Final Examination
Practical Portion
EEE243 Applied Computer Programming
7 December 2017

Examiner: T. Karamat, PhD

Examen Final
Partie Pratique
GEF243 Programmation informatique appliquée
7 décembre 2017

Examinateur: Capt Adrien Lapointe, CD, MSc

Instructions

- You may begin the practical portion of the exam immediately.
- The practical portion is out of 65 marks.
- The practical portion of this examination is open book. You may refer to course notes and your code.
- 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 can request notepaper as needed.
- If you wish your notepaper to be marked by the examiner, write your name and student number on it and turn it in at the end of the exam.

Instructions

- Vous pouvez commencer la partie pratique de l'examen tout de suite.
- La valeur de la partie pratique est de 65 points.
- La partie pratique de l'examen est à livre ouvert. Vous pouvez faire référence aux notes du cours et à votre code.
- 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 pouvez demander des pages de papier de notes si nécessaire.
- Si vous voulez que les pages de notes soient corrigées par l'examinateur, écrivez-y votre nom et numéro de collège et remettez-les à la fin de l'examen.

Hints: 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. Test your code as you go. Start small and build from there.

Draw memory diagrams of all pointer manipulations before writing the code!

Indices : Compilez et exécutez votre code **souvent**. Vous serez pénalisé si votre programme ne compile et n'exécute pas ou s'il produit des avertissements. Testez votre code souvent. Commencez petit et ajoutez des fonctionnalités au fur et à mesure.

Dessinez des diagrammes de mémoire pour toutes les manipulations de pointeurs avant de coder.

Introduction

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

If you haven't already done so, **add your name** to the comment block at the beginning of the `student.c` file.

Ensure that you respect the interfaces that are provided to you in the header files. We will mark the exam using automated tests. Therefore any modifications you make to the function prototypes will break those tests and we will not be able to give you any marks.

Make sure your code compiles.

Requirements

You have been provided with six files: `student.c`, `student.h`, `main.c`, `support.c`, `support.h` and `sphere.h`.

In `student.c` you must implement the functions outlined in the four questions.

You **must not modify** `student.h`, `support.c`, `support.h` and `sphere.h`.

You can modify `main.c` but it will not be marked. It is where you can run the functions you implemented. You will notice that some code is commented out in that file. You may use that code and modify it for your tests.

Question 1

Colours are often encoded using three components: red, green, and blue (RGB). Each component is usually represented using one byte (0 to 255 values). This representation is called "true colour" or sometimes 16 million colours given that this 24-bit representation allows for 256^3=16,777,216 colour variations.

We want to create a function that converts a colour from the true colour space to an 8-colour space. The colours for the 8-colour space are defined as the

Introduction

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

Si vous ne l'avez pas encore fait, **ajoutez votre nom** au bloc commentaire au début du fichier `student.c`.

Assurez-vous de respectez les interfaces qui vous sont fournies dans les fichiers d'en-tête. Nous corrigerons l'examen en utilisant des tests automatisés. Ainsi, toutes modifications aux prototypes de fonction briseront les tests et nous ne pourrons pas vous donner de points.

Assurez-vous que votre code compile.

Exigences

Six fichiers vous sont fournis : `student.c`, `student.h`, `main.c`, `support.c`, `support.h` et `sphere.h`.

Dans `student.c`, vous devez implémenter les fonctions indiquées dans les quatre questions.

Vous **ne devez pas modifier** `student.h`, `support.c`, `support.h` et `sphere.h`.

Vous pouvez modifier `main.c`, mais le fichier ne sera pas corrigé. C'est l'endroit où vous pouvez appeler les fonctions que vous avez créées. Vous remarquerez que du code a été mis en commentaires dans ce fichier. Vous pouvez utiliser ce code et le modifier pour vos tests.

15 Question 1

Les couleurs sont souvent représentée en utilisant trois composantes: rouge, vert et bleu (RVB ou RGB en anglais). Chaque composante est généralement représentée en utilisant un octet (de 0 à 255 valeurs). Cette représentation est appelée « couleur vraie » ou parfois 16 millions de couleurs étant donné que 24 bits permet 256^3 = 16 777 216 variations de couleurs.

Nous voulons créer une fonction qui convertit une couleur de l'espace de couleurs vraies en un espace à 8 couleurs. Les couleurs de l'espace à 8 couleurs

enumeration `StdCol` in `student.h`.

The true colour space can be represented in three dimensions with each of the red, green, and blue being an axis. The possible values form a cube and the intent here is to equally divide the cube into 8 smaller ones labelled with one of the colours defined in `StdCol`.

