PRÀCTICA 3

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 **21 de maig 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 de cerca i ordenació.

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 «TAD en memòria dinàmica», «TAD de TAD», «Mètodes de cerca» y «Mètodes d'ordenació».

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.
- S'ha de poder generar una llista d'estadístiques d'infecció, severitat i mortalitat per tipus d'infecció.
- Es vol organitzar la informació i les estadístiques de l'epidèmia de tal forma que es puguin realitzar cerques i obtenir llistes ordenades de dades.

En aquesta tercera pràctica es proposa implementar una cerca i ordenació d'infeccions (tInfection) segons el seu grau d'infecció, severitat i mortalitat, obtinguda segons els seus índexs de contagis, casos crítics i defuncions.

Exercici 1: Gestió de les estadístiques d'infecció [35%]

Junt amb l'enunciat disposeu del codi de la pràctica anterior, en el que hem afegit els fitxers **research.h** i **research.c**.

Aquests fitxers contenen els tipus de dades i els mètodes per a la gestió de les estadístiques, la cerca i l'ordenació de les infeccions segons el seu grau d'infecció, severitat i mortalitat en tot el món.

Cada recerca (**tResearch**) conté un apuntador a les estadístiques de la infecció. Ademés de una definició **tResearchListNode** amb doble punter i **tResearchList** com una llista doblement encadenada.

Es demana:

1. Donada la definició de **tInfectionStats** del fitxer research.h implementa els mètodes següents:
    a. Implementa el mètode **research_init**(..) que donades una infecció i una estadística per a aquesta, retorna un element de tipus tError que prèviament inicialitza els camps de **tResearch**. Per a fer-ho, la funció rep com a paràmetre un **tResearch** buit i un tCountry amb les dades sobre el país.
    Per a obtenir els valores correctes de l'estat (stats) de cada país, farem servir cadascuna de les funcions implementades en la pràctica 2, és a dir:
        i. Infectivity = calcula el total d'infectats del país, emprant la funció **country_totalCases.**
        ii. Severity = calcula el total d'infectats crítics del país, emprant la funció **country_totalCriticalCases.**
        iii. Lethality = calcula el total de defuncions del país, emprant la función **country_totalDeaths.**

    b. Implementa el mètode **research_free**(..) que allibera la memòria ocupada per un element de tipus **tResearch**.

    c. Implementa el mètode **research_compare(..)** que donades les estadístiques de dues infeccions les compara per a determinar quina és més perillosa. La funció retorna 1 si la primera obté *millors* estadístiques (guanya com a infecció més perillosa, quelcom que obviament no és positiu des d'un punt de vista humà). Un -1 si la segona obté millors estadístiques i un 0 si totes dues tenen el mateix resultat en les estadístiques. Per a determinar l'ordre es farà segons els següents criteris:
        1. Guanya la infecció que té més **Infectivity**.
        2. En cas d'empat guanya la malaltia amb més **Lethality**.
        3. Si segueix l'empat es considera la quantitat de casos crítics (**Severity**)
        4. Finalment, si segueix l'empat es considera que ambdues infeccions tenen la mateixa estadística.

Una vegada implementats els mètodes especificats en aquest exercici haurien de passar correctament els test referents a l'exercici 1 [PR3_EX1_XX]. Els principals conceptes que han de quedar clars després d'aquest exercici són:

- Càlculs matemàtics bàsics utilitzant un TAD

Exercici 2: Creació d'una llista doblemente encadenada [35%]

A l'exercici anterior hem obtingut una sèrie de resultats i estadístiques on cada sèrie té les seves pròpies dades. Les estadístiques es guarden en una **llista doblemente encadenada** d'investigacions **tResearchList** on els nodes son de tipus **tResearchListNode** i contenen la infecció i les seves estadístiques **tResearch** (el fet d'utilitzar una llista forma part dels requeriments de la realització d'aquest exercici acadèmic).

Es demana que donada la definició de **tResearchList** del fitxer research.h, implementeu els mètodes següents en el fitxer research.c:

1. **researchList_create()**: inicialitza una estructura tResearchList com una llista buida.

2. **researchList_insert()**: Afegeix un element a la llista en una posició concreta. En cas que la posició sigui incorrecta la funció retorna un valor ERR_INVALID_INDEX, si hi ha algún error reservant espai de memòria per al nou element retorna un valor ERR_MEMORY_ERROR. Si no hi ha hagut errors retorna OK.

3. **researchList_delete()**: elimina un element de la llista d'una posició concreta. En cas que la posició sigui incorrecte retorna un valor ERR_INVALID_INDEX, si la llista és buida retorna un valor ERR_EMPTY_LIST. Si no hi ha errors retorna OK.

4. **researchList_get()**: retorna un element de la llista situat en una posició concreta. En cas que la posició sigui incorrecte retorna un valor NULL.

5. **researchList_empty()**: indica si la llista que es passa és buida (true) o, per el contrari, conté elements (false).

6. **researchList_free()**: elimina tots els elements de la llista.

Una vegada implementats tots els mètodes especificats en l'exercici haurien de passar correctament tots els test referents a l'exercici 2 [PR3_EX2_XX]. Els principals conceptes que han de quedar clars després de l'exercici són:

- Definició d'una **llista doblement encadenada**

Exercici 3: Cerca i ordenació d'infeccions [30%]

Treballant amb llistes és usual realitzar recorreguts, cerques o realitzar ordenacions d'aquestes. En aquest exercici es demana que implementeu els següents mètodes:

1. **researchList_getPosByCountry()**: donada una llista d'investigacions de diferents països, que pot ésser desordenada, i un tCountry, retorna la posició del element tResearch dins la llista (1 si és primer, 2 si és segon, etc.). Si no troba la infecció a la llista o aquesta és buida, retornarà -1. **La funció ha d'estar implementada recursivament**.

2. **researchList_bubbleSort**(): donada una llista desordenada d'elements tResearch l'ordena segons les seves estadístiques, amb ordre descendent. S'ha de realitzar l'implementació del algoritme **Bubble Sort** (**ordenació per intercanvi**). La llista resultant tindrà l'element tResearch amb millor estadística en primera posició, el segon en segona posició i així successivament.

   Per a fer-ho es proposa en primer lloc implementar la funció **researchList_swap**(..) per a poder intercanviar dos elements de la llista segons la seva posició. En cas de cridar la funció amb índexs fora de rang (valor superior a la mida de la llista) la funció retorna ERR_INVALID_INDEX. En cas de realitzar correctament l'intercanvi retorna OK.

Nota: es proposa utilitzar la funció **research_compare(..)** como a criteri d'ordenació/comparació de major/menor entre els elements tResearch, segons el seu tipus tInfectionStats.

Un cop implementats tots els mètodes de l'exercici haurien de passar correctament tots els tests referents a l'exercici 3 [**PR3_EX3_XX**]. Els principals conceptes que han de quedar clars després de l'exercici són:

- Cerques dins d'una llista doblement encadenada.
- Ordenació d'una llista doblement encadenada.