Consignes de rendus.

Veuillez 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_siam/`

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.
Tous les caractères doivent être en **minuscules**.

Si vous êtes en monôme, omettez le `nom2`.

Noms de répertoires et de fichiers:

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 pour un répertoire ou un fichier.

Notez qu'en particulier, sont **interdits**:

- Les espaces
- Les accents
- 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 fastidieux à traiter en ligne de commande.

Dépots 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_siam.tar.gz`

Une fois décompressée en ligne de commande, cette archive doit donner le répertoire:
`nom1_nom2_projet_siam/`

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.

Prenez l'habitude après chaque rendu sur le dépôt de fichier de **télécharger ce que vous venez d'uploader et de vérifiez son contenu**.
Il arrive régulièrement que vous ne finissiez pas la démarche d'upload involontairement, ou que vous rendiez une archive incomplète ou non viable.
Vous êtes responsable de vérifier le contenu adéquate de ce que vous avez rendus au niveau du e-campus.

Si vous vous rendez compte d'un problème, ou si vous avez un doute, envoyez également une copie de l'archive correspondante par e-mail à damien.rohmer@cpe.fr. Il n'y a pas de pénalités associés tant que cela se passe avant la date de rendu de votre groupe.

Organisation de vos répertoires:

Vos répertoires doivent présenter une structure claire et standard.
Lisez l'annexe concernant l'organisation des répertoires pour d'avantages d'informations.

Remarque importante:
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 explorer.
- Ce répertoire doit contenir le fichier `nom.txt` complété de manière valide avec vos noms et groupe (voir annexe concernant la complétion du fichier nom).

Lorsqu'il vous est demandé de déposer un fichier exécutable, vous placerez celui-ci dans un répertoire spécifique **`bin/`**
Si vous avez des librairies binaires (.so ou .a), elles seront placés dans un répertoire `lib/`
Si vous avez des scripts de tests, ils seront placés dans le répertoire `test/`
Si vous avez un rapport, vous le placerez dans un répertoire `rapport/`

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.

Éliminez les Warnings au fur et à mesure. Vos rendus de projets ne doivent **pas présenter de Warnings** lors de la compilation (avec les options Wall et Wextra activés au minimum).

Vos programme **ne doivent pas avoir d'erreurs mémoires**.
Vous disposez des outils de debugs (`gdb, Valgrind, ...`) permettant de détecter les fuites mémoires et les tutoriaux sont réalisés dans le module.
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.
Veuillez vous référer à l'annexe sur le plagiat et les citations pour d'avantages de détails.

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 différent 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 sécurités.

Attention: La perte du projet complet pour cause de crash d'un ordinateur personnel ne sera pas considéré comme une excuse valable pour obtenir un délais de rendu plus long.
Faites des sauvegardes régulières sur d'autres supports. Tout disque dur d'ordinateur est un support temporaire et est voué 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:
timothe.bordiga@cpe.fr
martine.breda@cpe.fr
anthony.chomienne@cpe.fr
remi.marenco@cpe.fr
jonas.pauthier@cpe.fr
damien.rohmer@cpe.fr

En cas de rendu en retard (c'est à dire dès que l'on dépasse l'heure limite prévue du dépôt de votre groupe), une pénalité de 2 points sera mise en place. Si le retard dépasse 1 jour, la pénalité sera augmentée de 1 point par jour supplémentaire.

Notes de projet:

Vos rendus de projets seront notés suivant différents critères.

Plusieurs, voir l'intégralité des notes de résultats de vos rendus sera calculée sous la forme suivante: **coeff_consigne x note_resultat**,

Avec:

- **coeff_consigne**: un coefficient pondérateur entre 0 et 1 tel que coeff_consigne=1 si l'ensemble des consignes sont respectées, et décroit si une ou plusieurs consignes ne sont pas respectées.
- **note_resultat** correspond à une note sur 20 points que vous obtenez en fonction du bon fonctionnement de votre programme.

