PRÀCTICA 2

Format i data de lliurament

El lliurament s'ha de fer a l'apartat "Lliurament i registre d'AC" de l'aula de teoria, abans del dia **20 d'abril de 2020 a les 23:59**.

S'ha de lliurar un fitxer en format ZIP que contingui un directori **'UOC2019_2'** amb el codi resultant dels exercicis.

Els lliuraments han de complir els següents punts per a poder ser avaluats correctament:

- Cal lliurar un fitxer en format ZIP que contingui el directori principal 'UOC2019_2' (i que no contingui altres fitxers .zip a dins).
- El sistema i nom de fitxers del projecte lliurat han de seguir l'estructura del codi adjuntat amb l'enunciat de les pràctiques (no modifiqueu ni el nom dels fitxers, ni els de les carpetes ni en modifiqueu l'estructura de fitxers).
- El codi lliurat no ha de presentar errors de compilació per poder ser avaluat. Si una part del codi us funciona correctament però heu fet algun exercici que genera algun error, esborreu o comenteu aquesta part per tal que la resta de codi compili i es pugui avaluar la pràctica.
- Si l'entrega conté codi comentat, s'aconsella indicar mitjançant comentaris els motius d'aquesta decisió.
- El codi ha de poder-se executar. Pot ser que alguns tests fallin, però, com a mínim, el programa ha de ser capaç de mostrar el resultat de les proves.
- S'han de lliurar tots els fitxers .c i .h utilitzats en el projecte i han d'estar presents en el lliurament en les carpetes correctes (src, tests, etc.).
- S'han de lliurar els fitxers .workspace i .project que defineixen el projecte complet de CodeLite.
- Es recomana afegir dins de la carpeta principal del projecte un fitxer de nom README.txt especificant el sistema operatiu utilitzat.
- Els fitxers de test s'han de lliurar sense modificacions. Si per a alguna prova durant el desenvolupament han estat modificats, en el lliurament s'ha de tornar a la versió original.

El incompliment de qualsevol dels punts indicats pot suposar un suspens de la pràctica.

Presentació

Durant les diferents pràctiques desenvoluparem un únic projecte, del qual anireu dissenyant i implementant funcionalitats.

Treballarem seguint una metodologia basada en proves. Això significa que amb l'enunciat us facilitarem un conjunt de proves funcionals de l'aplicació. L'objectiu és implementar el codi necessari per a passar totes aquestes proves. Els exercicis que es proposen et guiaran per a facilitar la resolució de les diferents funcionalitats provades. És important destacar el fet que un codi que passi tots els test no significa que sigui un codi 100% correcte. Existeixen altres criteris d'avaluació que es detallen més endavant. A més, els test no tenen perquè provar que s'han realitzat tots els punts sol·licitats en l'enunciat, és a dir, poden no cobrir el 100% de l'enunciat.

Durant la programació és normal que sorgeixin dubtes relacionats amb el llenguatge, tant amb la sintaxi com en la forma de treballar amb les estructures de dades. Dirigiu els vostres dubtes sobre la programació al fòrum del laboratori de l'assignatura.

Objectius

Els objectius d'aquesta pràctica són:

- Treballar amb Tipus Abstractes de Dades (TAD)
- Implementar algoritmes recursius

Competències

Transversals

- Capacitat de comunicació en llengua estrangera.

Específiques

- Capacitat de dissenyar i construir aplicacions informàtiques mitjançant tècniques de desenvolupament, integració i reutilització.

Recursos

Per a realitzar aquesta pràctica disposeu dels següents recursos:

Recursos Bàsics

Laboratori de Pràctiques de Programació: Disposeu d'un laboratori assignat a l'assignatura. En aquest laboratori podeu resoldre tots els dubtes sobre la programació en llenguatge C, i la instal·lació i utilització de l'entorn de programació.

Material de Laboratori: En el laboratori, a part del col·laborador docent, disposeu de diferents documents que us ajudaran. En concret, hi ha un manual de C i una guia de programació en C.

Transparències de Síntesi: Als materials de l'assignatura (xWiki) teniu les transparències de síntesi. Per a aquesta pràctica us serà d'especial interès els apartats «Recursivitat I», «Recursivitat II» y «TAD en memòria dinàmica».

Recursos Complementaris

Cercador web: La forma més ràpida d'obtenir informació actualitzada sobre els paràmetres, funcionament i exemples d'utilització de qualsevol mètode estàndard de C (i en general de qualsevol llenguatge), és mitjançant un cercador web.

Criteris de valoració

Per a la valoració dels exercicis es tindrà en compte:

- El codi obté el resultat esperat donades unes condicions i dades d'entrada dissenyades per a provar algunes situacions de funcionament normal i altres casos especials.
- El codi lliurat segueix la guia d'estil i les bones pràctiques de programació.
- Se separa correctament la declaració i implementació de les accions i funcions, utilitzant els fitxers correctes.
- El codi està correctament comentat, valorant especialment la utilització de comentaris en anglès.
- El grau d'optimització en temps i recursos utilitzats a la solució lliurada.
- El codi realitzat és modular i estructurat, tenint en compte la reutilització del codi.
- Es realitza una gestió de la memòria adequada, alliberant la memòria quan sigui necessari.

