Mid-Term Test
Practical Portion
EEE243 Applied Computer Programming
23 October 2014, 08:00 – 10:50 hrs

Examiner: Dr W. Greg Phillips, CD, PhD, PEng

Instructions

- You may begin the practical portion of the exam immediately.
- The practical portion is out of 30 marks with a bonus question worth 3 marks.
- The practical portion of this test is open book. You may refer to course notes and you may access the Internet.
- Communication with any person other than the invigilators during the practical portion of the test **is forbidden**.
- You will not receive an exam booklet for the practical portion of the test; you have received three pages of notepaper instead.
- Write your name and student number on the notepaper, and turn it in at the end of the exam if you wish it to be marked by the examiner.
- Ask the invigilator for more notepaper if you require it.

Hint: Compile and execute your program **often**. You will lose marks if your program does not compile and execute as submitted, or if it generates compiler warnings. Start small and build from there.

Test de Mi-session
Partie Pratique
GEF243 Programmation informatique appliquée
23 octobre 2014, 13h40–16h30

Examinateur: Capt Adrien Lapointe, CD, MSc

Instructions

- Vous pouvez commencer la partie pratique tout de suite.
- La valeur de la partie pratique est de 30 points, avec une question boni qui vaut 3 points.
- La partie pratique de l'examen est à livre ouvert. Vous pouvez faire référence aux notes du cours et vous pouvez accéder à l'Internet.
- La communication avec toutes personnes autre que les surveillants pendant la partie pratique de l'examen **est interdite**.
- Vous ne recevrez pas de livret d'examen pour la partie pratique du test; vous avez plutôt reçu trois pages de papier de notes.
- Écrivez votre nom et numéro de collège sur le papier de notes, et remettez-le à la fin de l'examen si vous voulez qu'elles soient corrigées par l'examinateur.
- Nous avons amplement de papier de notes; demandez en plus si vous en avez besoins.

Indice : Compilez et exécutez votre code **souvent**. Vous serez pénalisé si votre programme ne compile pas ou n'exécute pas à la fin. Commencez petit et ajoutez des fonctionnalités au fur et à mesure.

Introduction

Open the `petridish` project that you created at the beginning of the examination.

Add your name and college number to the comment block at the beginning of the `petri.c` file. You may need to click the ⊕ at the top left of the file to reveal the comment block.

The code provided is the skeleton of a simulation of a Petri dish, the container used in laboratories for bacterial cultures. The dish is represented by a square array, where we insert bacteria in one or more cells. In each simulation turn, cells that are adjacent to already-infected cells also become infected.

Your job is to write a part of the simulation program. This includes a function to display the current state of the Petri dish, a function to determine whether a given cell has any infected neighbours, and a function to advance the simulation by one turn.

You must provide evidence of your design for each part of the tasks. Clear, well-structured code that works and includes appropriate comments is sufficient. However, if your code does not work or is not clear or well structured, you may receive part marks for design documentation in the form of flowcharts, pseudo-code, memory diagrams, or textual descriptions of your design. Write these on the provided notepaper.

Introduction

Ouvrez le projet `petridish` que vous avez créé au début de l'examen.

Ajoutez votre nom et numéro de collège au bloc commentaire au début du fichier `petri.c`. Il sera peut-être nécessaire de cliquer sur le symbole ⊕ en haut à gauche du fichier pour voir le bloc commentaire.

Le code fourni est le squelette d'une simulation de plaque de Pétri, le contenant utilisé en laboratoire pour les cultures bactériennes. La plaque est représentée par un tableau carré, où nous insérons une bactérie dans une ou plusieurs cases. À chaque tour de simulation, les cases adjacentes à une bactérie sont infectées à leur tour.

Votre tâche est d'écrire une partie du programme de simulation. Plus précisément, une fonction affichant l'état courant de la plaque de Pétri, une fonction pour déterminer si une case donnée est adjacente à une case infectée et une fonction pour faire avancer la simulation d'un tour.

