

Mi registro técnico de liquidación de deudas

Programación (26)

diario (4)

pensamientos (8)

libro (3)

hogar

programación

[42Seoul] Módulo 07 del CPP - Plantilla

Aggrodonk

2022. 9. 11. 18:28

Introducción

success

100

Solo – about 7 hours – 0 XP

subject.pdf

main.cpp

Ratey

youjeon's CPP Module 07

youjeon's group

This team is locked 4 days ago and closed 4 days ago

GIT REPOSITORY

git@vogsphere.42seoul.kr:vogsphere/intra-uuid-36c43727-0727-423e-9492-6465ab30478a-4389822-youjeon

EVALUATIONS

Peer evaluations (2/2)

EVALUATED BY DKIM2, KHONVOUM 4 DAYS AGO

template에 관련된 과제였는데, template과 연산자 오버로딩에 대해서 잘 설명해주셨고 []연산자에 const 오버로딩도 잘 구현하신 것 같습니다. 고생하셨습니다.

YOUR FEEDBACK, 4 DAYS AGO :

오랜만에 cpp 통과하신 평가자를 만났습니다. 평가자께서 평가표를 보며 직접 코드를 확인하시면서 중요하고 궁금하신 부분만 물어보시고 그거에 대답하는 형태로 진행을 하였는데, 보통은 이런식으로 평가를 하면 평가가 상대적으로 빨리 끝나지만 과제의 평가지에 주어진 코드가 깨진 코드들이 많아서 생각보다 오래걸렸던것 같습니다. 평가 감사드립니다!

4 / 4

4 / 4

4 / 4

4 / 4

EVALUATED BY YUJEE 4 DAYS AGO

씨름들 과제를 아직 시작하지 않아서 간단한 개념부터 설명을 시작해주셨습니다! template typename d을 사용하는 방법이나 클래스 내에서 연산자를 정의해주는 방법을 설명 해주셔서 과제를 시작할 때 도움이 될 것 같아요. 또 클래스를 생성할 때 0으로 초기화가 되었다는 것을 알려주셔서 저는 따로 초기화를 안하고 사용할 수 있을 것 같네요 하하 예시를 이용해서 코드를 함께 검증해봤고 기대했던 값이 잘 나오는 것을 확인했습니다! 하루종일 평가 받으시느라 정말 수고 많으셨습니다! 화이팅하세요!~!~!~!

YOUR FEEDBACK, 4 DAYS AGO :

아직 cpp를 시작도 안하시고 평가도 몇번 못해봐다고 하셔서 이 과제에서 요구하는 내용과 다른 과제를 진행할때도 문제가 생길수있는 내용을 위주로 더 상세하게 설명을 했는데, 잘 전달이 되고 이후에 과제하실때에 도움이 되었으면 좋겠습니다. 과제에 대한 내용을 설명한 이후에 평가표를 보면서 진행하였습니다. 평가 감사드립니다!

4 / 4

4 / 4

4 / 4

4 / 4

Recuerdo que me sentí avergonzado porque el código de prueba que figuraba en la hoja de evaluación estaba roto.

La tarea es implementar plantillas de funciones y plantillas de clases.

Siempre que comprenda la plantilla, la implementación no es una tarea muy difícil.

<https://modoocode.com/219>

	<h2>Masticar C++ - <9 - 1. Marco para imprimi...</h2> <p>modoocode.com</p>
--	-------------------------------------------------------------------------------

https://www.hanbit.co.kr/store/books/look.php?p_code=E6410226806

	<h2>Pensando en: programación STL en C++</h2> <p>La explicación se centró en ayudar a las personas que conocen los conceptos básicos de C++ pero no tienen experiencia con STL a co...</p> <p>www.hanbit.co.kr</p>
--	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

ex00

Cree la siguiente plantilla de función

- swap: recibe dos argumentos e intercambia sus valores. no devuelve nada
- min: recibe dos argumentos y devuelve el menor de los dos. Si son iguales devuelve el segundo.
- max: recibe dos argumentos y devuelve el mayor de los dos. Si son iguales devuelve el segundo.

