Comment réaliser les exercices de ce cours ?

Pour apprendre la programmation, vous devez pratiquer !

Cependant, il ne faut pas pratiquer n'importe quoi, n'importe comment. On voit beaucoup de débutant qui souhaitent apprendre le C++ pour réaliser un projet. L'intention est bonne. Peut-être est-ce votre cas ?

Le problème dans ce cas est que l'on a souvent tendance à se focaliser sur le projet et pas sur les notions apprises. Si une notion vue dans le cours ne trouve pas d'application pratique dans le projet, elle ne sera pas correctement assimilée, voir complètement oubliée. Même les notions importantes.

Ce cours part d'une idée simple : **Si on ne pratique pas une notion, on ne la connaît pas.**

C'est pour cela que ce cours vous propose de nombreux exercices, travaux pratiques et projets. Si vous avez une idée de projet, je vous conseille d'accepter l'idée qu'il faut le mettre temporairement de côté, le temps d'avoir appris les bases du C++ et éviter de vous perdre dans votre apprentissage.

Les solutions aux exercices ne sont volontairement pas données. La raison est qu'il n'existe pas toujours “la bonne réponse”, mais plusieurs approches peuvent être correctes pour une même problématique. En proposant vos solutions sur les forums, vous pourrez obtenir des conseils détaillés sur les points à améliorer dans votre code.

Quelle est la différence entre exercices, travaux pratiques et projets dans ce cours ?

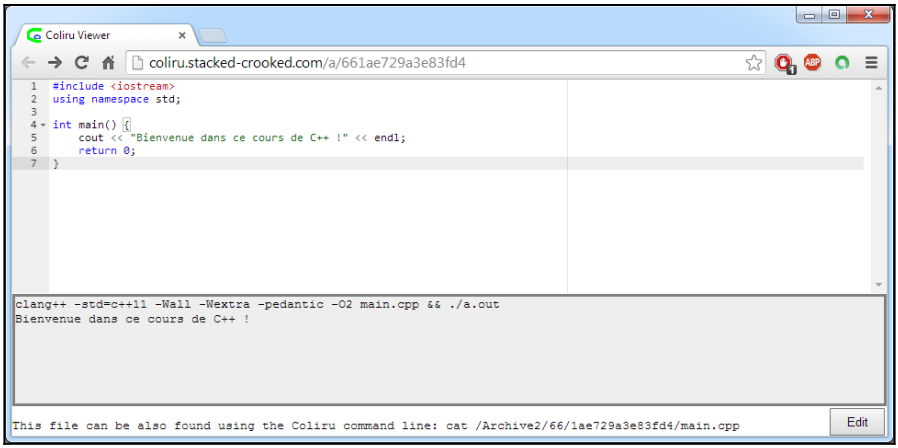
Les exercices

Chaque section d'un chapitre s'accompagne d'une série d'exercices, abordant la notion qui vient d'être expliquée. Je vous conseille de réaliser un maximum d'exercices. Les exercices sont rapides à réaliser (normalement quelques minutes) et ne nécessitent que quelques lignes de code.

Pour chaque exercice, un code est fourni. Vous devez modifier le code en suivant les instructions, le compiler et l'exécuter pour vérifier que le programme réalise la tâche demandée correctement. Si vous faites une erreur, le compilateur vous la signalera par un message d'erreur et il ne sera pas possible de lancer l'application. En cas de non respect des bonnes pratiques de codage, le compilateur signalera le problème avec un message d'avertissement. Le programme pourra quand même se lancer, mais sera susceptible d'avoir un comportement incorrect.

Dans ce cours, un exercice ne devra être considéré comme réalisé que lorsque vous n'aurez plus aucun message signalant un problème. (Ne vous inquiétez pas, vous apprendrez aussi à comprendre ces messages d'erreur).

Dans les chapitres suivants du cours, vous apprendrez à installer un environnement de développement complet pour compiler vos programme. Vous pourrez utiliser ces outils pour réaliser les exercices. Cependant, pour vous simplifier la tâche, vous pouvez également réaliser une majorité des exercices donnés sur un éditeur en ligne. Pour cela, un lien est donné pour chaque exercice, par exemple : Réaliser l'exercice. Cliquez dessus avec le bouton droit de la souris et choisissez “Ouvrir le lien dans un nouvel onglet” pour ouvrir l'éditeur en ligne Coliru :



Cliquez ensuite sur le bouton “Edit” en bas à droite pour modifier le code :

La fenêtre est divisée en trois parties :

- la partie du haut contient le code de l'exercice ;
- la partie du milieu (en gris) affiche les messages d'erreur du compilateur et les messages de l'application ;
- la partie du bas contient les commandes permettant de lancer la compilation, puis le programme.

