

(https://profile.intra.42.fr)

SCALE FOR PROJECT CPP MODULE 07 (/PROJECTS/CPP-MODULE-07)

You should evaluate 1 student in this team



Git repository

git@vogsphere-v2.42.fr:vogsphere/intra-uuid-f50d27fb-ae30-4



Introduction

Merci de respecter les règles suivantes:

- Restez polis, courtois, respectueux et constructifs pendant le processus d'évaluation. Le bien-être de la communauté repose là-dessus.
- Identifiez avec la personne évaluée ou le groupe évalué les éventuels dysfonctionnements de son travail. Prenez le temps d'en discuter et débattre des problèmes identifiés.
- Vous devez prendre en compte qu'il peut y avoir de légères différences d'interprétation entre les instructions du projet, son scope et ses fonctionnalités. Gardez un esprit ouvert et notez de la manière la plus honnête possible. La pédagogie n'est valide que si la peer-évaluation est faite sérieusement.

Guidelines

- Ne notez que ce qui est contenu dans le dépôt Git cloné de l'étudiant(e) ou du groupe.
- Vérifiez que le dépôt Git appartient bien à l'étudiant(e) ou au groupe, que le projet est bien celui attendu, et que "git clone" est utilisé dans un dossier vide.
- Vérifiez scrupuleusement qu'aucun alias n'a été utilisé pour vous tromper et assurez-vous que vous évaluez bien le rendu officiel.
- Afin d'éviter toute surprise, vérifiez avec l'étudiant(e) ou le groupe les

potentiels scripts utilisés pour faciliter l'évaluation (par exemple, des scripts de tests ou d'automatisation).

- Si vous n'avez pas fait le projet que vous allez évaluer, vous devez lire le sujet en entier avant de commencer l'évaluation.


- Utilisez les flags disponibles pour signaler un rendu vide, un programme ne fonctionnant pas, une erreur de Norme, de la triche... Dans ces situations, l'évaluation est terminée et la note est 0, ou -42 en cas de triche. Cependant, à l'exception des cas de triche, vous êtes encouragé(e)s à continuer la discussion sur le travail rendu, même si ce dernier est incomplet. Ceci afin d'identifier les erreurs à ne pas reproduire dans le futur.

- Si le sujet requiert un fichier de configuration, vous ne devriez jamais avoir à le modifier. Si vous souhaitez éditer un fichier, prenez le temps d'expliquer pourquoi à la personne évaluée et de vous assurer que vous avez son accord.

- Vous devez aussi vérifier l'absence de fuites mémoire. Toute mémoire allouée sur le tas doit être libérée proprement avant la fin de l'exécution du programme.

Vous avez le droit d'utiliser tout outil disponible sur la machine tel que leaks, valgrind ou e_fence. En cas de fuites mémoire, cochez le flag approprié.

Attachments

 [subject.pdf \(https://cdn.intra.42.fr/pdf/pdf/41389/fr.subject.pdf\)](https://cdn.intra.42.fr/pdf/pdf/41389/fr.subject.pdf)

 [main.cpp \(/uploads/document/document/7055/main.cpp\)](/uploads/document/document/7055/main.cpp)

Tests préliminaires

Si un cas de triche est suspecté, la notation et l'évaluation prennent fin immédiatement. Pour le signaler, sélectionnez le flag "Cheat". Faites attention à l'utiliser avec calme, précaution et discernement.

Prérequis

Le code doit compiler avec c++ et les flags -Wall -Wextra -Werror
Pour rappel, ce projet doit suivre le standard C++98. Par conséquent, des fonctions C++11 (ou autre standard) et les containers ne sont PAS attendus.

Ne notez pas l'exercice si vous trouvez :

- Une fonction implémentée dans un fichier d'en-tête (sauf pour les fonctions templates).

- Un Makefile compilant sans les flags demandés et/ou avec autre chose que c++.

Sélectionnez le flag "Fonction interdite" (Forbidden function) si vous rencontrez :

- L'utilisation d'une fonction "C" (*alloc, *printf, free).
- L'utilisation d'une fonction interdite dans le projet.
- L'utilisation de "using namespace " ou du mot-clé "friend".
- L'utilisation d'une bibliothèque externe, ou de fonctionnalités propres aux versions postérieures à C++98.

