1. Cree una función que reciba un **char&** como argumento y lo modifique. En **main()**, imprima una variable **char**, llame a la función con esa variable e imprima para comprobar que ha cambiado.

2. Cree una función que retorne el próximo valor en una sucesión de Fibonacci cada vez que es llamada. Agregue un argumento **bool** con valor por defecto **false**, que indique cuando resetear la secuencia para volver a empezar. Pruebe la función en un **main**.

3. Utilice inicialización agregada para crear un arreglo de **double** en la cual se especifica el tamaño del arreglo pero no se prevén suficientes elementos. Imprima el arreglo utilizando el operador **sizeof** para determinar el tamaño del arreglo. Ahora cree un arreglo de **double** utilizando inicialización agregada con dimensionamiento automático. Imprima el arreglo.

4. Utilizando inicialización agregada cree un arreglo de strings. Cree un **stack** para mantener esos strings (agréguelos con el método **Stack::push**). Finalmente remuévalos e imprímalos.

5. Demuestre el dimensionamiento automático e inicialización agregada con un arreglo de objetos creados con instancias de la clase del ejercicio 2 de la práctica anterior. Agregue una función miembro a la clase para que imprima un mensaje. Calcule el tamaño del arreglo y recórralo llamando al nuevo método.

6. Defina tres constantes enteras, luego súmelas para generar un valor que determina el tamaño de un arreglo en una definición. Intente compilar el mismo código en C y vea que ocurre.

7. Cree una definición **const** en un archivo de encabezado (.h), incluya ese archive desde dos archivos fuente (.cpp), compile esos archivos y haga el link de ellos. No debería tener errores. Intente lo mismo con archivos fuentes en C.

8. Cree un arreglo constante de caracteres y trate de cambiar algún caracter. Probar con:

```
char *p = "ojo";
char q[] = "chau";
const char *r = "hola";
```

9. Escriba una clase que tiene una función miembro **const** y una non-**const**. Escriba tres funciones que reciban un objeto de esa clase como argumento; la primera la toma por valor, la segunda por referencia y la tercera por una referencia **const**. Dentro de las funciones, intente llamar a ambas funciones miembro de la clase, explique los resultados.

10. Cree una clase que contenga un entero, un constructor que inicialice el entero con el argumento del constructor, una función miembro que imponga un valor a ese entero con su argumento, y una función **print()** que lo imprima. Ponga a la clase en un archivo header e inclúyalo en dos archivos fuentes. En uno de ellos defina un objeto instancia de su clase y en el otro declare el mismo identificador como **extern** y pruébelo desde un **main.** Recuerde que tendrá que linkar ambos archivos objetos.

11. En el ejercicio anterior haga **static** a la instancia y verifique que el linker da un error.