



Piscine C

Rush 00

Résumé: Ce document est le sujet du rush 00 de la piscine C de 42.

Version: 5.2

Table des matières

| | | |
|-------------|---------------------------------|-----------|
| I | Consignes | 2 |
| II | Préambule | 4 |
| III | Sujet commun | 5 |
| IV | Rush 00 | 6 |
| V | Rush 01 | 8 |
| VI | Rush 02 | 9 |
| VII | Rush 03 | 10 |
| VIII | Rush 04 | 11 |
| IX | Rendu et peer-evaluation | 12 |

Chapitre I

Consignes

- Le groupe sera inscrit en soutenance automatiquement.
- N'annulez pas cette soutenance, vous n'en aurez pas une deuxième.
- Toute demande de précision sur un des sujets compliquera le sujet.
- Vous devez suivre la procédure de rendu pour tous vos exercices.
- Le sujet peut changer jusqu'à une heure avant le rendu.
- Les exercices sont très précisément ordonnés du plus simple au plus complexe. En aucun cas nous ne porterons attention ni ne prendrons en compte un exercice complexe si un exercice plus simple n'est pas parfaitement réussi.
- Votre programme doit compiler avec les flags -Wall -Wextra -Werror, en utilisant le compilateur `cc`.
- Si votre programme ne compile pas, vous aurez 0.
- Votre programme doit être codé à la Norme. Si vous avez des fichiers ou fonctions bonus, celles-ci seront incluses dans la vérification de la norme et vous aurez 0 au projet en cas de faute de norme.
- Les exercices du rush sont à réaliser par groupe de 2, 3 ou 4.
- Le numéro du rush imposé pour votre groupe suivra la règle suivante : rang alphabétique de la première lettre du login du team leader (de 1 à 26) modulo 5.
- Vous devrez donc réaliser le sujet indiqué avec les binômes imposés et vous présenter en soutenance à l'heure dite avec tous vos binômes.
- Lors de la soutenance, le projet devra être terminé. Les soutenances servent à présenter et à expliquer votre travail dans les moindres détails.
- Chaque membre du groupe devra parfaitement être au courant du travail réalisé, chacun des membres sera interrogé, la note du groupe étant basée sur les moins bonnes explications.
- Évidemment, vous devez tout faire pour prendre contact avec vos binômes : té-

léphone, mail, pigeon voyageur, séance de spiritisme, etc. Aucune excuse ne sera acceptée en ce qui concerne les problèmes de groupe.

- Si après avoir vraiment tout essayé un de vos binômes reste injoignable : réalisez votre rush on s'arrangera en soutenance. Même si c'est le chef de groupe : vous avez tous accès au dépôt.
- Vous pouvez, à titre optionnel, réaliser plusieurs sujets pour avoir un éventuel bonus.



Il faudra absolument avoir le sujet obligatoire réalisé parfaitement pour prétendre aux sujets bonus. Si votre bonus fonctionne, mais pas le sujet original, vous aurez 0.

Chapitre II

Préambule

Voici les paroles du générique de Minus et Cortex :

Minus : Dis Cortex, tu veux faire quoi cette nuit ?

Cortex : La même chose que chaque nuit, Minus : tenter de conquérir le monde !

C'est Minus et Cortex

C'est Cortex et Minus

L'une est plein d'astuce

L'autre un vrai nimbus

Deux souris diaboliques

Du génie génétique

Quelles canailles,

Ces p'tites souris cobayes, -bayes, -bayes, -bayes,

-bayes, -bayes, -bayes, -bayes, -bayes !

Dans leur tête elles projettent

Des plans sur la comète

Pour partir à la conquête

De toute la planète

C'est Cortex et Minus

C'est Minus et Cortex

Qui ont le réflexe

De vouloir sans complexes

Tendre une souricière

À la Terre toute entière

Quelles canailles,


Ces p'tites souris cobayes, -bayes, -bayes, -bayes,

-bayes, -bayes, -bayes, -bayes.

Plutôt que de conquérir le monde, vous allez vous employer à conquérir ce rush !

