TP3+ : Dessin ASCII dans le terminal

MP2I Lycée Pierre de Fermat

Le but de cet exercice est de créer un logiciel (très basique) de dessin. L'écran du dessin sera simulé dans le terminal, et sera une grille de dimensions $n \times m$. Chaque case de cette grille contiendra un caractère ASCII, et on pourra avec le logiciel tracer diverses formes sur cette grille.

Question 1. Créez un nouveau fichier C, et définissez y (avec `#define`) deux constantes N et M, qui seront respectivement le nombre de lignes et de colonnes de la grille. Prenez par exemple $N = 24$, $M = 30$.

La grille sera un tableau 2D, qui sera une variable globale :

```
1 char grille[N][M];
```

Question 2. Écrivez une fonction qui affiche la grille. Dans la fonction main, écrivez un bout de code qui initialise toutes les cases de la grille au caractère espace, et mettez à la main des caractères dans la grille afin de tester l'affichage.

Nous allons maintenant implémenter des fonctions permettant à l'utilisateur/ice de dessiner sur l'écran. On souhaite pouvoir écrire des commandes comme :

```
draw char 13 12 #   // met le caractère # aux coordonnées (13, 12)
draw line 2 4 13 6  // dessine une ligne entre les points (2, 4) et (13, 6)
```

Il faut faire attention que dans le système de coordonnées habituel, l'axe des x croît vers la droite, et l'axe des y croît vers le haut. Mais dans un tableau 2D, la première case affichée est celle d'indice [0][0]

Question 3. Écrire une fonction `void xy_to_ij(int x, int y, int* i, int* j)` qui prend en entrée une coordonnée (x, y) et renvoie le couple (i, j) (en stockant les valeurs dans les cases pointées par les deux derniers paramètres) tel que la case $[i][j]$ de la grille correspond au point de coordonnées (x, y). Par exemple, pour $(x, y) = (0, 0)$, le couple correspondant sera $(i, j) = (N - 1, M - 1)$.

Question 4. Écrire une fonction `void draw_char(int x, int y, char c)` qui dessine le caractère c aux coordonnées (x, y). Testez bien cette fonction et la précédente.

Question 5. Dans le main, implémentez une boucle permettant à l'utilisateur/ice de choisir entre deux commandes : `draw` et `quit`. La commande `quit` a pour effet d'arrêter immédiatement le programme. La commande `draw` permet de dessiner une forme. Elle attend alors qu'on rentre le type de forme, et lance la fonction correspondante. Pour l'instant, la seule forme est `char`, qui lit deux entiers (x, y) et un caractère c, et lance `draw_char(x, y, c)`. (*On rappelle l'existence de la fonction* `int strcmp(char* s1, char* s2)` *qui permet entre autre de tester l'égalité entre deux chaînes de caractères.*

Nous allons maintenant implémenter une fonction qui permet de tracer une ligne. Le type de caractère utilisé dépendra de l'angle de la ligne à tracer. Si la ligne est plutôt verticale, on utilisera `|`, si elle est plutôt horizontale, on utilisera `-`, et si elle est plutôt diagonale, on utilisera `/` ou \. Plus précisément, en notant dx l'écart horizontal et dy l'écart vertical entre les deux extrémités de la ligne à tracer :

— Si les deux points sont confondus, i.e. si $dx = dy = 0$, on utilisera le caractère

— Si $dx = 0$, on utilisera `|`

— Si $|\frac{dy}{dx}| < \frac{1}{2}$, on utilisera `-`

— Si $\frac{dy}{dx} \in [\![\frac{1}{2}, 2]\!]$, on utilisera `/`

— Si $\frac{dy}{dx} \in [\![-2, -\frac{1}{2}]\!]$, on utilisera \

— Sinon (donc si $|\frac{dy}{dx}| > 2$), on utilisera `|`

Question 6. Écrire une fonction `char slope(int x1, int y1, int x2, int y2)` qui détermine le caractère à utiliser pour tracer une ligne entre (x_1, y_1) et (x_2, y_2).

Question 7. Écrire une fonction `void draw_line(int x1, int yi, int x2, int y2)` qui trace une ligne entre les deux points spécifiés. Commencez par écrire une fonction qui marche si la pente est faible (i.e. si $|\frac{dx}{dy}| \leq 1$) et si le premier point est à gauche du second, en itérant sur les valeurs de x entre x_1 et x_2. Lorsque cette dernière fonctionne bien, généralisez la fonction en essayant de ne pas écrire du code plus redondant que nécessaire.

Question 8. (Bonus) Réimplémentez la fonction précédente avec l'algorithme de Bresenham (cf DS) !

Question 9. Ajoutez à l'interface utilisateur la possibilité de tracer des lignes avec la fonction. Testez bien votre nouvelle fonctionnalité.

Question 10. Implémentez toutes les fonctionnalités que vous pouvez imaginer : des rectangles, des cercles, d'autres commandes que `draw` (par exemple de quoi copier-coller une partie de l'image, de quoi faire une rotation, un miroir ou toute autre transformation...).