Enunciat

En el context de les epidèmies per virus d'origen animal molt contagioses i amb alta mortalitat, se'ns proposa el desenvolupament d'un sistema de registre dels diferents casos produits en països al voltant del món.

Les indicacions que ens han donat són les següents:

- S'ha de poder gestionar un conjunt de reservoris del virus, emmagatzemant el seu nom i els animals que en formen part.
- S'ha de poder gestionar un conjunt d'agents infecciosos potencialment perillosos i contagiosos de recent aparició.
- Es vol registrar l'evolució de les infeccions que es produeixen a cada país per aquests agents infecciosos. Per a cadascun, s'emmagatzemarà informació relativa a aquesta.
- S'ha de poder gestionar la informació sobre les epidèmies en les principals ciutats de cada país.
- Es vol registrar la informació del número d'infectats i curats en cada una de les ciutats.
- També es necessita conèixer un indicador sobre si el col·lapse sanitari és a prop de produir-se en un determinat país.
- S'ha de poder afegir per cada un dels països totes aquestes dades.

En aquesta segona pràctica es proposa la implementació de una llista de ciutats per a cadascun dels països i la extracció d'algunes dades estadístiques fent servir recursivitat.

La ciutat tindrà les següents dades: nom, població, número de casos, número de casos crítics, número de recuperats, número de defuncions i número de llits hospitalaris.

El país tindrà les següents dades: nom, una variable que indica si s'està a prop del col·lapse hospitalari i un punter al primer element de la llista de ciutats.

Exercici 1: Creació d'una llista [35%]

Junt amb el enunciat disposeu del codi de la pràctica anterior, en el que hem afegit els fitxers **country.h**, **country.c**, **city.h** y **city.c**. Aquests fitxers contenen els tipus de dades i els mètodes per la gestió de les estadístiques d'infectats en cadascuna de les ciutats d'un país.

Cada país (**tCountry**) conté un punter a la llista de ciutats. La llista de ciutats (**tCityList**) conté un punter al primer element de la llista (**tCityNode**). Els elements de la llista són formats per les dades d'una ciutat (**tCity**) i un punter al següent element de la llista.

En introducir nous tipus de dades s'ha decidit modificar alguns dels tipus de dades existents. Per tant, serà necessari adaptar algunes de les funcions implementades en la PR1. A més a més, alguns dels tests que estaven funcionant, ja no ho faran. D'aquest canvi podem extreure algunes conclusions:

- A vegades ens interessa modificar tipus de dades existents per afegir noves funcionalitats al nostre programa.
- Els tests ens permeten validar que els canvis que hem introduït no afecten al codi existent. Si fallen, ens permeten detectar els errors i corregir-los.

Es demana:

1. Donada la definició de la llista de ciutats del fitxer **city.h**, implementa els mètodes següents en el fitxer **city.c**:
    a. **cityList_create**: Inicialitza la llista de ciutats.
    b. **cityList_empty**: Indica si la llista que es passa és buida (true) o per el contrari conté elements (false).
    c. **cityList_size**: Retorna el número d'elements d'una llista. Si la llista és buida retorna 0.
    d. **cityList_insert**: Afegeix un nou element a la llista a la posició que es passa com a paràmetre (0 per afegir a la capçalera de la llista, INT_MAX per afegir-l'ho al final de la llista). Si l'índex és més gran que el número d'elements, es retornarà l'error ERR_INVALID. Si la ciutat ja existeix a la llista, es retornarà ERR_DUPLICATED. Si hi ha algun error reservant la memòria per al nou element, retornarà un valor ERR_MEMORY_ERROR. Si no hi ha errors, retornarà OK.

2. Donada la definició modificada de **tInfection** en el fitxer **infection.h**, on s'ha modificat el camp **country** per a que sigui un punter del tipus **tCountry**, implementa els mètodes següents en el fitxer **infection.c**:
    a. S'ha de modificar la implementació del mètode **infection_init** en el fitxer **infection.c** per a que inicialitzi el camp **country,** inclosa la llista de ciutats.

3. Donada la definició de **tCountry** en el fitxer **country.h**, implementa els mètodes següents en el fitxer **country.c**:
    a. **country_init**: Inicialitza una estructura **tCountry** amb el nom que es passa com a paràmetre i l'indicador de col·lapse s'inicialitza a fals. També s'haurà de crear la llista de ciutats buida.
    b. **country_addCity**: Afegeix una nova ciutat a la llista de ciutats del país amb les dades de la estructura **tCity** que es passa com a paràmetre.