Chapitre III

Sujet commun

| | |
|---|---------------|
|  | Exercice : 00 |
| rush0X | |
| Dossier de rendu : <i>ex00/</i> | |
| Fichiers à rendre : <code>main.c</code> , <code>ft_putchar.c</code> , <code>rush0X.c</code> | |
| Fonctions Autorisées : <code>write</code> | |

- Les fichiers à rendre seront le `main.c`, un `ft_putchar.c` et votre `rush0X.c`, où 0X correspondra au numéro du rush. Par exemple, `rush00.c`.
- Exemple de `main.c` :

```
int      main()
{
    rush(5, 5);
    return (0);
}
```

- Vous devrez donc écrire la fonction `rush` prenant en paramètre deux variables de type entier nommées respectivement `x` et `y`.
- Votre fonction `rush` devra afficher à l'écran un rectangle de `x` caractères de largeur, et `y` caractères de hauteur.
- Votre fonction ne devrait jamais planter ou entrer dans une boucle infinie.
- Votre `main` sera modifié en soutenance pour pouvoir changer les paramètres de l'appel à la fonction `rush`. Par exemple, ce genre de chose sera testé :

```
int      main()
{
    rush(123, 42);
    return (0);
}
```

Chapitre IV

Rush 00

- `rush(5,3)` affichera ceci :

```
$>./a.out
o---o
|   |
o---o
$>
```

- `rush(5, 1)` ceci :

```
$>./a.out
o---o
$>
```

- `rush(1, 1)` ceci :

```
$>./a.out
o
$>
```

- `rush(1, 5)` ceci :

```
$>./a.out
o
|
|
|
o
$>
```

- `rush(4, 4)` ceci :

```
$> ./a.out  
o--o  
|  |  
|  |  
o--o  
$>
```


Chapitre V

Rush 01

- `rush(5,3)` affichera ceci :

```
$>./a.out
/***\
*   *
\***/
$>
```

- `rush(5, 1)` ceci :

```
$>./a.out
/***\
$>
```

- `rush(1, 1)` ceci :

```
$>./a.out
/
$>
```

- `rush(1, 5)` ceci :

```
$>./a.out
/
*
*
*
\
$>
```

- `rush(4, 4)` ceci :

```
$>./a.out
/**\
*  *
*  *
\**/
$>
```

Chapitre VI

Rush 02

- `rush(5,3)` affichera ceci :

```
$> ./a.out
ABBBA
B  B
CBBBC
$>
```

- `rush(5, 1)` ceci :

```
$> ./a.out
ABBBA
$>
```

- `rush(1, 1)` ceci :

```
$> ./a.out
A
$>
```

- `rush(1, 5)` ceci :

```
$> ./a.out
A
B
B
B
C
$>
```

- `rush(4, 4)` ceci :

```
$> ./a.out
ABBA
B  B
B  B
CBBBC
$>
```

Chapitre VII

Rush 03

- `rush(5,3)` affichera ceci :

```
$> ./a.out
ABBBC
B  B
ABBBC
$>
```

- `rush(5, 1)` ceci :

```
$> ./a.out
ABBBC
$>
```

- `rush(1, 1)` ceci :

```
$> ./a.out
A
$>
```

- `rush(1, 5)` ceci :

```
$> ./a.out
A
B
B
B
A
$>
```

- `rush(4, 4)` ceci :

```
$> ./a.out
ABBC
B  B
B  B
ABBC
$>
```

Chapitre VIII

Rush 04

- `rush(5,3)` affichera ceci :

```
$> ./a.out
ABBBC
B  B
CBBBA
$>
```

- `rush(5, 1)` ceci :

```
$> ./a.out
ABBBC
$>
```

- `rush(1, 1)` ceci :

```
$> ./a.out
A
$>
```

- `rush(1, 5)` ceci :

```
$> ./a.out
A
B
B
B
C
$>
```

- `rush(4, 4)` ceci :

```
$> ./a.out
ABBC
B  B
B  B
CBBA
$>
```

Chapitre IX

Rendu et peer-evaluation

Rendez votre travail sur votre dépôt `Git` comme d'habitude. Seul le travail présent sur votre dépôt sera évalué en soutenance. Vérifiez bien les noms de vos dossiers et de vos fichiers afin que ces derniers soient conformes aux demandes du sujet.

Vu que votre travail ne sera pas évalué par un programme, organisez vos fichiers comme bon vous semble du moment que vous rendez les fichiers obligatoires et respectez les consignes du sujet.



`Vous ne devez rendre uniquement les fichiers demandés par le sujet de ce projet.`