Start with a few functions

El ejercicio propuesto se centra en la implementación de tres plantillas de funciones en C++98: swap, min y max. Estas funciones están diseñadas para operar con argumentos de cualquier tipo, siempre que ambos sean del mismo tipo y admitan los operadores de comparación necesarios.

Propósito de aprendizaje:

El objetivo principal de este ejercicio es familiarizar al estudiante con las **plantillas de funciones** en C++98. Las plantillas permiten escribir funciones genéricas que pueden operar con diferentes tipos de datos, promoviendo la reutilización de código y la flexibilidad en la programación. Además, se busca que el estudiante comprenda cómo definir y utilizar estas plantillas, así como la importancia de los operadores de comparación en funciones genéricas.

Descripción detallada de las funciones a implementar:

1. **swap**: Esta función intercambia los valores de dos argumentos proporcionados. No retorna ningún valor. Para lograr el intercambio, es necesario que los argumentos sean pasados por referencia, de modo que los cambios realizados dentro de la función afecten a las variables originales. El uso de referencias en C++98 permite que una función modifique directamente las variables que se le pasan, en lugar de trabajar con copias de ellas.
2. **min**: Esta función compara dos valores y retorna el menor de ellos. Si ambos valores son iguales, retorna el segundo. Para que la comparación sea posible, los tipos de los argumentos deben soportar el operador de comparación "menor que" (<). Al utilizar plantillas, esta función puede operar con cualquier tipo que implemente dicho operador, ya sean tipos primitivos como int o float, o tipos definidos por el usuario que sobrecarguen el operador <.
3. **max**: Similar a min, pero retorna el mayor de los dos valores. Si ambos son iguales, retorna el segundo. Requiere que los tipos de los argumentos soporten el operador "mayor que" (>). Al igual que min, esta función es genérica y puede trabajar con cualquier tipo que implemente el operador >.

Cómo abordar el ejercicio en C++98:

Para implementar estas funciones en C++98, es fundamental comprender las **plantillas de funciones** y el uso de **referencias**.

- **Plantillas de funciones**: Permiten definir funciones genéricas que operan con diferentes tipos de datos. La sintaxis básica para declarar una plantilla de función es:

```
template <typename T>
 void funcionEjemplo(T parametro) {
```

```
    // Implementación
}
```

Aquí, T es un parámetro de tipo que se determinará en tiempo de compilación según el tipo de argumento que se pase a la función.

- **Referencias**: Son alias de variables existentes. Permiten que una función modifique directamente la variable original pasada como argumento. Para declarar una referencia, se utiliza el operador & en la declaración del parámetro de la función:

```
void funcionConReferencia(int &ref) {
    // ref es una referencia a una variable int
}
```

En el caso de la función swap, al recibir sus parámetros como referencias, puede intercambiar los valores de las variables originales.

Recursos recomendados:

A continuación, se proporciona una selección de recursos en video en español que te ayudarán a abordar con éxito el ejercicio propuesto:

1. **"Curso C++ Básico: Plantillas de funciones"**: Este video ofrece una introducción detallada sobre cómo trabajar con plantillas de funciones en C++, incluyendo su relación con las funciones sobrecargadas.

   https://www.youtube.com/watch?v=H_2L1hDrWjk&utm_source=chatgpt.com

2. **"Tutorial C++: Operadores de Comparación"**: En este tutorial, se explican los operadores de comparación en C++, fundamentales para implementar las funciones min y max.

   https://www.youtube.com/watch?v=s9jL3dFyKQM&utm
3. **"Curso de C++: Plantillas de Funciones"**: Este video profundiza en la creación y uso de plantillas de funciones en C++, proporcionando ejemplos prácticos para su comprensión.

   https://www.youtube.com/watch?v=HKDrYKWTgos&utm
4. **"Curso de C++ Avanzado: Especialización de Plantillas de Funciones"**: Este recurso aborda la especialización de plantillas de funciones, ampliando el conocimiento sobre cómo adaptarlas a diferentes tipos de datos.

