu final s'ha d'entregar en la tasca d'aules que indique el professor.

3. PROGRAMA

- El programa ha de complir els següents requisits:
- Ha d'incorporar la llibreria `mysql-connector-j` que vos facilitem a Aules.
- Ha d'incorporar la llibreria `CiudadansEpd` que vos facilitem a Aules.
- Ha d'incorporar 4 classes noves:
 - **VanHelsing** és un nou tipus d'humà immune a qualsevol altre ésser.
 - **BdadesDelPoble** classe d'utilitat per gestionar connexions i bases de dades.
 - **ExcepcionsDelPoble** classe d'utilitat on gestionen excepcions pròpies del poble.
 - **FixtersDelPoble** classe d'utilitat per gestionar fixters de text.

Pregunta 1: Classe `ElPobleDormit` (25 punts)

- La funció **`main()`** està preparada per a incorporar les noves funcionalitats:
 - **Guardar Partida:** Aquesta nova opció al menú permet guardar la població actual. Quan es guarda la partida, es creen les taules "llop", "huma" i "vampir". Cal destacar que el personatge Van Helsing, com que és un humà, es guarda a la taula "huma".
 - **Carregar Partida:** Aquesta nova opció del menú permet carregar una població des de la base de dades. El programa extreu els éssers guardats a la base de dades per a seguir jugant amb aquesta població.
 - **Escriure fitxer:** Després de passar cada any, el programa guarda automàticament la població total en un fitxer.
- La funció **`combatre`** ha de comprovar si un vampir convertirà l'últim humà del poble. Si això passa, es llançarà una excepció pròpia (que mostrarà l'ocorregut) i es crearà a VanHelsing en lloc de convertir-se en vampir. Cal destacar que només hi haurà un VanHelsing.
- La funció **`verificarPoblació`**: Aquesta funció ha de tenir en compte a VanHelsing.
- **Excepcions**, s'ha de respectar el tractament de les excepcions amb les noves excepcions que s'incorporen al `main` i les classes.



Pregunta 2: Classe VanHelsing (15 punts)

Es tracta d'un nou tipus d'humà, no pot haver cap subtipus de VanHelsing.

S'ha d'implementar totalVanHelsing, tot hi que només n'hi ha un, la seua població és comptabilitza separada als humans.

Es crea amb el nom "VAN_HELSING", la seua vida és de 100 anys i és vulnerable a Ningú. Ja disposeu del constructor humà amb el paràmetre nom que li dona automàticament els valors de la vida i vulnerable.

Al combat sempre guanya però no lluitarà amb altres humans, tampoc es reproduïx ni mor, ni de combat ni de vell.

Pregunta 3: Classe BdadesDelPoble (30 punts)

Crea la funció **getConnexio()**, que s'encarregarà de connectar amb la base de dades "poble" i tornar la connexió. Prèviament la pots crear a la pestanya "Services" com hem vist a classe.

Crea la funció **existeixTaula()**, agafa com a paràmetre d'entrada el nom de la taula a comprovar i torna un boolean a true si la taula ja existeix.

La SQL necessària per comprovar-ho ha de ser "SELECT COUNT(*) FROM information_schema.tables WHERE table_name = NOM_DE_LA_Taula" on NOM_DE_LA_Taula s'ha de substituir pel paràmetre d'entrada.

Crea la funció **eliminarTaula()**, agafa com a paràmetre d'entrada el nom d'una taula i la elimina. La SQL necessària per eliminar la taula ha de ser "DROP TABLE NOM_DE_LA_Taula" on NOM_DE_LA_Taula s'ha de substituir pel paràmetre d'entrada.

Crea la funció **crearTaula()**, agafa com a paràmetre d'entrada el nom d'una taula i crea la taula deixant ésser amb els paràmetres id+nomTaula, nom, vida i vulnerable.

La SQL necessària per crear la taula ha de ser "CREATE TABLE IF NOT EXISTS NOM_DE_LA_Taula + "(idNOM_DE_LA_Taula INT NOT NULL AUTO_INCREMENT, " + " nom VARCHAR(15) NOT NULL, " + " vida INT, " + " vulnerable VARCHAR(10), " + " PRIMARY KEY(idNOM_DE_LA_Taula));" on NOM_DE_LA_Taula s'ha de substituir pel paràmetre d'entrada.

Crea la funció **guardarDades()**, que ha d'eliminar totes les taules i crear-les de nou. A continuació, s'ha d'inserir cada ciutadà a la taula corresponent.

PD: els vampirs han de guardar-se amb 0 de vida.

La SQL necessària per crear la taula ha de ser "INSERT INTO NOM_DE_LA_Taula (nom, vida, vulnerable) VALUES (VALOR_NOM, VALOR_VIDA, VALOR_VULNERABLE)" on NOM_DE_LA_Taula s'ha de substituir pel paràmetre d'entrada, i VALOR_NOM, VALOR_VIDA, VALOR_VULNERABLE són els valors del ciutadà a guardar.

Crea la funció **carregarDades()**, borra la llista de ciutadans i fa un reset de les poblacions (teniu una funció reset a les classes de la llibreria), després crida la funció **carregarTaula()**.

Crea la funció **carregarTaula()**, agafa com a paràmetre una llista de ciutadans i el nom de la taula, afegeix a la llista els ciutadans de la taula indicada.

La SQL necessària per crear la taula ha de ser "SELECT * FROM NOM_DE_LA_TAULA" on NOM_DE_LA_TAULA s'ha de substituir pel paràmetre d'entrada.