Las tres plantillas deben funcionar sin importar qué tipo se utilice, pero los tipos de los dos argumentos son los mismos y admiten operadores de comparación.

Las plantillas deben definirse en un archivo de encabezado.

```
template<typename T>
void swap(T &a, T &b)
{
    T tmp = a;
    a = b;
    b = tmp;
}

template<typename T>
T min(T &a, T &b)
{
    if (a < b)
        return (a);
    else
        return (b);
}

template<typename T>
T max(T &a, T &b)
{
    if (a > b)
        return (a);
    else
        return (b);
}
```

Una "plantilla de función" es una plantilla. En otras palabras, se refiere a una muestra que imprime una función específica. Sin esta función habría que crear decenas de funciones llamadas swap para recibir todo tipo de argumentos, pero gracias a esta función es como si se crearan decenas de funciones con una sola implementación. La función creada de esta manera se denomina "función de plantilla".

Otra ventaja de esta función es que no solo el tipo de datos sino también cualquier clase se puede usar como argumento siempre que admita la operación utilizada dentro de él ($= > <$ en la asignación). Lo entenderá al ver que las funciones de conversión que ya ha utilizado funcionan sin importar el tipo de datos o la clase que venga.

ex01

Cree una plantilla de función que tome tres argumentos y no devuelva nada.

- El primer parámetro es la dirección de la matriz.
- El segundo parámetro es la longitud de la matriz.
- El tercer parámetro es una función que se llamará para cada elemento de la matriz.

Por supuesto, la tercera función podría ser una función de plantilla.

```

template<typename T>
void iter (T *array, size_t length, void (*fn)(T&))
{
    for (size_t i = 0; i < length; i++)
    {
        fn(array[i]);
    }
}

template<typename T>
void print(T &str)
{
    std::cout << str << " ";
}

template<typename T>
void sqrt(T &num)
{
    num = num * num;
}

```

Cree una plantilla de función de la misma manera que 00, pero haga que acepte un puntero de función como argumento.

Hay muchas condiciones y puede parecer difícil, pero no lo es si entiendes el modelo. Al principio, estaba un poco confundido acerca de tener que poder recibir un puntero de función como argumento como función de plantilla, pero después de ver que funcionaba incluso si lo hacía igual, lo implementé de inmediato.

ex02

Debe continuar con las pruebas agregando sus propias pruebas al main.cpp que se proporciona en el tema.

El main contiene varias pruebas usando la plantilla de clase Array con argumentos de tipo T. Los elementos que se deben implementar en la clase son los siguientes.

- Constructor sin argumentos: crea una matriz vacía.
- Constructor que recibe unsigned int n como argumento: cree una matriz de tamaño n e inicialícela (consejo: `int * a = new int();` y luego imprima `*a`
- Operador de copia y operador de asignación. Por supuesto, después de copiar ambos, no deberían ocurrir problemas en la otra matriz.
- Asegúrese de utilizar el nuevo `[]` al realizar asignaciones. Está prohibida la preasignación de memoria y no se debe acceder a la memoria no asignada.

- Se puede acceder a los elementos a través del operador [] y se debe generar una excepción std::si están fuera del rango.
- La función miembro size() devuelve el número de elementos de la matriz. No requiere argumentos y no debe modificar la instancia.

```

#ifndef ARRAY_HPP
# define ARRAY_HPP

# include <iostream>
# include <stdexcept>

template <typename T>
class Array {
private:
    std::size_t len;
    T *array;
public:
    Array(void);
    Array(std::size_t n);
    Array(const Array& obj);
    Array& operator=(const Array& obj);
    ~Array(void);
    std::size_t size(void) const;
    T& operator[] (std::size_t i);
    const T& operator[] (std::size_t i) const;
};

# include "Array.hpp"

#endif

```

Mientras traducía, vi por primera vez una parte en la que no entendía lo que significaba: `int * a = new int();` devuelve 0. Probablemente esto signifique que se inicializa automáticamente a 0.

El operador de copia y el operador de asignación parecen indicar que se debe utilizar una copia profunda, y la función de tamaño es `size_t size(void);` Creo que bastaría con declararlo como .

