Projet Algo 5 - Partie 1

Licence Informatique - 2nde année

Année 2021-2022

L'objectif général de ce projet est de mettre en pratique les algorithmes de parcours de graphe vus en cours, sur un exemple concret. Celui-ci consistera à explorer un labyrinthe et à en trouver la sortie.

Organisation du projet

Le projet sera découpé en 3 partie successives, permettant d'aboutir à sa réalisation. Chaque partie sera à rendre à une date fixée et une correction vous sera fournie après chacune de ces dates, afin de pouvoir poursuivre les parties suivantes dans les meilleures conditions.

Le projet est à réaliser seul et les fichiers sources demandés devront être rendus par mail. Les fichiers devront contenir en commentaires le nom de l'étudiant à qui ils appartiennent ; le nom de fichier à rendre devra également contenir ce nom. Par exemple, si le fichier à rendre se nomme `grille.cpp` et que l'étudiant se nomme `toto`, le fichier rendu devra porter le nom `grille_toto.cpp`.

Au cours de la correction, si plusieurs projets devaient être semblables dans leur contenu, la note attribuée à la réalisation se verra diviser par le nombre de copies retrouvées.

Première partie

Dans cette première partie, vous allez expérimenter l'utilisation de fonctions permettant de générer une image au format SVG (*Scalable Vector Graphics*). Ceci vous permettra par la suite de visualiser (i) les labyrinthes qui seront créés et (ii) les chemins/graphes qui y seront associés.

Principe du SVG

Le format d'images SVG est un format **vectoriel**, c'est à dire que, plutôt que de sauvegarder la couleur de chaque pixel d'une image comme le font les formats classiques tels le PNG ou encore le JPEG, il contient des instructions de dessins permettant de reconstituer l'image : dessins de lignes, de figures géométriques, de courbes, etc. (voir figure 1).

```
<?xml version="1.0" encoding="utf-8"?>
<svg xmlns="http://www.w3.org/2000/svg"
  version="1.1" width="300" height="300">
  <rect width="300" height="300"
   x="0" y="0" fill="white" />
  <polygon points="100,10 40,198 190,78
    10,78 160,198" style="fill:lime;stroke:
    purple;stroke-width:5;fill-rule:nonzero;"/>
  <circle cx="250" cy="250" r="40" stroke="black"
    stroke-width="3" fill="red" />
</svg>
```

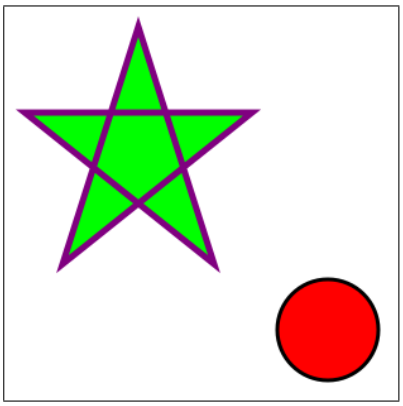


FIGURE 1 – Un exemple d'instructions graphiques contenues dans un fichier au format SVG (à gauche) et l'image correspondante (sur la droite) affichée lors de l'ouverture du fichier par un outil de visualisation.

Pour vous éviter de vous plonger dans les règles d'écriture de ce format, le fichier `grille.cpp` qui vous est fourni contient des fonctions permettant de générer certaines primitives graphiques dans le fichier SVG qui représentera une image :

— `bool ouvrirFichier(const string nomFichier, ofstream &out, int largeur, int hauteur)` : Cette fonction permet d'ouvrir le fichier qui doit servir à la sauvegarde des instructions de dessin. Le premier paramètre représente le nom qu'aura le fichier (extension `.svg`), le second paramètre représentera la variable de type fichier qui sera intialisée par la fonction et utilisée par les fonctions suivantes et les paramètres `largeur` et `hauteur` représenteront la taille de l'image en nombre de pixels par colonne et par ligne. Elle retourne `true` si l'ouverture s'est effectuée correctement, `false` sinon ;

— `void fermerFichier(ofstream &out)` ; cette fonction doit être appelée à la fin de l'écriture des instructions de dessin, afin de clore proprement le fichier SVG ;

— `void ligne(ofstream &out, int x1, int y1, int x2, int y2, const string& color, int width)` : cette fonction permet de tracer une ligne dans le fichier SVG, entre les pixels `(x1,y1)` et `(x2,y2)`. Une couleur de tracé doit être précisée (voir ci-après), ainsi qu'une épaisseur (en nombre de pixels) ;

— `void rect(ofstream &out, int x, int y, int width, int height, const string &color)` permet de générer un rectangle dont le coin supérieur gauche est aux coordonnées (x, y), et dont les dimensions sont fournies par les paramètres `width` et `height` en nombre de pixels). La couleur fournie à la fonction corrrespond à la couleur de remplissage du rectangle (ses contours sont de la même couleur) ;

— `void texte(ofstream &out, int x, int y, int size, const string &txt, const string &color)` permet de générer un texte (paramètre `txt`), dont le centre sera situé aux coordonnées (x, y). Le paramètre size permet de préciser la hauteur (en pixels) que doivent avoir les caractères.