Cette note de résultat est calculé en réalisant différents tests de fonctionnement sur votre programme. Les tests suivront l'avancement des séances et pourront couvrir:

- Le bon fonctionnement lors de l'envoie de paramètres correctes.
- Le fonctionnement attendu lors de l'envoie de paramètres incorrectes (vérification des variables mises à jours, arrêts du programme dans certains cas, respect des contrats, etc).
- La présence de commentaires pertinents (description des contrats, algorithmes, explications de choix de codage, etc).

- L'absence d'erreurs mémoires. Les erreurs mémoires sont vérifiés automatiquement pour chaque test par Valgrind. Celui-ci doit indiquer aucune erreur. Si une ou plusieurs erreurs sont détectées le test ne rapportera qu'au maximum la moitié des points lui correspondant.
- La pertinences de vos tests (voir en particulier la séance spécifique sur les tests de l'application et l'annexe qui lui est associée: annexe_scripts_de_tests).

Rappel: Un rendu associé à un programme qui ne compile pas sur un PC de CPE sous Linux obtiendra la note de 0.

Les tests déjà effectués sur votre archive pour une séance donnée pourront parfois être relancés et notés à nouveau sur vos rendus suivants. Il est donc important de corriger vos erreurs au fur et à mesure.
Soyez particulièrement réactif d'un rendu d'une séance à l'autre afin d'éviter d'être pénalisés plusieurs fois de suite.

Le code de vos archives sera également évalué à une ou plusieurs reprises vis à vis de sa qualité en plus des notes de résultats. Notez que cette correction n'aura pas forcément lieu sur votre dernier rendu. Il pourra être réalisé sur n'importe quel rendu intermédiaire. Il est donc important de coder proprement tout au long de l'avancement de votre projet. Ne vous dites pas que vous commenterez et factoriserez votre code à la fin avant le dernier rendu.

A titre d'information, voici le barème prévisionnel associé à la lisibilité de vos codes (attention, il s'agit de critères prévisionnel pouvant être modifiés dans certains cas).

Lisibilité:

A: Excellente:
Lignes courtes et simples à lire.
Les parties répétitives sont factorisées dans des fonctions séparées.
Noms de variables appropriées.

B: Bonne lisibilité:
Peu ou pas de lignes complexes (présence de variables intermédiaires).
Bonne indentation.
Quelques passages pouvant être factorisé dans des fonctions.

C: Lisibilité passable:
Présence de longues lignes à certains endroits.
Indentation correcte ou approximative par moments.
Des répétitions qui pourraient être évitées.

D: Difficile à lire:
Lignes très longues sur chaque passage complexe.
Indentation approximative.
Trop de répétitions.

E: Illisible, ou pas indenté du tout.

Barème prévisionnel:

Attention, ce barème est prévisionnel. En fonction du déroulement du projet, ce barème pourra être modifié.

Note du projet:

- Note d'avancements de projet de séance en séance, environ 40%
- Note du rapport, environ 10%
- Note de l'état final du projet:
    - environ 30% pour le rendu minimal
    - environ 20% pour les travaux supplémentaires

Votre notre de projet pourra être adaptée de manière individuelle pour différencier les deux membres du binôme afin d'être le plus cohérent et juste en fonction du niveau et de l'implication de chacun (c-a-d les deux personnes du binôme n'auront pas forcément la même note finale).

(Rappel: Votre note de projet comptera pour 50% de la note de votre module, les 50% restants proviendront de la note individuelle d'examen écrit).

Les notes tiendront compte entre autre des différents points suivants:
- La clarté et lisibilité de votre code
- La bonne application des méthodologies de programmation: contrat, algorithmes, tests
- L'efficacité de vos algorithmes
- La précision de vos résultats et l'absence d'erreurs
- La précision et l’exhaustivité de vos tests
- Le respect des consignes de rendus.