Cal tenir en compte si hi ha alguna funció que pot produir una excepció. En aquest cas s'indicarà a la capçalera del mètode però es tractarà en el programa principal.

Menú:

1. Mostrar el cens actual
2. Passar un any
3. Guardar Partida
4. Carregar Partida
5. Eixir del programa

Selecciona una opció: 3

S'ha guardat la partida

Menú:

1. Mostrar el cens actual
2. Passar un any
3. Guardar Partida
4. Carregar Partida
5. Eixir del programa

Selecciona una opció:

S'ha recuperat la partida

Nom del ciutadà: VAN_HELSING, vida: 100, vulnerable: Ningú

Nom del ciutadà: VAMPIR2, vulnerable: Llop

Nom del ciutadà: VAMPIR4, vulnerable: Llop

Nom del ciutadà: VAMPIR5, vulnerable: Llop

Nom del ciutadà: VAMPIR6, vulnerable: Llop

Nom del ciutadà: VAMPIR7, vulnerable: Llop

Nom del ciutadà: VAMPIR8, vulnerable: Llop

Nom del ciutadà: LLOP1, vida: 5, vulnerable: Huma


L'any 0 hi ha un cens de: 8 ciutadans.

0 humans, 1 llops i 6 vampirs i Van Helsing

Menú:

1. Mostrar el cens actual
2. Passar un any
3. Guardar Partida
4. Carregar Partida
5. Eixir del programa

Selecciona una opció:

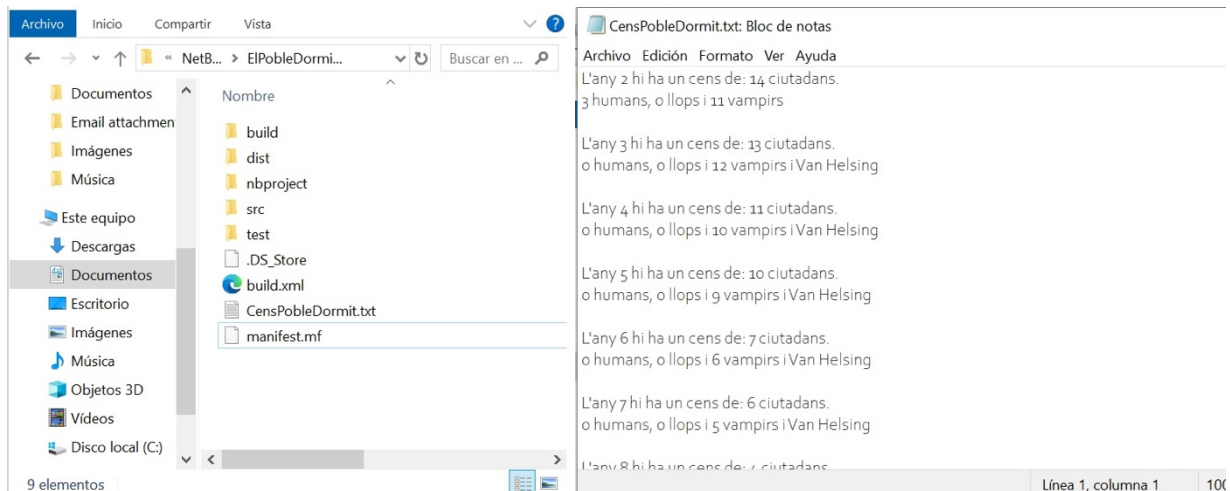
SELECT * FROM vampir LIM1... X				
 Max. rows: <input type="text" value="100"/> Fetched Rows: 8				
#	idvampir	nom	vida	vulnerable
1	1	VAMPIR1	0	llop
2	2	VAMPIR2	0	llop
3	3	VAMPIR3	0	llop
4	4	VAMPIR4	0	llop
5	5	VAMPIR5	0	llop
6	6	VAMPIR6	0	llop
7	7	VAMPIR7	0	llop
8	8	VAMPIR8	0	llop

[illegible]

Pregunta 4: Classe FitxersDelPoble (15 punts)

Conté com a paràmetre un fitxer fixe ubicat a la carpeta del projecte.

Crea la funció **escriureFitxer()**, agafa com a paràmetre una línia de text i la guarda en el fitxer. Esta funció s'ha de cridar cada any que passa per poder veure l'evolució del cens. Per simplificar no cal netejar-lo cada nova partida.



Pregunta 5: Classe ExcepcionsDelPoble (15 punts)

La classe d'excepció tracta l'excepció pròpia quan un vampir intenta convertir l'últim humà del poble, en lloc de convertir-se en vampir, aquest humà es convertirà en Van Helsing. Aquesta excepció, llançarà un missatge d'error per indicar el que ha ocorregut.

Menú:

1. Mostrar el cens actual
 2. Passar un any
 3. Guardar Partida
 4. Carregar Partida
 5. Eixir del programa
- Selecciona una opció: 2

Passant un any...

HUMA2 ataca VAMPIR4 pero mor i es converteix en VAMPIR14.

Ops! El vampir intenta convertir l'ultim humà, però este es convertirà en Van Helsing

HUMA7 ataca LLOP2 guanya i ven la seua pell per a fer abrics.

VAMPIR2 ataca HUMA6 i l'huma es converteix en VAMPIR15.

LLOP1 ataca VAMPIR9 guanya i es fa un collar amb els seus ullals.

VAMPIR15 ataca HUMA7 i l'huma es converteix en VAN_HELSING.

L'any 2 hi ha un cens de: 9 ciutadans.

0 humans, 1 llops i 7 vampirs i Van Helsing