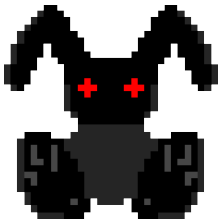

EPITECH.
L'ECOLE DE L'INNOVATION ET DE L'EXPERTISE INFORMATIQUE

Skill Test 3A

Affichage de texte

Sujet : Chatty Screen

Le Laboratoire aux Lapins Noirs
lapinsnoirs@epitech.eu

Ce document contient un sujet d'examen associé au module d'infographie

Nom du dépôt de rendu : chatty_promotion (Exemple chatty_2042)

Index

1 – Consignes

L'examen doit être réalisé avec la LibLapin comme unique outil. Les seules fonctions autorisées sont celles précisées dans la section sujet.

La compilation sera effectuée avec les flags -Wall -Wextra.

Si votre rendu comporte un fichier binaire, un fichier .o ou un fichier tampon type « #*# » ou « *~ », vous obtiendrez la note de 1,5.

Si votre votre fonction ne s'appelle pas de la bonne façon, le programme de correction ne pourra pas la trouver et vous obtiendrez la note de 1,5.

Si votre programme est trop lent (>2 secondes), boucle à l'infinie, reçoit un signal SIGSEV, SIGFPE ou SIGPIPE, vous obtiendrez la note de 1,5.

La réussite de la compilation de votre programme avec la moulinette vous apporte la note de 2. Cette note évolue en fonction des résultats obtenus aux exercices.

Votre rendu ne doit pas comporter de main. Nous compilerons l'intégralité des fichiers .c rendus avec la moulinette. Les fichiers .h seront pris en compte si situé à la racine de votre rendu.

2 – Sujet

Préliminaire :

1 points

```
void                    tekpixel(t_bunny_pixelarray     *pix,
                                 t_bunny_position       *pos,
                                 unsigned int           color) ;
```

Écrivez la fonction suivante, qui dessine un pixel de la couleur color à la position pos dans pix.

Préliminaire :

2 points

```
void        tekletter(t_bunny_pixelarray *pix,
                      char               c) ;
```

Écrivez la fonction suivante, qui configure la partie t_bunny_clipable de pix de manière à ce qu'un bunny_blit utilisant pix affiche le caractère c.

Pour connaître la position et le format à utiliser, référez vous au fichier font.png fournit avec ce sujet (et avec la LibLapin)

Vous pouvez ignorer les champs scale, origin, et rotation du t_bunny_clipable.

Écrivez la fonction suivante :

5 points

```
void            tektext(t_bunny_pixelarray      *out,
                        t_bunny_pixelarray      *fontpng,
                        const t_bunny_position  *pos,
                        const char              *str) ;
```

Ecrivez la fonction tektext qui affiche dans out, à la position pos, la chaine de caractère str à l'aide de la police font. La position pos marque le coin en haut à gauche du texte.

Les caractères doivent être espacés d'un pixel situé sur leur droite.

Vous devez gérer le saut de ligne. En cas de texte multi-ligne, vous devez évidemment revenir à la position de départ en X avant de continuer à écrire. Chaque ligne est séparée d'un pixel.

Vous n'avez pas à gérer tabulation.

Vous pouvez ignorer les champs scale, origin, et rotation du t_bunny_clipable.

N'hésitez pas a utiliser bunny_load_pixelarray pour tester.

Écrivez la fonction suivante :

5 points

```
void            tektitle(t_bunny_pixelarray     *out,
                         t_bunny_pixelarray     *fontpng,
                         const char             *str) ;
```

tektitle fonctionne de la même façon que textext à deux différences près :

- Le texte devra être centré dans out.

- Vous ne devez plus ignorer le champ scale dans t_bunny_clipable.

- Le pixel d'espacement doit lui aussi subir l'influence du champ scale.

Écrivez les fonctions suivante :

5 points

```
typedef struct          s_bunny_symbols
{
        size_t          length ;
        char            data[0] ;
}                       t_bunny_symbols ;

t_bunny_symbols         *teknew_symbol(const char       *str,
                                       size_t           len) ;
```

Cette fonction retourne un pointeur sur une structure de type t_bunny_symbols dont le champ length vaut len et dont data doit contenir les mêmes caractères que str.

Une utilisation astucieuse de bunny_malloc, auquel vous avez le droit vous permettra de réaliser cette fonction.

```
void                    tekdelete_symbol(t_bunny_symbols        *sym) ;
```

Cette fonction libère la mémoire utilisée par la structure dont l'adresse est passée en paramètre.

Vous avez le droit à bunny_free.

```
void                    teksymbols(t_bunny_pixelarray           *out,
                                   t_bunny_pixelarray           *fontpng,
                                   const t_bunny_position       *pos
                                   const t_bunny_symbols        *sym) ;
```

La fonction teksymbols affiche dans out, à la position pos les caractères sym avec la police font. Vous devez évidemment gérer l'attribut scale. Les caractères de fonctions sont désactivés : il s'agit uniquement de dessiner les glyphes.

Il ne doit pas y avoir d'espaces entre les glyphes, contrairement aux fonctions précédentes.

…n'oubliez pas de déclarer la structure dans votre fichier…