Les couleurs Elles sont représentées par une chaîne de caractères, qui correspond au nom de la couleur. Par exemple, `white`, `black`, `red` sont des couleurs valides. Vous pouvez facilement trouver la liste des noms de couleurs valides sur le web.

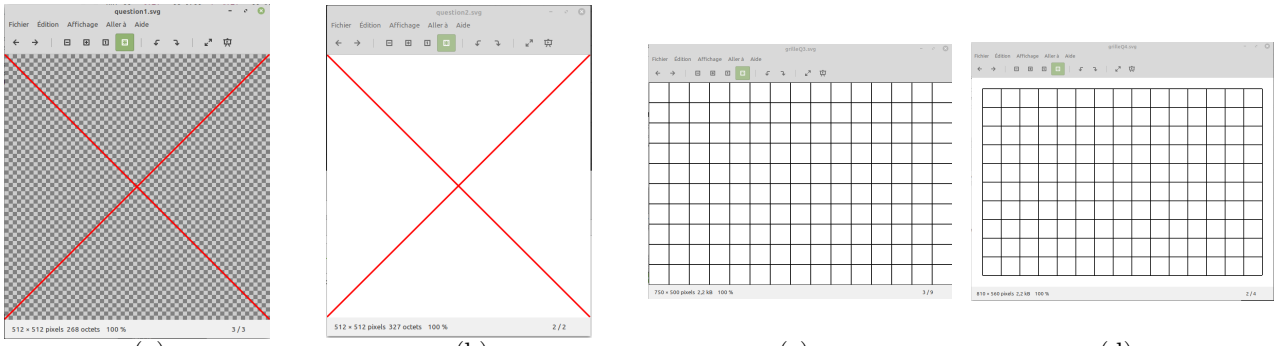
Système de coordonnées Une image sera définie par ses dimensions en largeur et en hauteur, qui seront exprimées en nombre de pixels. Les coordonnées de chaque pixel se trouveront dans les intervalles $[0, largeur - 1]$ pour les colonnes et $[0, hauteur - 1]$ pour les lignes, avec une origine (pixel de coordonnées $(0, 0)$) située **en haut et à gauche** de l'image.

Application

1. ajoutez au fichier `grille.cpp` la fonction suivante :

   ```
   void dessiner(ofstream &out, int largeur, int hauteur)
   ```

   Cette fonction doit permettre de dessiner une croix rouge dans une image de taille $largeur \times hauteur$, qui sera sauvée dans un fichier dont le nom sera `image01.svg` (voir figure 2.a). Le premier paramètre `out` doit correspondre au fichier svg dans lequel dessiner l'image, qui aura dû être ouvert auparavant ;



(a) (b) (c) (d)

FIGURE 2 – Les différentes images qui doivent être obtenues pour les 4 questions de cet exercice..

2. par défaut, le fond de l'image reste transparent. Modifiez la fonction pour qu'elle rajoute un rectangle blanc de même taille que l'image, afin que le fond de l'image soit blanc (voir figure 2.b). Vous prendrez garde à l'ordre dans lequel les deux figures seront sauvegardées dans le fichier, puisqu'elles seront dessinées dans le même ordre au moment du chargement de l'image ;
3. ajoutez la fonction suivante à votre application :

   ```
   void dessinerGrille(ofstream &out, int nbc, int nbl)
   ```

pour qu'elle dessine une grille de cellules carrées dans l'image (voir figure 2.c). Le nombre de colonnes de la grille sera fourni par le paramètre `nbc` et son nombre de ligne par le paramètre `nbl`. La taille des cellules (côté d'une cellule en nombre de pixels) devra être définie par une constante `CELLSIZE`. Il est conseillé d'utiliser la fonction `ligne` fournie pour tracer le quadrillage demandé et vous prendrez garde à créer une image qui soit de la bonne dimension lors de l'appel de la fonction `ouvrirFichierSVG`.

4. Modifiez votre fonction et l'appel à la fonction `ouvrirFichierSVG`, pour que votre application rajoute une marge sur chaque bord de l'image (voir figure 2.d). La dimension de la marge, représentée par une constante `MARGE`, sera la même pour chaque bord de l'image.