The sub-cube within which a true-colour falls will then belong to the associated `StdCol`. For instance, the colour `(12, 23, 46)` would be `BLACK`.

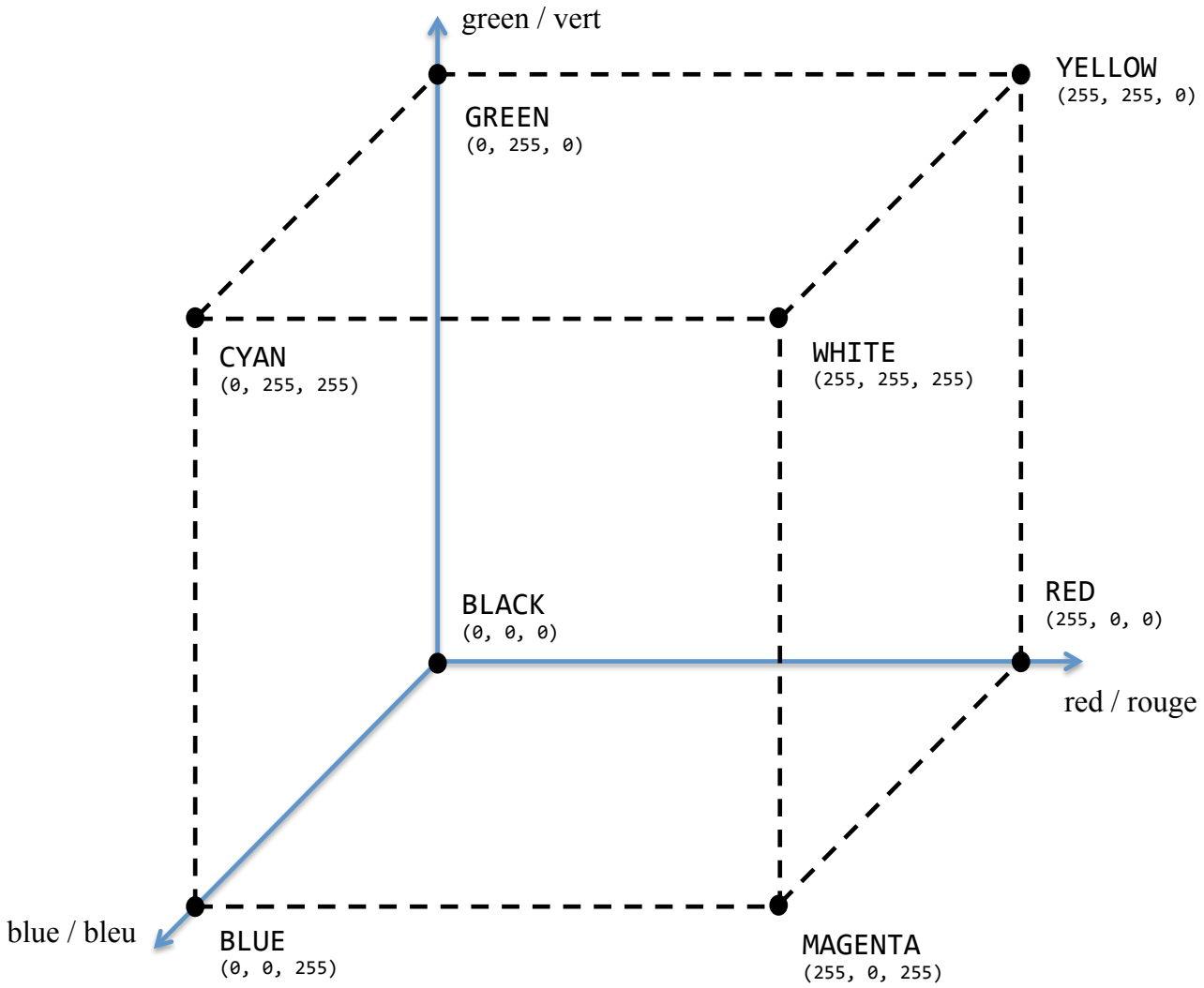
The graph below illustrates the distribution of colours in the RGB space.

sont définies par l'énumération `StdCol` dans `student.h`.

L'espace des couleurs réelles peut être représenté en trois dimensions, le rouge, le vert et le bleu étant chacun un axe. Les valeurs possibles forment un cube et l'intention ici est de le diviser également en 8 plus petits cubes étiquetés avec l'une des couleurs définies dans `StdCol`.

Le sous-cube dans lequel tombe une couleur vraie appartiendra alors au `StdCol` associé. Par exemple, la couleur `(12, 23, 46)` serait `BLACK`.

