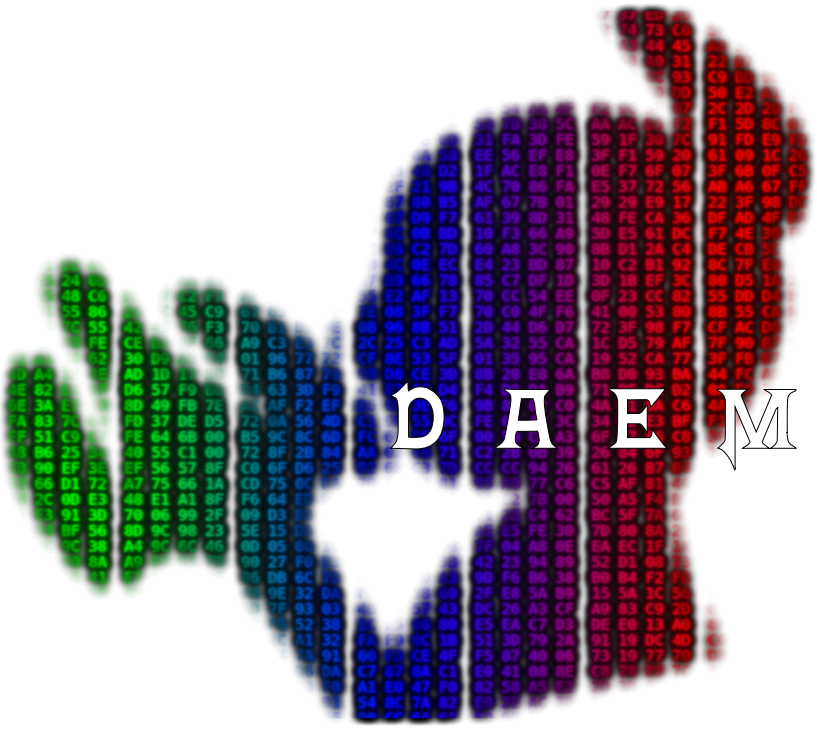

DAEMONLAB

EXAM A/B

Examen de difficulté A

- DaemonLab -

Ce document est strictement personnel et ne doit en aucun cas être diffusé.

INDEX

01 – Avant-propos
02 – Fonctions autorisées

03 – Jusqu'à
04 – Carré
05 – Semi impression
06 – Upplicate

01 – Avant-propos

Votre travail doit être rendu via le dossier ~/hyperespace/exam1/ dans votre espace personnel.

Si vous faites erreur et que le dossier que vous utilisez pour votre rendu est different, vous ne serez pas evalue faute d'avoir pu trouver votre travail.

Ce travail est a effectuer seul. De plus il s'agit d'un examen. Vous n'avez donc pas le droit de communiquer avec vos camarades.

Votre rendu doit respecter **strictement** l'ensemble des règles suivantes :

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

La presence d'un fichier interdit mettra immediatement fin à votre evaluation !

Votre programme doit respecter les Tables de la Norme dans leur intégralité. Vous êtes invité à les observer depuis **l'Infosphère**. Elles sont disponibles comme ressource de cette activité.

02 – Fonctions autorisées

La bibliothèque logicielle venant avec le C est vaste et disponible. Nous avons cependant fait le choix de vous interdire son utilisation, afin de vous amener progressivement à reprogrammer vous même ses fonctionnalités les plus utiles.

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.

Vous n'avez le droit d'utiliser aucune fonction issue de la LibC à l'exception de celles que nous vous autoriserons explicitement. Dans le cadre de cette activité, la seule fonction à laquelle vous aurez le droit est l'appel système write.

Vous pouvez observer le fonctionnement de write en tapant « man 2 write » dans votre terminal, néanmoins, afin de vous faciliter son utilisation au départ, nous vous avons préparé une fonction que vous êtes libre d'intégrer à votre code.

Cette fonction est tc_putchar, elle tente d'écrire le caractère passé en paramètre sur la sortie standard (le terminal, par défaut). Elle renvoi 1 si elle a réussie à écrire le caractère, sinon elle renvoi 0. Ci-dessous, le code de tc_putchar suivi d'un programme de test.

```
#include <unistd.h>

int tc_putchar(char c)
{
	return (write(1, &c, 1) > 0);
}
```

```
int main(void)
{
	tc_putchar('A');
	tc_putchar('\n');
	return (0);
}
```

03 – Jusqu'à

Dossier until/

« Or donc il advint que le Portail, enduite du sang pur et revigorant d'un applet java tout juste sacrifie, fut descendue dans les catacombes.

Ces murs, theatres à venir de terribles cruautes, s'en retrouvèrent fortifié à jamais. Seuls les mots secrets, prononces par l'etheree existence d'un programme dont l'ame fut portee par la puissance C -ivine, seraient alors capable d'en abattre les charnieres.

