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

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

# ШКАЛА ДЛЯ ПРОЕКТА CPP MODULE 01 (/ПРОЕКТЫ/CPP-МОДУЛЬ-01)

Вы должны оценить 1 студента в этой команде

Git-репозиторий

git@vogsphere.msk.21-school.ru:vogsphere/intra-uuid-d5bec79

## Введение

Пожалуйста, соблюдайте следующие правила:

- Оставайтесь вежливым, учтивым, уважительным и конструктивным на протяжении всего процесса оценки. От этого зависит благополучие общества.

- Определите вместе со студентом или группой, чья работа оценивается, возможные недостатки в их проекте. Найдите время, чтобы обсудить и обсудить проблемы, которые, возможно, были выявлены.

- Вы должны учитывать, что могут быть некоторые различия в том, как ваши коллеги могли понять инструкции проекта и объем его функций. Всегда будьте непредвзяты и оценивайте их как можно честнее. Педагогика полезна только и только в том случае, если экспертная оценка проводится серьезно.

Методические рекомендации

- Оценивайте только ту работу, которая была сдана в репозиторий Git оцениваемого учащегося или группы.

- Дважды проверьте, что репозиторий Git принадлежит учащимся. Убедитесь, что проект соответствует ожиданиям. Также убедитесь, что «git clone» используется в пустой папке.

- Внимательно проверьте, чтобы не использовались вредоносные псевдонимы, чтобы обмануть вас и заставить оценить что-то, что не является содержимым официального репозитория.

- Чтобы избежать каких-либо неожиданностей и, если применимо, просмотрите вместе все используемые сценарии.

для облегчения выставления оценок (скрипты для тестирования или автоматизации).

- Если вы не выполнили задание, которое собираетесь оценивать, вы должны прочитать всю тему до начала процесса оценивания.

- Используйте доступные флаги, чтобы сообщить о пустом репозитории, неработающей программе, ошибке Norm, читерстве и так далее.
В этих случаях процесс оценки завершается и окончательная оценка равна 0 или -42 в случае списывания. Однако, за исключением списывания, учащимся настоятельно рекомендуется вместе просмотреть сданную работу, чтобы выявить ошибки, которые не следует повторять в будущем.

- Вам никогда не придется редактировать какой-либо файл, кроме файла конфигурации, если он существует. Если вы хотите отредактировать файл, найдите время, чтобы объяснить причины оцениваемому учащемуся и убедитесь, что вы оба согласны с этим.

- Также необходимо проверить отсутствие утечек памяти. Любая память, выделенная в куче, должна быть должным образом освобождена до окончания выполнения.
Вам разрешено использовать любой из различных инструментов, доступных на компьютере, таких как утечки, valgrind или e_fence. В случае утечек памяти отметьте соответствующий флаг.

# Вложения

- subject.pdf (https://cdn.intra.42.fr/pdf/pdf/40222/en.subject.pdf)

## Предварительные испытания

Если подозревается мошенничество, оценка здесь останавливается. Используйте флаг «Обман», чтобы сообщить об этом. Принимайте это решение спокойно, взвешенно и, пожалуйста, используйте эту кнопку с осторожностью.

Предпосылки

Код должен компилироваться с помощью C++ и флагов -Wall -Wextra -Werror Не забывайте, что этот проект должен соответствовать стандарту C++98. Таким образом, функции или контейнеры C++ 11 (и более поздних версий) НЕ ожидаются.

Любое из этих значений означает, что вы не должны оценивать рассматриваемое упражнение: - Функция реализована в заголовочном файле (кроме шаблонных функций).
- Makefile компилируется без необходимых флагов и/или другого компилятора, кроме c++.

Любое из этих действий означает, что вы должны пометить проект как «Запрещено».
Функция":

- Использование функции "C" (*alloc, *printf, бесплатно).
- Использование функции, не разрешенной в инструкции по упражнению.

- Использование «используя пространство имен " или ключевое слово "друг".

- Использование внешней библиотеки или функций из версий, отличных от C++98.

 Да  Нет

---

# Ex00: Браиииииииннннззз3

Цель этого упражнения — понять, как распределять память в C++.

---

Makefile и тесты

Существует Makefile, который компилируется с использованием соответствующих флагов.

Существует по крайней мере основной для проверки упражнения.

 Да  Нет

---

Класс зомби

Есть класс зомби.

Он имеет атрибут частного имени.

У него есть как минимум конструктор.

У него есть функция-член declare(void), которая печатает: ": BraiiiiiiinnnzzzZ..."

Деструктор печатает отладочное сообщение, включающее имя зомби.

 Да  Нет

---

новыйЗомби

Существует функция newZombie(), имеющая прототип: [ Zombie* newZombie( std::string name ); ]

Он должен выделить Zombie в куче и вернуть его.

В идеале он должен вызвать конструктор, который принимает строку и инициализирует имя.

Упражнение должно быть помечено как правильное, если Зомби может заявить о себе.

с именем, переданным функции.

Есть тесты, чтобы доказать, что все работает.

Зомби корректно удаляется до конца программы.

 Да  Нет

---

случайныйЧумп

Существует функция randomChump(), имеющая прототип: [ void randomChump( std::string name ); ]

Он должен создать зомби в стеке и заставить его объявить о себе.

В идеале зомби должен быть размещен в стеке (поэтому неявно удаляется в конце функции). Его также можно выделить в куче, а затем

явно удалено.

Студент должен обосновать свой выбор.

Есть тесты, чтобы доказать, что все работает.

 Да  Нет

---

# Ex01: Moar мозг!

Цель этого упражнения — выделить несколько объектов одновременно с помощью new[], инициализировать их и правильно удалить.