Suivez les instructions données et modifiez le code dans la première partie pour réaliser l'exercice (dans ce premier exemple, vous n'avez rien à modifier). Cliquez ensuite sur le bouton “Compile, link and run...” pour lancer la compilation et l'exécution :

Dans cet exemple, le code permet d'afficher le message suivant “Bienvenue dans ce cours de C++ !”

Vous pouvez lancer la compilation autant de fois que vous le souhaitez. Si vous avez un doute sur la syntaxe à utiliser, le compilateur pourra vous indiquer le type d'erreur et vous aider à trouver la solution. Une erreur pouvant générer plusieurs messages, vous devez corriger les erreurs dans l'ordre donné par le compilateur.

Dans de nombreux exercices, vous ne devrez modifier que les parties du code indiquées par `@@`. Par exemple, un exercice pourra vous demander de remplacer par l'une des propositions données entre les `@@` :

```
// Choisissez la syntaxe correcte parmi les syntaxes proposées

int main() {
    @@int, double, string ou bool@@ i { 123 };
}
```

Il faut dans ce cas par exemple écrire :

```
// Choisissez la syntaxe correcte parmi les syntaxes
proposées

int main() {
    int i { 123 };
}
```

Dans d'autres cas, il faudra remplacer les `@@@@` par la syntaxe correcte :

```
// Choisissez la syntaxe correcte parmi les syntaxes
proposées

int main() {
    @@@@ i { 123 };
}
```

Dans d'autres cas, il faudra remplacer `@@@@@@` par une ou plusieurs lignes, sans modifier les autres lignes :

```
// Choisissez la syntaxe correcte parmi les syntaxes
proposées

int main() {
    @@@@@@
}
```

Les travaux pratiques

A la fin de chaque chapitre, une série de travaux pratiques vous est proposée. Ces travaux pratiques consistent généralement à écrire un programme simple de quelques dizaines à quelques centaines de lignes. Les instructions sont moins détaillées que pour les exercices et nécessiterons de votre part un travail de conception de votre programme.

Vous pouvez écrire vos programmes pour les travaux pratiques en utilisant Coliru et les partager en utilisant le bouton “Share!” en bas à droite pour obtenir une adresse URL que vous pouvez partager sur les

forums.

Vous pouvez également écrire vos programmes en utilisant les outils de développement que vous souhaitez. Dans ce cas, pour partager vos codes sources et obtenir les avis des autres, vous pouvez utiliser un site de gestion de version du code tel que GitHub.

Pour cela, vous devez créer un compte sur GitHub et créer un nouveau dépôt (*repository*) que vous pouvez par exemple nommer “cours-cpp”. Créez ensuite un répertoire pour chaque travail pratique et chaque projet.

utilisation détaillée de github et git windows et linux

Vous apprendrez dans la suite du cours à concevoir correctement vos programmes, mais voici déjà un conseil : n'hésitez pas à vous inspirer des codes C++ existants. Vous pouvez en particulier regarder comment sont conçues les bibliothèques comme la bibliothèque standard ou Boost.

Les projets

Le but des projets est de travailler sur un programme de taille relativement importante. Vous pourrez ajouter des fonctionnalités, corriger votre code et le faire évoluer pendant votre apprentissage de ce cours.

Plusieurs projets vous sont proposés. Choisissez en un ou deux (il est préférable de faire un projet correctement à fond, que de tenter de faire cinq projets et les faire à moitié) selon les thèmes que vous préférez.

Les projets nécessiteront de créer plusieurs fichiers, ce qui ne permet pas d'utiliser les éditeurs en ligne. Il faudra donc avoir installé les outils de développement nécessaires pour créer vos projets. Utilisez ensuite un outil de partage de code comme GitHub pour présenter vos projets sur les forums et avoir les avis des autres développeurs.

Les différents travaux pratiques et projets sont issues des thématiques suivantes. Je vous conseille d'essayer de vous focaliser sur une ou deux thématiques pour commencer (en particulier sur la thématique

“Généraliste”). Dès que vous aurez fini la lecture du cours et réalisé la majorité des exercices d'une thématique donnée, vous pourrez vous lancer dans les exercices d'une autre thématique.

Les thématiques sont les suivantes :

- Généraliste : cette thématique regroupe des algorithmes génériques, qui sont utilisés dans de nombreux cas d'applications.
- analyse de texte : dictionnaire, parser
- web : génération de pages html, xml, json, web application
- infographie : analyse d'image, rendering, raytracing, géométrie
- modélisation : créer un modèle, simulation
- crytologie : mot de passe, décodage/encodage
- analyse de données : statistique, data mining, machin learning
- traitement du signal
- Vision par ordinateur (*computer vision*)

Voir l'annexe “Liste des exercices par thématiques” pour plus de détails.

Chapitre précédent	Sommaire principal	Chapitre suivant

Cours, C++