

Consignes de rendus:

Veillez lire attentivement et vérifiez à chaque rendu que vous respectez les consignes de rendus.
Il est de votre responsabilité de les vérifier.

Tout rendu ne vérifiant pas les consignes ne sera pas traité.
Tout rendu non traité et noté obtiendra la note de 0.

Noms du répertoire racine:

Votre répertoire contenant le projet doit se nommer:
nom1_nom2_projet_echec/

avec:

nom1: le 1er nom de votre binôme (uniquement le nom, pas le prénom).

nom2: le 2eme nom de votre binôme (uniquement le nom, pas le prénom).

Respectez l'ordre alphabétique afin de rester dans un ordre cohérent au fur et à mesure des dépôts.

Si vous êtes en monôme, omettez le nom2.

Noms de répertoires et de fichiers:

Dans vos noms, ainsi que dans tous noms de fichiers ou de répertoires, vous n'utiliserez que les caractères ASCII de base. C'est à dire les caractères situés entre a-z. L'underscore (_) est autorisé et permet de séparer les mots.

Notez qu'en particulier, sont interdits:

- Les accents
- Les espaces
- Les cédilles
- Les majuscules.

ex. de noms de répertoires ou de fichiers non conformes:

Umberto_Chala/

mon projet/

Audré_humbert/

à remplacer par:

umberto_chala/

mon_projet/

audre_humbert/

But: Rendre vos fichiers et répertoires **portables**.

Les caractères spéciaux, majuscules, et accents peuvent posséder des encodages différents suivant les systèmes (ex. Passage Windows/Linux). Les espaces sont excessivement lourds à traiter en ligne de commande.

Dépôts de fichiers:

Lors d'un upload sur le dépôt de fichier, vous déposerez de manière systématique une archive **tar.gz** du même nom que le répertoire archivé.

Dans votre cas, votre archive doit se nommer:

```
nom1_nom2_projet_echec.tar.gz
```

Une fois décompressée, cette archive doit donner le répertoire:

```
nom1_nom2_projet_echec/
```

Attention: Une archive .zip (ou .rar) renommée en .tar.gz n'est pas une archive tar.gz !
Ces archives ne seront pas traitées.

Rappel:

Pour archiver un repertoire <REP> en ligne de commande, on utilisera la syntaxe:

```
$ tar cvfz <REP>.tar.gz <REP>
```

Pour décompresser une archive NOM.tar.gz en ligne de commande, on utilisera la syntaxe:

```
$ tar xvfz NOM.tar.gz
```

Attention: Il est de votre responsabilité de vérifier l'intégrité de votre archive.

Une archive invalide ne sera pas traitée.

Pour vérifier votre archive, il est conseillé de décompresser celle-ci et de vérifier son contenu avant son envoi.

Organisation de vos répertoires:

Vos répertoires doivent présenter une structure claire et standard.

On organisera donc une structure en sous-repertoire tel que, au moment du rendu vous aillez :

un répertoire **src**/ (comme source) contenant uniquement les fichiers sources de votre projet.

un répertoire **bin**/ (comme binaire) contenant uniquement le (/ou les) fichier(s) binaire(s) executable(s) de votre projet

un répertoire **test**/ contenant uniquement les fichiers de tests du projet (voir séance sur les tests)

(optionnel : un répertoire **lib**/ (comme librairie) contenant uniquement les librairies de votre projet. (.so, .a))

Voici un exemple de schéma valide en début de projet:

```
nom1_nom2_projet_echec/src  /piece.h
                             piece.c
                             jeu.h
                             jeu.c
                             ...
                             /bin  /jeu_echec
                             /lib  /libecriture_fichier.a
                             /test /
```

Le répertoire **src/** contiendra uniquement les fichiers textes du code source. C'est à dire, les **.c**, les **.h**, potentiellement le Makefile et/ou les scripts de compilation.

Il ne doit pas contenir de fichiers temporaires, ou de binaires.