Vous devez fournir une des traces de votre conception pour chaque partie des tâches. Le code clair et bien structuré qui fonctionne et qui inclus des commentaires appropriés est suffisant. Toutefois, si votre code ne fonctionne pas ou s'il n'est pas clair ou bien structuré, vous pouvez recevoir des points partiels pour la documentation de conception sous la forme d'organigrammes, pseudo-code, les schémas de la mémoire ou de descriptions textuelles de votre conception. Écrivez-les sur le papier de notes fournit.

Requirements

You have been provided with three files: `main.c`, `petri.c` and `petri.h`. In `petri.c` you must implement at least the three given functions, without modifying their signatures. In `main.c` you must write code that tests and demonstrates successful execution of your program. You may also implement any other functions that you judge necessary.

Petri dish representation

The Petri dish is represented by a square, two-dimensional array of integers, each of which is called a "cell". Cells that are infected are indicated by the value 1. Cells that are not infected are indicated by the value 0. The size of the dish is specified by `EDGE_LENGTH` in `petri.h`.

`main()`

The `main()` function, in `main.c`, must do the following:

- declare an array called `petri_dish` of the appropriate size;
- initialize the dish with a mix of infected and uninfected cells, appropriate to test the function of the rest of your program
- print the initial state of the array using `print_petri_dish`
- advance the simulation by one turn by calling `spread_infection`
- print the new state of the dish using `print_petri_dish`

For testing purposes, your `main()` function may repeat this process for several different initial dishes.

`print_petri_dish()`

This function prints the Petri dish that is passed to it as a parameter. An example of an appropriate printout is given below. Test your implementation of this function before continuing.

Exigences

Nous vous fournissons trois fichiers : `main.c`, `petri.c` et `petri.h`. Dans `petri.c`, vous devez implémenter au moins trois fonctions données sans en modifier leurs signatures. Dans `main.c`, vous devez écrire du code pour tester et démontrer que votre programme exécute correctement. Vous pouvez implémenter d'autres fonctions que vous jugez nécessaire.

Représentation de la plaque de Pétri

La plaque de Pétri est représentée par un tableau d'entiers carré à deux dimensions où chaque élément représente une case. Les cases infectées ont une valeur de 1 tandis que celles qui ne le sont pas ont une valeur de 0. La taille de la plaque est spécifiée par `EDGE_LENGTH` dans `petri.h`.

`main()`

La fonction `main()` dans `main.c` doit :

- déclarer un tableau d'entiers appelé `petri_dish` de la bonne grandeur
- initialiser la plaque avec un mélange de case infectées et non-infectées approprié pour tester le reste du programme
- afficher l'état initial du tableau en utilisant `print_petri_dish`
- faire avancer la simulation d'un tour en appelant `spread_infection`
- afficher le nouvel état de la plaque avec `print_petri_dish`

Dans le but de tester votre programme, votre fonction `main()` peut répéter ce processus pour plusieurs plaques initiales.

`print_petri_dish()`

Cette fonction affiche la plaque de Pétri qui lui est passée en paramètre. Vous verrez ci-dessous un exemple d'affichage approprié. Testez votre implémentation avant de continuer.

```
1 1 0 1 1 1 0 1 1
1 1 0 1 1 1 0 1 1
0 0 1 1 1 0 0 0 0
1 1 1 1 1 0 0 0 0
1 1 1 1 1 0 0 1 1
1 1 0 0 0 0 0 1 1
0 0 0 1 1 1 0 1 1
1 1 0 1 1 1 0 1 1
1 1 0 1 1 1 0 1 1
```

`are_adjacent_infected()`

This function will be used by the `spread_infection` function. Given a petri dish and the row and column index of a cell in the dish, it determines whether or not the cell has any infected neighbours. It returns `true` if one or more of the cell's neighbours is infected, `false`

`are_adjacent_infected()`

Cette fonction sera utilisée par la fonction `spread_infection`. Elle détermine si une case est adjacente à une case infectée lorsqu'on lui fourni une plaque, un numéro de rangée et de colonne. La fonction retourne `true` si une ou plusieurs cases adjacentes sont

otherwise.