Le graphique ci-dessous illustre la répartition des couleurs dans l'espace RVB.



You need to write the code in the function `convert_col` to convert a true colour to the 8-colour space. We provide you with the constant `COLOR_THRES` to help divide the colour space

Vous devez écrire le code dans la fonction `convert_col` pour convertir une couleur vraie en l'espace à 8 couleurs. Nous vous fournissons la constante `COLOR_THRES` pour diviser l'espace

evenly.

Your function must have the following prototype that is provided in your starting code.

colorimétrique uniformément.

Votre fonction doit avoir le prototype suivant qui est fourni dans votre code de départ.

```
enum StdCol convert_col(unsigned char color[3])
```

Note that the function returns the converted color as a value from the enumeration `StdCol`.

Notez que la fonction retourne la couleur convertie en tant que valeur provenant de l'énumération `StdCol`.

Question 2

20 Question 2

We provide you an incomplete version of an action history system as it could be implemented to track the user actions in a text editor. Such functionality is useful to undo and redo actions.

Nous vous fournissons une version incomplète d'un système d'historique des actions. Celui-ci pourrait être implémenté pour enregistrer les actions de l'utilisateur dans un éditeur de texte. Une telle fonctionnalité est utile pour annuler et refaire des actions (*undo* et *redo*).

Each action is represented as a `struct Action` element and they are held in a doubly linked list. There is a special node at the head of the list that represents the original document with no action applied to it.

Chaque action est représentée en tant qu'élément `struct Action` qui est contenu dans une liste doublement chaînée. Il y a un nœud spécial en tête de liste qui représente le document original sans qu'aucune action ne lui soit appliquée.

We simulate the user's action with the function `create_Action` that randomly selects one of the 3 types of actions. The function `add_node` adds the new action to the list.

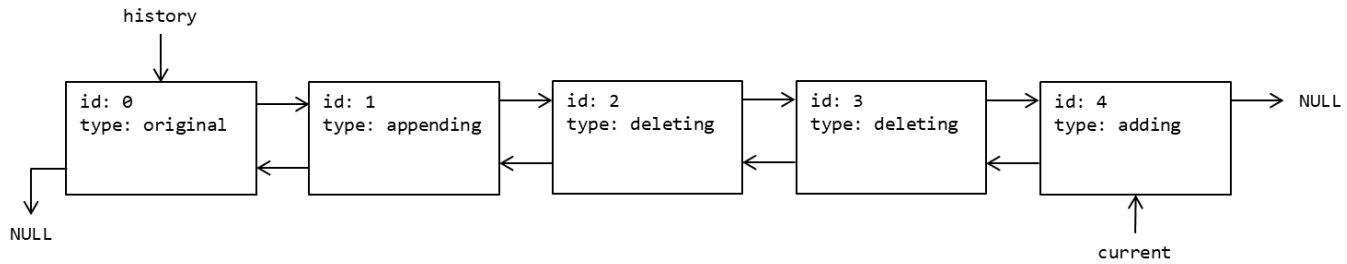
Nous simulons les actions de l'utilisateur avec la fonction `create_Action` qui sélectionne aléatoirement l'un des 3 types d'actions. La fonction `add_node` ajoute la nouvelle action à la liste.

We provide you the necessary functions to initialize the list and print its content.

Nous vous fournissons les fonctions nécessaires pour initialiser la liste et afficher son contenu.

The figure below shows the list as it could be after a few actions have been done to the original document and no undo has been done.

La figure ci-dessous montre la liste telle qu'elle pourrait être après que quelques actions aient été effectuées sur le document original et qu'aucune annulation n'ait été effectuée.

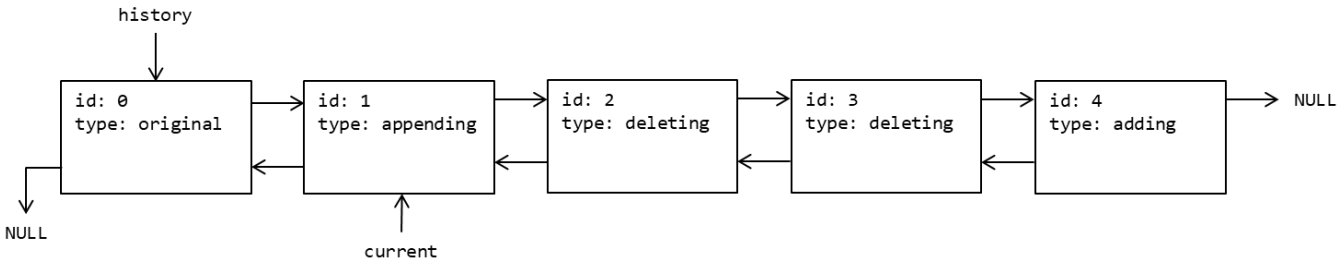


The completed system should allow the user to undo until the image is restored to its initial state and to redo until the end of the list is reached. The pointer `current` indicates where in the history the system is.