En particulier:

- Pas d'exécutables
- Pas de fichiers objets (.o)
- Pas de fichiers cachés (commençant par **.**, ou terminant par **~**) => faites View/Afficher_les_fichiers_cachés dans votre exploreur.

Lorsqu'il vous est demandé de déposer un fichier exécutable, vous placerez celui-ci dans un répertoire spécifique **bin/**

Si vous avez des bibliothèques binaires (.so ou .a), elles seront placés dans le répertoire **lib/**

Si vous avez des scripts de tests, ils seront placés dans le répertoire **test/**

Fichiers:

Tous vos fichiers (.c, .h, Makefile, tests, ...) doivent impérativement contenir vos noms, prénoms, et groupe suivant la syntaxe suivante:

```
// #Nom1: ZZZ
// #Prenom1: ZZZZ
// #Nom2: ZZZ
// #Prenom2: ZZZZ
// #Groupe: Z
```

(dans les tests et Makefile, les // seront omis et la ligne débutera par le caractère #).

Dans ces lignes, vous remplacerez les caractères Z, par les chiffres et numéros adéquates. Vous prendrez soin de respecter la syntaxe débutant par les symboles spécifiques // #.

N'oubliez pas que les accents, cédilles et espaces ne sont pas autorisés.

Compilation et execution:

Tous programme rendu doit compiler.

Tous programme ne compilant pas entraine la note de 0.

Si vous n'arrivez pas à corriger une erreur de compilation, **commentez** la partie provoquant l'erreur et ajoutez des commentaires de manière adéquate.

Vos programme ne doivent pas avoir d'erreurs mémoires.

Vous disposez des outils de debugs (gdb, Valgrind, ...) permettant de détecter les fuites mémoires.

Si un programme contient une erreur mémoire (souvent finissant par une erreur de segmentation, ou Seg Fault) que vous n'arrivez pas à régler, commentez la partie générant cette erreur et ajoutez des commentaires de manière adéquate.

Fraude et copie:

Toute fraude est interdite.

Toute copie sur un binôme voisin sans le citer, ou à partir de sources externes (livre, internet, ...) sans citation entraîne automatiquement l'invalidation de votre module et l'alerte de la direction des études.

Il est indispensable de citer explicitement dans votre code/compte-rendus/scripts toute copie/inspiration d'un binôme voisin ou de toute autre source externe.

Notez qu'il n'y a pas de taille minimale admissible de copie. Même la copie de 3 lignes de codes impose de citer vos sources.

Conclusion:

- Copier **en citant** est autorisé.
- Copier sans citer n'est pas autorisé.

Sauvegarde de vos données:

Vous êtes **responsables** de sauvegarder votre projet au fur et à mesure sur différents supports. Les supports aisément accessibles dans le monde informatique sont, par exemple:

- Des dépôts sur internet (site perso, dropbox, ...)
- Des clés USB
- Des CD/DVD
- Des disques durs externes
- Gestion de version (github, sourceforge, ...)

Notez que sur les PC de CPE vous n'avez pas accès au lecteur CD pour des questions de sécurité.

Attention: La perte du projet complet pour cause de crash d'un ordinateur personnel ne sera pas considérée comme une excuse autorisée.

Faites des sauvegardes régulières sur d'autres supports. Tout disque dur d'ordinateur est un support temporaire et est voué à être inutilisable au bout de quelques mois/années.

Respect des délais:

Les dates de rendus sont indiquées sur les dépôts du e-campus. Rendez à temps vos travaux.

Dans le cas d'une panne de serveur ou d'une incertitude, envoyez vos rendus par mails aux enseignants concernés (+incluez dans tous les cas en copie damien.rohmer@cpe.fr).

Rappels des mails:

damien.rohmer@cpe.fr

martine.breda@cpe.fr

nathael.pajani@cpe.fr

mohamed.sallami@cpe.fr

En cas de rendu en retard, une pénalité de 1 point par jour de retard sera mise en place.