   https://www.youtube.com/watch?v=3vVUSW-O-8M&utm

5. **"Primeros pasos con Standard Template Library de C++"**: Este video introduce la biblioteca estándar de plantillas en C++, lo que puede ser útil para comprender el contexto más amplio de las plantillas en el lenguaje.

   https://www.youtube.com/watch?v=uBKzM-MBj18&utm

Estos recursos te proporcionarán una comprensión sólida de las plantillas de funciones, operadores de comparación y la organización de archivos en C++, facilitando la implementación exitosa del ejercicio propuesto.

El código

Este código implementa y prueba tres funciones genéricas en C++98 utilizando **plantillas de funciones** (*function templates*). Estas funciones permiten operar con cualquier tipo de dato siempre que cumplan con ciertos requisitos, como soportar operadores de comparación.

1. whatever.hpp (Archivo de cabecera)

El archivo **whatever.hpp** contiene las definiciones de tres plantillas de funciones: swap, min y max.

1.1. #pragma once

Esta línea es un **include guard** (protección contra inclusión múltiple). Evita que el archivo de cabecera se incluya más de una vez en el mismo proyecto, lo que podría causar errores de compilación.

#pragma once

1.2. swap (Intercambiar valores)

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

- Esta función **intercambia los valores** de a y b.
- Usa **paso por referencia** (T &a, T &b), lo que significa que modifica directamente las variables originales.
- Crea una variable temporal tmp del mismo tipo que a y b para guardar el valor de a.
- Luego, asigna el valor de b a a y el valor de tmp (que almacenaba a) a b.

- Esto **no devuelve nada** (void), solo modifica los valores.

📌 **Ejemplo práctico:** Si a = 2 y b = 3, después de llamar a swap(a, b), a será 3 y b será 2.

1.3. min (Obtener el menor valor)

```
template <typename T>
T		min(T a, T b) {
	return (a < b ? a : b);
}
```

- Esta función **devuelve el menor** de los dos valores a y b.
- Utiliza el **operador ternario** (? :), que es una forma compacta de escribir una estructura if-else:
  - Si a < b, devuelve a.
  - Si a >= b, devuelve b.
- Los parámetros se pasan **por valor** (copia), por lo que no se modifican las variables originales.

📌 **Ejemplo práctico:** Si a = 2 y b = 3, min(a, b) devolverá 2.

1.4. max (Obtener el mayor valor)

```
template <typename T>
T		max(T a, T b) {
	return (a > b ? a : b);
}
```

- Similar a min, pero **devuelve el mayor** valor.
- Usa el **operador ternario** para decidir:
  - Si a > b, devuelve a.
  - Si a <= b, devuelve b.

📌 **Ejemplo práctico:** Si a = 2 y b = 3, max(a, b) devolverá 3.

2. main.cpp (Función main)

Este archivo contiene el programa principal, que prueba las funciones definidas en whatever.hpp.

```
int	main() {

	int a = 2;
	int b = 3;

	::swap( a, b );
```

```
std::cout << "a = " << a << ", b = " << b << std::endl;
std::cout << "min( a, b ) = " << ::min( a, b ) << std::endl;
std::cout << "max( a, b ) = " << ::max( a, b ) << std::endl;
```

- **Intercambia los valores** de a y b usando swap(a, b).
- **Imprime los valores** intercambiados (a = 3, b = 2).
- **Llama a min(a, b)**, que devuelve 2, e imprime el resultado.
- **Llama a max(a, b)**, que devuelve 3, e imprime el resultado.

