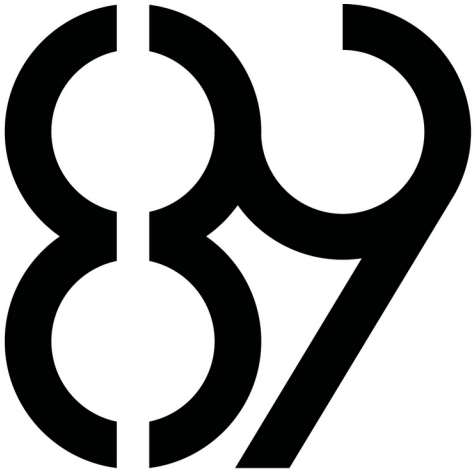


- Le Laboratoire aux Lapins Noirs -
lapinsnoirs@ecole-89.com

Ce mini-projet consiste à réaliser un logiciel d'affichage d'image au format vectoriel simple.

Nom de code : !vectrex
Clôture du ramassage : 19/01/2020 23:59

Médailles accessibles :

Définition des médailles à venir

INDEX

01 – Détails administratifs

Votre travail doit être envoyé via l'interface de ramassage du **TechnoCentre** :

http://technocentre.ecole89.com/ramassage

Le numéro de code présent sur ce sujet vous est propre : vous devrez le renseigner en rendant votre travail. En cas d'erreur, votre travail ne sera pas associée à l'activité et votre travail ne sera pas ramassé.

Pour cette activité, vous rendrez votre travail sous la forme d'une archive au format tar.gz. Cette archive devra contenir l'ensemble de votre travail tel que demandé dans la section 4.

Pour créer cette archive .tar.gz, il vous suffit d'utilliser la commande suivante :

```
$> tar cvfz mon_fichier.tar.gz fichier1 fichier2 fichier3
```

Le nom « mon_fichier.tar.gz » étant à remplacé par le nom que vous souhaitez donner votre fichier archive, et « fichier1 », « fichier2 », « fichier3 » par les fichiers ou dossiers que vous souhaitez mettre dans cette archive. Vous pouvez vérifier le contenu de votre archive à l'aide de la commande « **`tar -t mon_archive.tar.gz`** ».

Ce travail est a effectuer seul. Vous pouvez bien sur échanger avec vos camarades, néanmoins vous devez être l'auteur de votre travail. Utiliser le code d'un autre, c'est **tricher**. Et tricher annule **toutes** les médailles que vous avez reçu sur l'activité. La vérification de la triche est réalisée de la même manière que la correction : de manière **automatique**. Prenez garde si vous pensez pouvoir passer au travers.

Médailles accessibles :

Réussir à rendre :
Vous avez réussi à envoyer votre travail au système de correction.

02 – Propreté de votre rendu

Votre rendu, c'est à dire le contenu de l'archive ou du dépôt que vous entrez sur l'interface du TechnoCentre, doit respecter **strictement** l'ensemble des règles suivantes :

- Il ne doit contenir **aucun** fichier objet. (*.o)
- Il ne doit contenir **aucun** ichier tampon. (*~, #*#)
- Il ne doit pas contenir votre production finale (programme ou bibliothèque)

La présence d'un fichier interdit mettra immédiatement fin à votre évaluation.

Médailles accessibles :

Rendu propre
Votre rendu respecte les règles de propretés imposées.

03 – Règlement quant à la rédaction du code C

En général, le code source de votre programme doit impérativement respecter un ensemble de règles de mise en page définie par les Table de la Norme.

Pour cette activité, nous vous libérons de cette contrainte qui vous sera néanmoins très bientôt imposée pour l'ensemble de vos programmes écrits en C.

04 – Construction de votre rendu

Le programme de correction va construire une sous-partie déterminée de votre rendu afin d'effectuer des tests dessus. En voici les paramètres :

- Les fichiers qui seront compilés sont ceux qui auront l'extension *.c.
- Seuls les fichiers dans le(s) dossier(s) ./ seront compilés.
- Les fichiers seront compilés avec **-W -Wall -Werror**.

- Tous les fichiers seront compilés **ensemble**, cela signifie donc que chaque fonction doit être unique, et que vous pouvez utiliser les fonctions des autres exercices dans chaque exercice.

L'ensemble du code que vous rendez doit pouvoir être compilé.
En cas d'échec de la compilation, vous ne serez pas évalué.

Le nom de votre fichier exécutable sera vectrex.

Médailles accessibles :

Construction partielle
Les éléments requis de votre projet se construisent séparément.