---

Makefile и тесты

Существует Makefile, который компилируется с использованием соответствующих флагов.

Существует по крайней мере основной для проверки упражнения.

 Да  Нет

---

зомбиОрда

Класс Zombie имеет конструктор по умолчанию.

Существует функция зомбиHorde(), прототип которой выглядит следующим образом: [ Zombie* zombieHorde( int N, std::string name ); ]

Он выделяет N зомби в куче, явно используя new[].

После выделения происходит инициализация объектов для установки их имени.

Он возвращает указатель на первого зомби.

В основном достаточно тестов, чтобы подтвердить предыдущие пункты.

Пример: вызов ANOME() для всех зомби.

Наконец, все зомби должны быть удалены одновременно в файле main.

 Да  Нет

---

# Ex02: ПРИВЕТ, ЭТО МОЗГ

Демистифицируйте ссылки! Демистифицируйте ссылки! Демистифицируйте ссылки! Демистифицируйте ссылки! Демистифицируйте ссылки!

Демистифицируйте ссылки! Демистифицируйте ссылки! Демистифицируйте ссылки! Демистифицируйте ссылки! Демистифицируйте ссылки!

Демистифицируйте ссылки! Демистифицируйте ссылки!

---

Makefile и тесты

Существует Makefile, который компилируется с использованием соответствующих флагов.

Существует по крайней мере основной для проверки упражнения.

 Да  Нет

ПРИВЕТ ЭТО МОЗГ

Есть строка, содержащая «ПРИВЕТ, ЭТО МОЗГ».
stringPTR — указатель на строку. stringREF — это ссылка
на строку.
Адрес строки отображается с помощью строковой переменной, stringPTR и
stringREF.
Строка отображается с использованием stringPTR и stringREF.

 Да  Нет

# Ex03: Ненужное насилие

Цель этого упражнения — понять, что указатели и ссылки имеют некоторые небольшие различия, которые делают их более подходящими в зависимости от использования и жизненного цикла используемого объекта.

Makefile и тесты

Существует Makefile, который компилируется с использованием соответствующих флагов.
Существует по крайней мере основной для проверки упражнения.

 Да  Нет

Оружие

Существует класс Weapon, который имеет строку типа, getType() и setType().
Функция getType() возвращает константную ссылку на строку типа.

 Да  Нет

ЧеловекА и ЧеловекБ

HumanA может иметь ссылку или указатель на оружие.
В идеале его следует реализовать как эталон, поскольку Оружие существует с
момента создания до уничтожения и никогда не меняется.
HumanB должен иметь указатель на оружие, так как поле не установлено во
время создания, а оружие может иметь значение NULL.

 Да  Нет

# Ex04: Сед для неудачников

Благодаря этому упражнению студент должен был познакомиться с ifstream и ofstream.

Makefile и тесты

Существует Makefile, который компилируется с использованием соответствующих флагов.

Существует по крайней мере основной для проверки упражнения.

 Да  Нет

ex04

Есть функция replace (или другое название), которая работает как указано в теме.

Управление ошибками эффективно: попробуйте передать несуществующий файл,
измените разрешения, передайте его пустым и т. д.

Если вы можете найти ошибку, которая не обрабатывается и не является полностью эзотерической,
то это упражнение не дает очков.

Программа должна читать из файла, используя ifstream или эквивалент, и
записывать, используя ofstream или эквивалент.

Реализация функции должна производиться с помощью функций из std::string, а не
путем посимвольного чтения строки.

Это уже не Си!

 Да  Нет

# ex05: Карен 2.0

Цель этого упражнения — использовать указатели на функции-члены класса. Кроме того, это возможность открыть для себя различные уровни журнала.

Makefile и тесты

Существует Makefile, который компилируется с использованием соответствующих флагов.

Существует по крайней мере основной для проверки упражнения.

 Да  Нет

Наша любимая Карен

Существует класс Карен, по крайней мере, с 5 функциями, необходимыми в предмете.

Функция пожаловаться() выполняет другие функции, используя указатель на них.

В идеале учащийся должен реализовать способ сопоставления различных строк,
соответствующих уровню журнала, с указателями соответствующей функции-члена.

Если реализация отличается, но упражнение работает, вы должны отметить его как
правильное. Единственное, что не разрешено, это иметь if/elseif/else.

Учащийся мог выбрать изменить сообщение, отображаемое Карен, или отобразить
примеры, приведенные в теме, и то, и другое допустимо.

 Да  Нет

# Ex06: Карен-фильтр

Теперь, когда вы стали опытным программистом, вам следует использовать новые типы инструкций, операторы, циклы и т. д. Цель этого

Последнее упражнение состоит в том, чтобы заставить вас обнаружить оператор switch.

Makefile и тесты

Существует Makefile, который компилируется с использованием соответствующих флагов.

Существует по крайней мере основной для проверки упражнения.

 Да  Нет

Выключение Карен

Программа karenFilter принимает в качестве аргумента любой из уровней журнала ("DEBUG",
«ИНФО», «ПРЕДУПРЕЖДЕНИЕ» или «ОШИБКА»). Затем он должен отображать только сообщения
которые находятся на том же уровне или выше (DEBUG < INFO < WARNING < ERROR). Этот
должен быть реализован с помощью оператора switch с регистром по умолчанию.
Еще раз, пожалуйста, больше никаких if/elseif/else.

 Да  Нет

# Рейтинги

Не забудьте проверить флаг, соответствующий защите

 Хорошо

 Пустая работа  Незавершенная работа W Недопустимая компиляция  Чит г Сбой

л Запрещенная функция

# Заключение

Оставьте комментарий к этой оценке

Завершить оценку