📌 **Salida esperada:**

```
a = 3, b = 2
min(a, b) = 2
max(a, b) = 3
```

Después, el programa prueba las mismas funciones con **cadenas de texto** (std::string):

```
std::string c = "chaine1";
        std::string d = "chaine2";

        ::swap(c, d);
        std::cout << "c = " << c << ", d = " << d << std::endl;
        std::cout << "min( c, d ) = " << ::min( c, d ) << std::endl;
        std::cout << "max( c, d ) = " << ::max( c, d ) << std::endl;

return 0;
}
```

- **Intercambia los valores** de c y d usando swap(c, d).
- **Imprime los valores** intercambiados (c = "chaine2", d = "chaine1").
- **Llama a min(c, d)**, que devuelve "chaine1" (las comparaciones entre std::string se hacen alfabéticamente).
- **Llama a max(c, d)**, que devuelve "chaine2".

📌 **Salida esperada:**

```
c = chaine2, d = chaine1
min(c, d) = chaine1
max(c, d) = chaine2
```

Resumen del código

1. **swap(T &a, T &b)**: Intercambia los valores de a y b (paso por referencia).
2. **min(T a, T b)**: Devuelve el menor de los dos valores.
3. **max(T a, T b)**: Devuelve el mayor de los dos valores.

4. **El main prueba estas funciones con int y std::string.**

El código **demuestra la flexibilidad de las plantillas**, permitiendo escribir funciones genéricas que funcionan con múltiples tipos de datos sin necesidad de duplicar código.

Conceptos clave que se usan en el código

- **Plantillas de funciones (template <typename T>)** para definir funciones genéricas.
- **Paso por referencia (T &a, T &b)** en swap, para modificar valores originales.
- **Paso por valor (T a, T b)** en min y max, para evitar modificaciones.
- **Operador ternario (? :)** para hacer comparaciones de forma compacta.
- **Uso de std::string**, que soporta operadores de comparación (<, >).

Otros recursos recomendados

Aquí tienes algunos videos más que explican los conceptos usados:

Entiendo que los recursos previamente recomendados no están disponibles. A continuación, te proporciono una nueva selección de videos en español que te ayudarán a comprender los conceptos clave relacionados con el código discutido:

1. **"¿Qué es una plantilla de función en C++?"**
   Descripción: Este video ofrece una explicación detallada sobre las plantillas de funciones en C++, incluyendo su sintaxis y aplicaciones prácticas.
   Enlace: https://www.youtube.com/watch?v=T_Df_rwl2oM

2. **"Paso de parámetros por valor y por referencia | 2/7 | UPV"**
   Descripción: Este recurso de la Universidad Politécnica de Valencia explica las diferencias entre el paso de parámetros por valor y por referencia en C++, conceptos fundamentales para entender cómo se manejan los datos en funciones.
   Enlace: https://www.youtube.com/watch?v=n1zOM5AOKjE

3. **"Explicación y ejemplos de plantillas en C++"**
   Descripción: Este video proporciona una visión general sobre las plantillas en C++, incluyendo ejemplos prácticos que ilustran cómo y cuándo utilizarlas.
   Enlace: https://www.youtube.com/watch?v=5QI3unnfKF0

4. **"Curso C++. Funciones III. Paso por valor y por referencia. Vídeo 36"**
   Descripción: En este video se profundiza en las funciones en C++, específicamente en cómo se pasan los parámetros por valor y por referencia, y cómo esto afecta el comportamiento de las funciones.
   Enlace: https://www.youtube.com/watch?v=Jjgay95by2s

5. **"Plantillas en C++ /** Explicación e Implementación*/"**

Descripción: Este video aborda la implementación y uso de plantillas en C++, ofreciendo una explicación clara y ejemplos prácticos para facilitar su comprensión.
Enlace: https://www.youtube.com/watch?v=6AVfaqhRFy8

Espero que estos recursos te sean útiles para comprender y abordar con éxito el ejercicio propuesto.

📌 Conclusión:

El código implementa **tres funciones genéricas** con **plantillas de funciones** en C++98, permitiendo intercambiar valores (swap), encontrar el menor (min) y el mayor (max). Se prueban con int y std::string, mostrando cómo las plantillas hacen el código más reutilizable y flexible.