Take care near the edges of the dish! You must not attempt to access a cell outside the dish, since this is memory you don't own. The edges of the dish do not "wrap", so the top edge does not connect to the bottom and the left does not connect to the right. Figure 1 below illustrates an infected cell (the 1) and the cells that are considered adjacent to it (the A's) – all cells that are blank in this figure would have the value 0. The cells marked A are the ones for which this function would return `true`.

infectées et `false` sinon.

Soyez très prudent près des bords de la plaque. Vous ne devez pas essayer d'accéder des cases à l'extérieur, puisque c'est de la mémoire qui ne vous appartient pas. Notez que lorsque vous êtes sur les bords de la plaque, les cases de l'autre côté ne sont pas adjacentes. Par exemple, si vous êtes sur le bord gauche de la plaque, les cases à droite ne sont pas adjacentes. La Figure 1 illustre cet exemple. La case infectée contient 1 et les cases adjacentes un A, toutes les cases vides auraient la valeur 0. La fonction retournerait la valeur `true` pour les cases contenant un A.

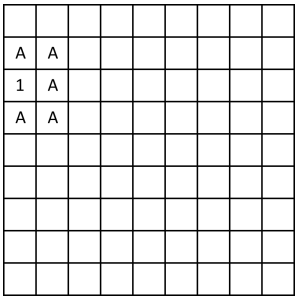

Figure 1: Adjacent cells / Cases adjacentes

You should test your `are_adjacent_infected()` function before continuing.

Testez votre fonction `are_adjacent_infected()` avant de continuer.

`spread_infection()`

This function advances the simulation by one turn, updating the `petri_dish` array by infecting any cell that is adjacent to an already-infected one.

Cette fonction fait avancer la simulation d'un tour en mettant à jour le tableau `petri_dish` en infectant les cases qui sont adjacente à une case déjà infectée.

Figure 2 illustrates an initial state of the Petri dish with only one infected cell (again, the cells shown as empty here would contain the value 0). Figure 3 shows what the dish would look like after advancing the simulation by one turn.

La Figure 2 illustre la plaque de Pétri dans son état initial avec seulement une case infectée (les cases vides contiendraient la valeur 0). La Figure 3 illustre le tableau après un tour de simulation.

It should be obvious that in a full simulation, every cell in the dish would rapidly become infected.

Il est évident que dans le programme de simulation complet toutes les cases seraient rapidement infectées.

Ensure you thoroughly test that your simulation works correctly for all interesting cases.

Assurez vous que la simulation fonctionne correctement dans tous les cas intéressants.

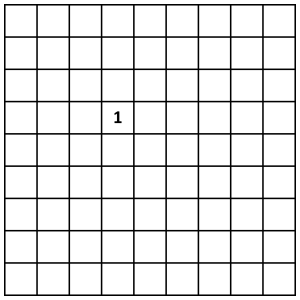

Figure 2: Initial bacteria / Bactérie initiale

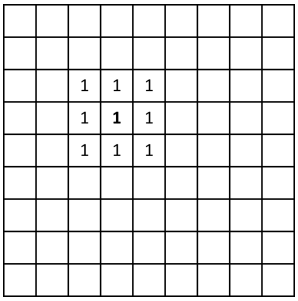

Figure 3: Bacterias after infection / Les bactéries après l'infection

Bonus task

Modify your `main()` program so that it runs simulation turns repeatedly until all cells have been infected, then stops. Your program must print the state of the dish after each turn.

For full bonus marks, your `main()` function must be cleanly structured and divided into functions where appropriate.

When you are finished

Ensure your name is in the header comment of the files and on your notes pages. Leave the computer logged in and your project open in Eclipse, and leave your notes pages on the keyboard.

End of the examination

3
bonus

Tâche boni

Modifiez votre fonction `main()` pour qu'elle exécute les tours de simulation jusqu'à ce que toutes les cases soient infectée, puis arrête. Votre programme doit afficher l'état de la plaque après chaque tour.

Pour recevoir tous les points bonis, votre fonction `main()` doit être proprement structurée et divisée en fonctions au besoin.

Lorsque vous avez terminé

Assurez-vous que votre nom est dans le bloc commentaire des fichiers et sur vos pages de notes. Laissez l'ordinateur connecté et votre projet ouvert dans Eclipse, et laissez vos pages de notes sur le clavier.

Fin de l'examen