☒ Yes

☐ No

Ex00 : Quelques fonctions pour commencer

L'objectif de cet exercice est d'écrire 3 fonction template simples : swap(), min() and max().

Types simples

Référez-vous au sujet pour vérifier la sortie attendue avec des types simples tels que int.

☒ Yes

☐ No

Types complexes

Est-ce que les fonctions marchent également avec des types complexes tels que :

```
class Awesome
{
public:
    Awesome(void) : _n(0) { }
    Awesome( int n ) : _n( n ) { }
    Awesome & operator= (Awesome & a) { _n = a._n; return *this; }
    bool operator==( Awesome const & rhs ) const { return (this->_n == rhs._n); }
    bool operator!=( Awesome const & rhs ) const { return (this->_n != rhs._n); }
    bool operator>( Awesome const & rhs ) const { return (this->_n > rhs._n); }
    bool operator<( Awesome const & rhs ) const { return (this->_n < rhs._n); }
    bool operator>=( Awesome const & rhs ) const { return (this->_n >= rhs._n); }
    bool operator<=( Awesome const & rhs ) const { return (this->_n <= rhs._n); }
    int get_n() const { return _n; }
private:
    int _n;
};

std::ostream & operator<<(std::ostream & o, const Awesome &a) { o << a.get_n(); return o; }

int main(void)
{
```

Awesome a(2), b(4);

```
swap(a, b);
std::cout << a << " " << b << std::endl;
std::cout << max(a, b) << std::endl;
std::cout << min(a, b) << std::endl;
return (0);
}
?
```

☒ Yes

☐ No

Ex01 : Iter

L'objectif de cet exercice est d'écrire une fonction générique d'itération dans un tableau.

Ça fonctionne ?

Utilisez le code suivant pour tester le rendu de l'étudiant(e) :

```
class Awesome
{
public:
    Awesome( void ) : _n( 42 ) { return; }
    int get( void ) const { return this->_n; }
private:
    int _n;
};

std::ostream & operator<<( std::ostream & o, Awesome const & rhs ) { o << rhs.get(); return o; }

template< typename T >
void print( T const & x ) { std::cout << x << std::endl; return; }

int main() {
    int tab[] = { 0, 1, 2, 3, 4 }; // <--- I never understood why you can't write int[] tab. Wouldn't that make more sense?
    Awesome tab2[5];

    iter( tab, 5, print );
    iter( tab2, 5, print );

    return 0;
}
```

Si tout fonctionne correctement, cela devrait afficher :

```
0
1
2
3
4
```

42
42
42
42
42

✓ Yes

✗ No

Ex02 : Array

L'objectif de cet exercice est d'écrire un template de classe qui se comporte comme un tableau. Si l'allocation interne du tableau ne provient pas d'un `new[]`, ne notez pas cet exercice. Demandez à la personne évaluée de prouver que son template marche avec des types simples et complexes avant de noter l'exercice.

Constructeurs

Est-il possible de créer un tableau vide et un tableau de taille spécifique ?

✓ Yes

✗ No

Accès

Les éléments doivent être accessibles en lecture et écriture grâce à l'opérateur `[]` (ou lecture uniquement si l'instance est constante). L'accès à un élément hors-limites doit jeter une `std::exception`.

✓ Yes

✗ No

Ratings

Don't forget to check the flag corresponding to the defense

✓ Ok

★ Outstanding project

📄 Empty work

💬 No author file

⚙️ Invalid compilation

📖 Norme

📄 Cheat

💥 Crash

💧 Leaks

🚫 Forbidden function

Conclusion

Leave a comment on this evaluation

Finish evaluation

Privacy policy (<https://signin.intra.42.fr/legal/terms/5>)

Terms of use for video surveillance (<https://signin.intra.42.fr/legal/terms/1>)

Declaration on the use of cookies (<https://signin.intra.42.fr/legal/terms/2>)

General term of use of the site (<https://signin.intra.42.fr/legal/terms/6>)

Legal notices (<https://signin.intra.42.fr/legal/terms/3>)

Règlement Intérieur (<https://signin.intra.42.fr/legal/terms/7>)