Dice que puede incluir o no `Array.hpp` en el archivo de envío. La razón por la que puede tener o no este archivo es porque la definición y la declaración del archivo de plantilla deben ocurrir simultáneamente en el encabezado.

En general, una clase contiene solo definiciones en el encabezado y la declaración y la implementación se realizan en un archivo `cpp`, pero dado que una clase de plantilla es una muestra para imitar una clase y no es una clase, se produce un error de enlace ya que no se puede trabajar solo con Esa información.

Para solucionar esto, existe una manera de implementar la función de clase dentro de un archivo de encabezado. Dado que esta es la única forma de hacerlo, la regla general estipula que las plantillas se pueden declarar en el encabezado: cualquier implementación de función colocada en un archivo de encabezado **(excepto las plantillas de funciones) significa 0 para el ejercicio.**

Un archivo tpp lo utilizan personas que no se sienten cómodas con tener la implementación y la declaración en el mismo lugar. Simplemente impleméntelo en el tpp donde creó la parte de declaración y escriba `#include "Array.tpp"` al final del archivo de encabezado. De hecho, esto también tiene la misma función implementada en el encabezado, pero si crees que debería estar separado, deberías implementarlo así.

No es muy diferente de lo que se implementó anteriormente. Sin embargo, pensé que sería mejor implementarlo por separado en lugar de implementarlo en el encabezado, así que lo intenté, pero tuve algunos problemas con la codificación porque el autocompletado o la verificación de errores del IDE no funcionaban correctamente en el archivo tpp.

Aparte de eso, no creo que sea muy diferente a crear una clase existente. Sin embargo, al implementar el operador `[]`, también se debe implementar `const`. De lo contrario, puede suceder que el operador predeterminado no esté cargado y se pueda generar un error.

[https://cplusplus.com/reference/array/array/operator\[\]/](https://cplusplus.com/reference/array/array/operator[]/)

	<h2>matriz::operador[] - Referencia de C++</h2> <p>mimatriz contiene: 0 1 2 3 4 5 6 7 8 9</p> <p>cplusplus.com</p>
--	--------------------------------------------------------------------------------------------------------------------

<https://www.tuwlab.com/ece/22227>

	<h2>[C++] La definición y la implementación d...</h2> <p>* Nota: Este artículo se escribió después de luchar durante los últimos tres días, de no poder escribir una sola línea de código debido a ...</p> <p>www.tuwlab.com</p>
--	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Uno

Suscribir

Otras publicaciones en la categoría '**Programación**'

[42Seoul] ft_containers[0] - Resumen de la tarea (1)	2022.09.24
[42Seúl] Módulo CPP 08 - STL (1)	2022.09.11
[42Seúl] Módulo CPP 06 - Conversión de tipo (1)	2022.09.11
[42Seoul] Módulo 05 de CPP: reutilización y manejo de excepciones (3)	2022.09.10
[42Seoul] CPP Módulo 04 - Polimorfismo y clases abstractas (0)	2022.09.10

etiqueta

#42Seúl, #cpp

Artículos relacionados con '**Programación**'

[42Seoul] ft_cont...

[42Seúl] Módulo ...

[42Seúl] Módulo ...

[42Seoul] Módul...

Mi registro técnico de liquidación de deudas

Este es el blog de Aggrodonk.

Suscribir



sí 2024.02.12 19:27 Informe

[Editar/eliminar](#) [dirección de comentario](#) [Escribir comentario](#)

¿No está std::size_t disponible desde C++ 11/17?
https://en.cppreference.com/w/cpp/types/size_t

☐ Secreto

Por favor ingrese sus valiosos comentarios.

Deja un comentario

Mensajes recientes

- [42Seúl] ft_containers[4] - Mapa...
- [42Seúl] ft_containers[3] - TR...
- [42 Seúl] ft_containers[2] - Beck...
- [42Seúl] ft_containers[1] - S...
- [42Seúl] ft_containers[0] - y...

buscar

Por favor esc

visitantes totales

100.591

hoy	55
ayer	68