PISTAS.

1.-Empieza por escribir una función que tome cualquier cantidad de argumentos y que devuelva un array que contenga el tamaño de cada tipo de argumento. Utiliza el operador sizeof para obtener el tamaño de cada tipo.

1. template<typename... Args>

2. constexpr auto get\_type\_sizes()

3. {

4. return std::array<std::size\_t, sizeof...(Args)>{sizeof(Args)...};

5. }

6.

2.-Llama a la función get\_type\_sizes con una lista de tipos y almacena el resultado en una variable. Después, muestra el tamaño de cada tipo en la consola utilizando el array devuelto por la función.

1. auto sizes = get\_type\_sizes<short, int, long, long long>();

2. std::cout << "Tamaño de short: " << sizes[0] << std::endl;

3. std::cout << "Tamaño de int: " << sizes[1] << std::endl;

4. std::cout << "Tamaño de long: " << sizes[2] << std::endl;

5. std::cout << "Tamaño de long long: " << sizes[3] << std::endl;

6.

3.-Para obtener el número total de argumentos, utiliza el operador sizeof en la lista de argumentos.

1. std::cout << "Número de argumentos: " << sizeof...(Args) << std::endl;

Para mostrar el tamaño de cada argumento en la consola, puedes utilizar una 4.- función auxiliar que llame a la función sizeof para cada argumento. Después, llama a esta función auxiliar con cada argumento utilizando el operador de expansión de paquetes.

1. template<typename T>

2. void show\_size(T arg)

3. {

4. std::cout << "Tamaño en bytes de arg: " << sizeof(arg) << std::endl;

5. }

6.

7. template<typename... Args>

8. void show\_size(Args... args)

9. {

10. (show\_size(args), ...);

11. }

12.

5.- Por último, para sumar una cantidad variable de argumentos, puedes utilizar un patrón similar al utilizado en el paso 4. Define una función auxiliar que sume dos argumentos y llámala utilizando el operador de expansión de paquetes para sumar todos los argumentos.

1. template<typename T>

2. T sum(T arg1, T arg2)

3. {

4. return arg1 + arg2;

5. }

6.

7. template<typename T, typename... Args>

8. T sum(T arg1, T arg2, Args... args)

9. {

10. return sum(arg1 + arg2, args...);

11. }

12.