Nota: Disposeu del mètode **city_init** que us pot ser d'ajuda per a realitzar aquest exercici. La definició de INT_MAX es troba en el fitxer de la llibreria **limits.h** (la llibreria ja es troba referenciada en el fitxer city.c que us donem amb l'enunciat)

Un cop implementats tots els mètodes de l'exercici, haurien de passar correctament tots els tests referents a l'exercici 2 [PR2_EX2_XX]. Els principals conceptes que han de quedar clars després de l'exercici són:

- Definició d'una llista
- Mètodes per a crear o afegir elements a una llista

Exercici 2: Operacions d'accés a una llista [35%]

A l'exercici anterior hem treballat amb els mètodes per a crear una llista i amb els que ens permeten afegir informació. En aquest exercici treballarem amb els mètodes que permeten obtenir informació de la llista, modificar informació i destruir-la.

Es demana:

1. Donada la definició de la llista de ciutats en el fitxer **city.h**, implementa els mètodes següents en el fitxer **city.c**:
    a. **cityList_find**: Buscarà la ciutat amb el nom que es passa com a paràmetre a la llista. Si la llista és buida o no el troba retornarà NULL, si el trobés, retornarà un punter a la ciutat (**tCity**) corresponent.
    b. **cityList_get**: Buscarà la ciutat que es troba a la posició de la llista que es passa per paràmetre. Si la llista és buida o la posició és més gran que el número d'elements retornarà NULL. Si la troba, retornarà un punter a la ciutat (**tCity**) corresponent.
    c. **cityList_delete**: Elimina la ciutat que es troba a la posició de la lista que es passa per paràmetre. Si la llista és buida o la posició és més gran que el número d'elements retornarà false. Si la troba, eliminarà l'element **tCity** i retornarà true.
    d. **cityList_update:** Buscarà la ciutat amb el nom que es passa per paràmetre i actualizarà les següents dades:
        - La data, copiant la que es passa per paràmetre.
        - El número de casos, casos crítics, recuperats i defuncions; sumant als valors actuals, els que es passen per paràmetre.

- La població s'actualitzarà restant del valor actual el número de defuncions que es passen per paràmetre.

e. **cityList_free:** Elimina tots els elements de la llista.

2. Donada la definició modificada de **tInfection** del fitxer ***infection.h***, on s'ha modificat el camp **country** per a que sigui un punter al tipus **tCountry**, implementa els mètodes següents en el fitxer **infection.c**:
   a. Modifica la implementació del mètode **infection_free** per a que elimini el país i les seves ciutats en eliminar la infecció.

Un cop implementats tots els mètodes de l'exercici, haurien de passar correctament tots els tests referents a l'exercici 2 [**PR2_EX2_XX**]. Els principals conceptes que han de quedar clars després de l'exercici són:

- Mètodes per a accedir i el·liminar elements d'una llista
- Com eliminar tots els elements d'una llista

NOTA: en aquest punt ja s'han implementat totes les funcions afectades per el canvi en els tipus de dades. Llavors haurien de tornar a passar correctament tots els test de la PR1.

Exercici 3: Recursivitat [30%]

Quan es treballa amb TADs, una pràctica habitual és la utilització de mètodes recursius per a resoldre problemes. En el fitxer **country.c** es mostra un exemple de dues implementacions diferents per a calcular el total de la població recorrent tots els elements de la llista. El mètode **city_populationIterative** és una implementació iterativa, mentres que el mètode **city_populationRecursive** resol el mateix problema de forma recursiva. Ambdós són implementats a **city.c**.

Es demana implementar les següents funcionalitats a **country.c** y **city.c** de forma **recursiva**:

1. **country_totalCases:** Donat un país compte el número de casos confirmats sumant els casos que hi ha a cadascuna de les ciutats fent servir la funció **cityList_casesRecursive**. Si no hi ha cap ciutat al país es retornarà 0.
2. **country_totalCriticalCases**: Retorna la suma total de casos crítics del país fent servir la funció **cityList_criticalCasesRecursive**. Si no hi ha cap ciutat al país es retornarà 0.
3. **country_totalDeaths**: Retorna la suma total de defuncions del país fent servir la funció **cityList_deathsRecursive**. Si no hi ha cap ciutat al país es retornarà 0.
4. **country_totalRecovered**: Calcula el total de persones donades d'alta en un país fent servir la funció **cityList_recoveredRecursive**. Si no hi ha cap ciutat al país es retornarà 0.

Per últim també es demana implementar les següents funcions al al fitxer **infection.c**:

1. **infection_update_recursive:** Actualitza l'estructura **tInfection** que es passa com a paràmetre amb les dades obtingudes de calcular els casos, casos crítics, defuncions i recuperats de totes les ciutats del país utilitzant les funcions recursives implementades anteriorment.

Un cop implementats tots els mètodes de l'exercici, haurien de passar correctament tots els tests referents a l'exercici 3 [**PR2_EX3_XX**]. Els principals conceptes que han de quedar clars després de l'exercici són:

- Com definir un mètode recursiu