05 – Fonctions interdites

Vous n'avez le droit à aucune autre fonction que celle précisée dans la liste ci-dessous :

Fonctions systèmes

- open
- close
- write
- read
- ioctl

Fonctions de la LibLapin

- bunny_start
- bunny_stop
- bunny_new_pixelarray
- bunny_delete_clipable
- bunny_save_picture
- bunny_blit

- bunny_set_*_function
- bunny_set_*_response
- bunny_loop

- bunny_malloc
- bunny_free

- bunny_load_ini
- bunny_ini_get_field
- bunny_delete_ini

- bunny_load_configuration
- bunny_configuration_getf
- bunny_delete_configuration

L'utilisation d'une fonction interdite est assimilée à de la triche.
La triche provoque l'arrêt de l'évaluation et la perte des médailles.

06 – VECTREX ?

La « VECTREX » est une console de jeu apparue en 1982. Elle a été conçue par Jay Smith et a la particularité d'être une console à affichage vectoriel.

Qu'est ce qu'un affichage vectoriel ? Une console plus habituelle dispose d'une mémoire ou chaque cellule contient la couleur d'un bloc de pixel ou d'un pixel seul. Cela signifie que pour faire apparaître une forme à l'écran, il faut écrire dans cette fameuse mémoire qui sera ensuite parcourue et affichée par l'écran.

Une console vectorielle ne dispose pas d'une mémoire utilisé comme tel. A la place, elle dispose d'une mémoire contenant des ordres de dessin. Par exemple : dessiner une ligne du point A au point B. Ainsi, au lieu d'avoir a remplir sa surface d'affichage a 100 % de couleurs déterminés, une console vectorielle demande a l'écran de tracer certains formes spécifiques.

Les opérations de remplissage de la mémoire dans une console classique étant extrêmement coûteuse d'un point de vue performance, la possibilité de s'en passer permettait aux développeurs de jeux des originalités comme la 3D.

Cette machine était tout à fait modeste si on la compare aux standards d'aujourd'hui : son microprocesseur était cadencé à 1,6 MHz et elle n'était doté que de 1Ko de RAM. C'est sur cette console que le premier casque de réalité virtuelle, conçu par John Ross, est apparu.

07 – Nature du projet

Le projet VECTREX consiste en la réalisation d'un afficheur de fichier d'un format particulier. Voici comme il s'utilise :

```
$> ./vectrex -h
Usage is :
     ./vectrex width height files+ -o output_file
```

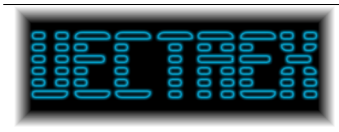
Le programme vectrex prend en paramètre la taille de la fenêtre à ouvrir, ainsi qu'un ou plus fichiers de configuration décrivant des formes à dessiner dans celle-ci.

Voici deux exemples de fichiers. Le premier est un format simple INI.

```
$> cat vectrex.ini
Shape="Dot"
Color=255, 0, 0
Coordinates=
10, 10,
20, 20
```

Ce fichier décrit une forme de type « Points ». Cela signifie que « Coordinates » contiendra des couples d'entiers, représentant des coordonnées X et Y qu'il faudra afficher. Les points seront de la couleur déterminé par « Color ». Le format de la couleur est « Color = Rouge, Vert, Bleu ».

Vous n'avez pas à effectuer le découpage du fichier de configuration vous-même dans ce projet- : Vous pouvez utiliser les fonctions des modules « bunny_ini » et « bunny_configuration » de la LibLapin.



Ci-dessous, un autre type de fichier au format plus complexe, DABSIC :

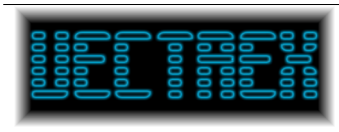
```
$> cat vectrex.dab
{Shapes
  [
    Type = "Dot"
    {Coordinates
      [
        Position = 10, 10
        Color = 255, 0, 0
      ],
      [
        Position = 30, 30
        Color = 255, 0, 0
      ],
    }
  ]
}

$>
```

Le champ « Shapes » est un tableau contenant ici une unique case. Cette première case contient un champ « Type » contenant la valeur « Dot », de la même manière qu'en INI. Un autre tableau, « Coordinates » est ensuite présent, contenant ici deux cases. Chacune de ces cases représentant un point. Les champs « Position » et « Color » donnent des informations concernant ces points.

Le champ « Shapes[].Coordinates[].Color » est optionnel si un champ « Shapes[].Color » est spécifié.

Le support du format INI par votre programme est obligatoire.
Le support du format DABSIC est recommandé.



08 – Lecture et affichage de points

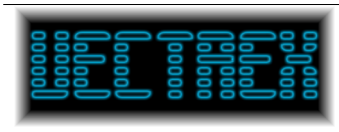
Votre premier travail va consister à parvenir à ouvrir une fenêtre à la bonne taille, ainsi qu'à charger un fichier de configuration et à l'afficher.