Le système complété devrait permettre à l'utilisateur d'annuler jusqu'à ce que l'image soit restaurée à son état initial et de refaire jusqu'à ce que la fin de la liste soit atteinte. Le pointeur actuel indique où se trouve le système dans l'historique.

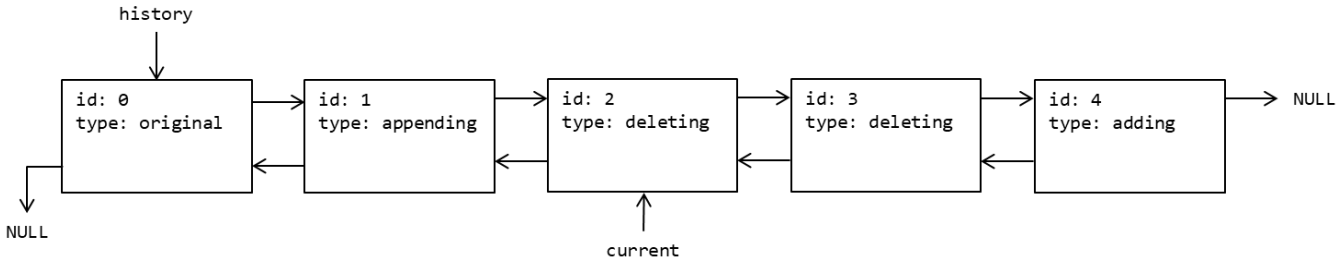
The figure below shows the list after three undo.

La figure ci-dessous montre la liste après trois annulations.



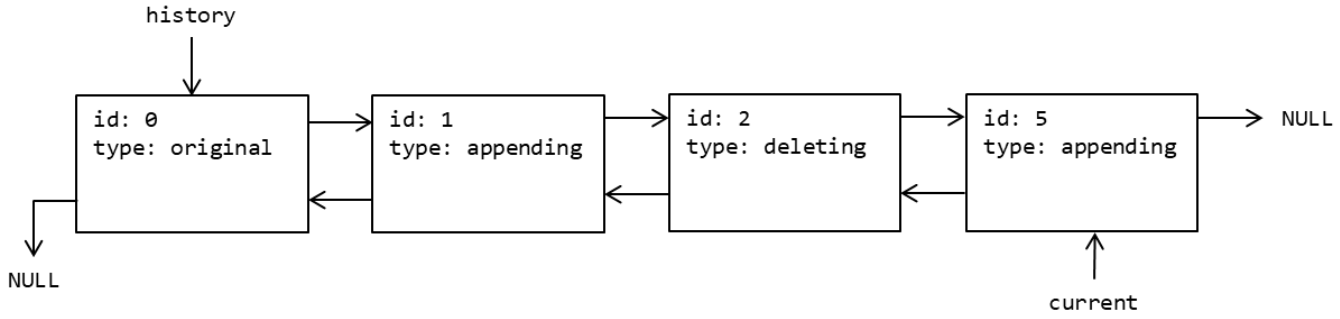
Now if we were to do one redo, we would end up with the following list.

Maintenant, si nous refaisions une fois, nous obtiendrions la liste suivante.



Any new action that is added will cause all the following actions in the history to be lost and the new action added after the current state as illustrated in the figure below.

Toute nouvelle action ajoutée entraînera la perte de toutes les actions suivantes dans l'historique et l'ajout de la nouvelle action après l'état actuel, tel qu'illustré dans la figure ci-dessous.



a. You need to implement the `redo` function with the behavior described above and with the following prototype.

a. Vous devez implémenter la fonction `redo` avec le comportement décrit ci-dessus et avec le prototype suivant.

```
struct Action *redo(struct Action *current)
```

b. You need to implement the `undo` function with the behavior described above and with the following prototype.

b. Vous devez implémenter la fonction `undo` avec le comportement décrit ci-dessus et avec le prototype suivant.

```
struct Action *undo(struct Action *current)
```

c. We provide you with the code to add nodes to the list. It currently results in a memory

c. Nous vous fournissons le code pour ajouter des nœuds à la liste. Il en résulte

leak because the nodes after the current one are not properly deleted. You need to implement the `delete_following_action` function so that it properly deletes the nodes. Your function must have the following prototype.

actuellement une fuite de mémoire car les nœuds après celui en cours ne sont pas correctement supprimés. Vous devez implémenter la fonction `delete_following_action` afin qu'elle supprime correctement les nœuds. Votre fonction doit avoir le prototype suivant.

```
void delete_following_action(struct Action *entry_point)
```

Question 3

10 Question 3

A tool that is often used in image processing is the histogram. It counts the number of pixels that have a given intensity.

