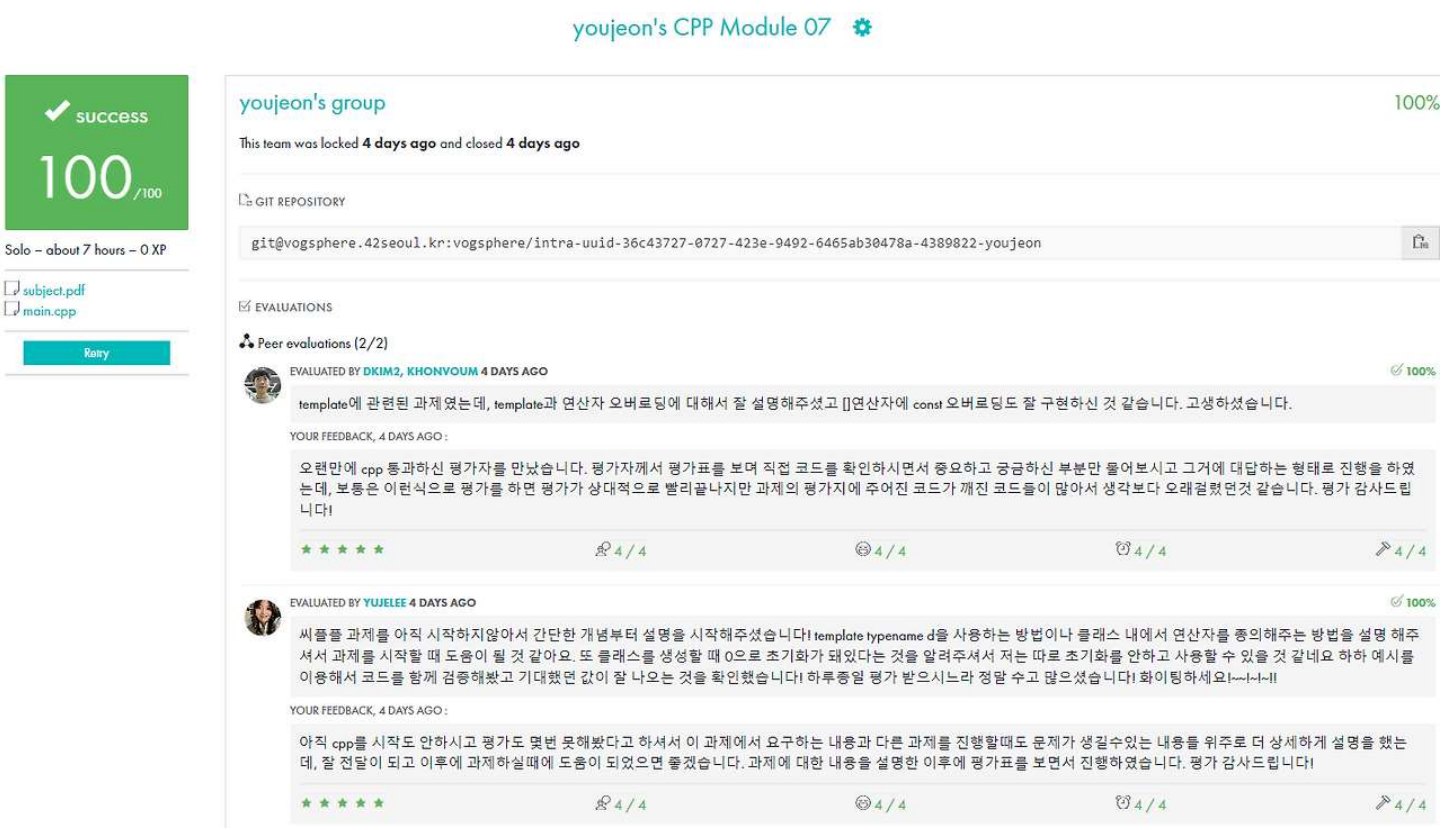
programación

[42Seoul] Módulo 07 del CPP - Plantilla

Aggrodonk 2022. 9. 11. 18:28

Introducción



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

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

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 argu mentos 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 impri me una función específica. Sin esta función habría que crear decenas de funciones llamadas swa p 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 "fun ción de plantilla".

Otra ventaja de esta función es que no solo el tipo de datos sino también cualquier clase se pued e usar como argumento siempre que admita la operación utilizada dentro de él (= > < en la asign ación). Lo entenderá al ver que las funciones de conversión que ya ha utilizado funcionan sin imp ortar 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 fu nción como argumento.

Hay muchas condiciones y puede parecer difícil, pero no lo es si entiendes el modelo. Al principi o, estaba un poco confundido acerca de tener que poder recibir un puntero de función como arg umento como función de plantilla, pero después de ver que funcionaba incluso si lo hacía igual, l o 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.tpp"

#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 p rofunda, y la función de tamaño es size_t size(void); Creo que bastaría con declararlo como .

Dice que puede incluir o no Array.tpp 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 simult áneamente en el encabezado.

En general, una clase contiene solo definiciones en el encabezado y la declaración y la implemen tación se realizan en un archivo cpp, pero dado que una clase de plantilla es una muestra para im primir una clase y no es una clase, se produce un error de enlace ya que no se puede trabajar sol o 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 plan tillas 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 decl aración 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 separad o, deberías implementarlo así.

No es muy diferente de lo que se implementó anteriormente. Sin embargo, pensé que sería mejo r implementarlo por separado en lugar de implementarlo en el encabezado, así que lo intenté, p ero tuve algunos problemas con la codificación porque el autocompletado o la verificación de err ores 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 implem entar el operador [], también se debe implementar const. De lo contrario, puede suceder que el o perador predeterminado no esté cargado y se pueda generar un error.

https://cplusplus.com/reference/array/array/operator[]/

matriz::operador[] - Referencia de C++

mimatriz contiene: 0 1 2 3 4 5 6 7 8 9

cplusplus.com

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

[C++] La definición y la implementación d...

※ Nota: Este artículo se escribió después de luchar durante los últim os tres días, de no poder escribir una sola línea de código debido a ...

www.tuwlab.com

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

nombre contraseña 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