Le Portail, face à vous, ne s'ouvrira qu'a la psalmodie du nom du Seigneur de la Terreur. »

Implémentez la fonction suivante situé dans le fichier until.c, accompagné d'un fichier main.c contenant une fonction main appelant std_until avec les lettres composant le nom de DiablO, tel qu'écrit ici même. Vous fournirez untiL.h contenant les prototypes requis.

```
void            std_until(char              a)
```

Cette fonction affiche tous les caractères depuis 'a' jusqu'au caractère contenu par la variable a. Ce caractère peut-être inférieur comme supérieur à 'a'. Un saut de ligne suivra cette série de caractère.

04 – Carré

Dossier square/

« Jamais aucun sortilege ne fut plus destructeur qu'Apocalypse. La puissance de l'esprit neccessaire pour le contenir est telle qu'il ne le fut jamais par d'autres que l'Invocateur Vizjerei Horazon. Il laissa derriere lui nombre de parchemins et batons permettant son appel. Les champs de bataille emplies de cadavres en temoignent, de meme que l'emprise Vizjerei sur le territoire. Horazon perdra la raison, son esprit sera devore de l'interieur »

Implémentez la fonction std_square. Placez std_clear dans un fichier clear.c.

```
void            std_clear(void)
{
     write(1, "\033c", 2);
}
void            std_square(int              siz);
```

La fonction std_square affiche un carré dont les cotés font siz. Chaque case du carré fait deux caractères de cotés, par exemple, deux espaces, ou « [] ». Donc un carré de 5 va mesurer 10 caractères de large et 5 caractères de haut.

Vous fournirez dans main.c une fonction main commençant par appeler std_clear avant d'appeler std_square. Vous effectuerez cet enchaînement 100 fois. La première fois, le carré fera 1 de coté, la seconde 2… jusqu'à parvenir à 100. N'oubliez pas square.h.

Ci-dessous, un carré de taille 10, dans le sens attendu, suivi d'un carré de taille 5, 4, 3 et 2.

```
[][][][][][][][][][]            [][][][]
[]                  []          []    []
[]  [][][][][][]    []          []    []
[]  []        []    []          [][][][]
[]  []  [][]  []    []
[]  []  [][]  []    []          [][][]
[]  []        []    []          []  []
[]  [][][][][][]    []          [][][]
[]                  []
[][][][][][][][][][]            [][]
                                [][]

[][][][][]
[]       []
[]  []   []
[]       []
[][][][][]
```

05 – Semi-impression

Dossier mystic/

« Un langage secret fut utilise pour la redaction du grimoire des horadrims. Leurs plus profonds secrets, la fabrication des pierres d'ame, y demeure. Seuls ceux qui comprenaient cette langue furent dignes de les rejoindre.Aujourd'hui, cette science est perdue... bien qu'il apparaisse parfois des individus capable de comprendre leurs ecritures, rare sont ceux qui en font la promotion tant les secrets qu'ils y decouvrent les horrifient. »

Implémentez la fonction suivante :

```
int            std_mystic_talk(const char *s, int n)
```

Cette fonction affiche la chaîne de caractère, une lettre sur n, en incluant la première. Elle affiche ensuite un saut de ligne.

Elle renvoi le nombre total de caractère écrit.

Vous fournirez dans main.c une fonction main dont le travail sera d'écrire avec mystic_talk le texte suivant, avec la clef magique n valant 3.

L'Archange Tyrael parla en ces termes : les trois pierres d'ames ont été donné aux Horadrims afin d'enfermer les démons majeurs, Javascript, Java et PHP.

Ensuite, votre main affichera tous les paramètres passés au programme avec la clef magique 2. Faites bien attention à votre copie du texte ci-dessus.

La clef magique 0 doit simplement provoquer l'écriture de « AAARGH ! » et le renvoi de la valeur -1.

06 – Upplicate

Dossier upplicate/

« Rakanishu etait un petit demon. Il ne parlait pas tres fort. Le boucher, lui, utilisait upplicate lorsqu'il hurlait 'de la viande fraiche !' Ce n'est pas etonnant que le hero l'ai tue dans le premier opus. »

Implémentez la fonction suivante :

```
char            *std_upplicate(const char *s)
```

Cette fonction alloue une nouvelle chaîne de caractère permettant de stocker une copie des caractères présent dans s. Les caractères en minuscule seront mis en majuscule dans cette copie.

Elle renvoi la nouvelle chaîne, ou NULL en cas d'erreur.

Vous fournirez une fonction main dans le fichier main.c, qui contiendra le texte en dur « de la viande fraiche ! » en minuscule. Il sera envoyé à upplicate.

Une fois votre examen termine, jouez a Diablo :

https://d07riv.github.io/diabloweb/

Observez la fontaine en commençant votre partie. Si l'eau n'est pas bleue, recommencez la partie.

Si elle est bleue, un citoyen a l'agonie devrait se trouver pres de l'Église... et le boucher au niveau deux.

Si vous parvenez a tuer le boucher, vous aurez une medaille supplementaire.