Un outil souvent utilisé dans le traitement d'image est l'histogramme. Celui-ci compte le nombre de pixels ayant une intensité donnée.

In a grey scale image where each pixel can take a value between 0 and 255, the histogram would have 256 entries. If a given image is 42x42 and all black, the entry for 0 in the histogram would contain the value 1764 (42 x42) and all the other entries 0. If the image was half black and half white, the entries for 0 and 255 in the histogram would both contain 882, and all the other entries 0.

Dans une image en tons de gris où chaque pixel peut prendre une valeur comprise entre 0 et 255, l'histogramme aura 256 entrées. Si une image donnée est 42x42 est toute noire, l'entrée pour 0 dans l'histogramme contiendra la valeur 1764 (42 x 42) et toutes les autres entrées 0. Si l'image était moitié noire et moitié blanche, les entrées 0 et 255 de l'histogramme contiendraient toutes deux 882 et toutes les autres entrées 0.

You need to implement the function to generate the histogram for the image of a sphere represented as the array `img` defined in `sphere.h`. We provide you with the function `show_histogram` to display the histogram in the console. Note that the function condenses the histogram so it does not show the entries with a count of 0.

Vous devez implémenter la fonction pour générer l'histogramme de l'image d'une sphère représentée comme le tableau `img` défini dans `sphere.h`. Nous vous fournissons la fonction `show_histogram` pour afficher l'histogramme dans la console. Notez que la fonction condense l'histogramme de sorte qu'il n'affiche pas les entrées avec un compte de 0.

Your function must have the following prototype that is provided in your starting code.

Votre fonction doit avoir le prototype suivant fourni dans votre code de départ.

```
unsigned int *generate_histogram(unsigned char image[IMG_SIZE][IMG_SIZE], unsigned char range)
```

Note that the `range` parameter indicates the maximum value covered by the histogram. In our previous example, it would have been 255.

Notez que le paramètre `range` indique la valeur maximale couverte par l'histogramme. Dans notre exemple précédent, il aurait été 255.

Question 4

20 Question 4

You need to implement a function that counts the number of words and the number of lines in a file (`lorem.txt`) and puts the results in another file (`final.txt`) using the following format:

Vous devez implémenter une fonction qui compte le nombre de mots et le nombre de lignes dans un fichier (`lorem.txt`) et mettre les résultats dans un autre fichier (`final.txt`) en suivant ce format:

"There are *WW* words and *LL* lines in the document"

« Il y a *WW* mots et *LL* lignes dans le document »

Where *WW* is the number of words and *LL* is the number of lines your program counted.

Note that there may be multiple spaces between words, which you may need to consider.

You can assume that there are no spaces at the end of the file, that lines do not start with spaces, and that the input file is not empty.

Your function must have the following prototype that is provided in your starting code.

Où *WW* est le nombre de mots et *LL* est le nombre de lignes que votre programme a compté.

Notez qu'il peut y avoir plusieurs espaces entre les mots, ce que vous devrez prendre en compte.

Vous pouvez supposer qu'il n'y a pas d'espaces à la fin du fichier, que les lignes ne commencent pas par des espaces et que le fichier d'entrée n'est pas vide.

Votre fonction doit avoir le prototype suivant fourni dans votre code de d.part.

```
void count_words_and_lines(char *in, char *out)
```

The parameters `in` and `out` are strings that contain the filenames for the input file and the output file respectively.

When you are finished
Ensure your name is in the header comment of the `student.c` file. Leave the computer logged in and your project open in Eclipse, and leave your notes pages on the keyboard. Leave quietly by the nearest door.

End of the examination

Les paramètres `in` et `out` sont des chaînes de caractères qui contiennent les noms de fichiers pour les fichiers d'entrée et de sortie respectivement.

Lorsque vous avez terminé
Assurez-vous que votre nom est dans le bloc commentaire du fichier `student.c`. Laissez l'ordinateur connecté et votre projet ouvert dans Eclipse. Laissez aussi vos pages de notes sur le clavier. Partez discrètement par la porte la plus proche.

Fin de l'examen