Les coordonnées indiquées dans le fichier de configuration sont indiqués par rapport au centre de la fenêtre et non par rapport au coin supérieur gauche !

Votre programme doit comporter la fonction suivante, qui sera évaluée séparément du logiciel vectrex lui-même :

```
void e89_vectrex_pixel(t_bunny_pixelarray    *px,
                       t_bunny_position      pos,
                       unsigned int          color)
```

Cette fonction dessinera dans **px**, à la position **pos** (par rapport au centre de px), un pixel de couleur **color**.



09 – Tracé de lignes

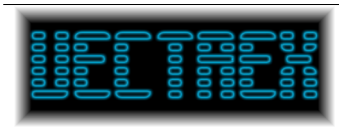
Votre second travail consistera à gérer un nouveau type de forme. Cette forme est « Line » et représente une ligne.

Lorsqu'un fichier emploi la forme ligne, les listes de coordonnées, au lieu de comporter seulement un couple X/Y en comporteront deux. Si un ensemble de coordonnées n'est pas complet (le nombre de couple est impair), l'ensemble incomplet est ignoré.

Votre programme doit comporter la fonction suivante, qui sera évaluée séparément du logiciel vectrex lui-même :

```
void e89_vectrex_line(t_bunny_pixelarray     *px,
                      t_bunny_position       *pos,
                      unsigned int           color)
```

Cette fonction dessinera dans px une ligne, depuis la position pos[0] (par rapport au centre de px) jusqu'à pos[1] (idem), de couleur color.



10 – Courbes

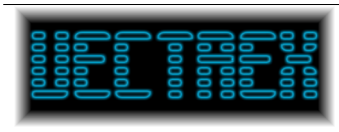
Cette partie est optionnelle. Vous allez à nouveau gérer un nouveau type de forme. Cette forme est **« Curve »** et représente une ligne courbe.

Lorsqu'un fichier emploi la forme courbe, les listes de coordonnées, au lieu de comporter seulement un couple X/Y en comportent 4. Si un ensemble de coordonnées n'est pas complet, l'ensemble incomplet est ignoré.

Votre programme doit comporter la fonction suivante, qui sera évaluée séparément du logiciel vectrex lui-même :

```
void e89_vectrex_curve(t_bunny_pixelarray    *px,
                       t_bunny_position      *pos,
                       unsigned int          color)
```

Cette fonction dessinera dans **px** une courbe, depuis la position **pos[0]** (par rapport au centre de px) jusqu'à **pos[3]** (idem), de couleur **color**. Les position **pos[1]** et **pos[2]** représentant des poids influençant la courbure de la courbe.



11 – Triangles

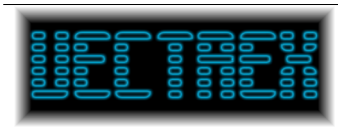
Cette partie est optionnelle. Vous allez à nouveau gérer un nouveau type de forme. Cette forme est **« Triangle »** et représente un triangle

Lorsqu'un fichier emploi la forme triangle, les listes de coordonnées, au lieu de comporter seulement un couple X/Y en comportent 3. Si un ensemble de coordonnées n'est pas complet, l'ensemble incomplet est ignoré.

Votre programme doit comporter la fonction suivante, qui sera évaluée séparément du logiciel vectrex lui-même :

```
void e89_vectrex_triangle(t_bunny_pixelarray      *px,
                          t_bunny_position        *pos,
                          unsigned int            color)
```

Cette fonction dessinera dans **px** un triangle dont les coins sont aux coordonnées **pos[0]**, **pos[1]** et **pos[2]**. L'ensemble du triangle sera rempli par la couleur **color**.



11 – Quadrilatères

Cette partie est optionnelle. Vous allez à nouveau gérer un nouveau type de forme. Cette forme est **« Quad »** et représente un quadrilatère.

Lorsqu'un fichier emploi la forme quadrilatère, les listes de coordonnées, au lieu de comporter seulement un couple X/Y en comportent 4. Si un ensemble de coordonnées n'est pas complet, l'ensemble incomplet est ignoré.

Votre programme doit comporter la fonction suivante, qui sera évaluée séparément du logiciel vectrex lui-même :

```
void e89_vectrex_quad(t_bunny_pixelarray          *px,
                      t_bunny_position            *pos,
                      unsigned int                color)
```

Cette fonction dessinera dans **px** un quadrilatère d'après les coordonnées situés dans **pos** et rempli de couleur **color**. Vous pouvez tout à fait utiliser **e89_vectrex_triangle** pour vous aider.