Los archivos generado deben respetar el siguiente formato de nombre guiaDeClase05_ejercicioNumero.c
Por ejemplo el archivo del ejercicio 1 debe llevar el nombre `guiaDeClase05_01.c` Si el ejercicio tuviera items a y b por ejemplo el nombre deberá ser `guiaDeClase05_01_A.c` para el punto A.
Todos los archivos deberán ser subidos al repositorio dentro de una carpeta con el nombre guiaDeClase05
Todos los archivos deben estar comentados con doxygen y las respuestas a las preguntas realizadas deben ser contestadas usando el tag \note

1. Implemente un programa en el cual debe definir un vector de diez números enteros (use como tipo de dato int) e incialicelo en tiempo de ejecución con los números del 0 al 9. Luego imprima el vector en orden ascendente y descendente. Use define para definir la cantidad de elementos.

2. Escriba un programa en el cual debe definir un vector que almacene todas las letras del alfabeto, salvo la eñe (use como tipo de dato char). Inicialicelo en tiempo de ejecución. Imprímalo en orden alfabético.

3. Escriba un programa en el cual se defina un vector inicializado en tiempo de compilación con la tabla de multiplicar del cinco. Pida al usuario que ingrese un número entre cero y diez, luego devuelva el resultado de la multiplicación por cinco (la operación debe resolverse indexando el vector). El programa termina cuando el usuario ingresa un número fuera del rango válido.

4. Implemente un código que genere un número pseudo aleatorio entre [0; 99] , luego el programa debe pedirle al usuario números e indicarle si acertó o no el número generado pseudo aleatoriamente.
   Si el usuario repite un número el programa deberá indicarlo, lo mismo si el usuario ingresa uno fuera de rango.

5. Realice un programa que permita al usuario ingresar las alturas de un grupo de como máximo cien personas. El fin del ingreso de datos ocurre cuando la altura ingresada sea menor que cero.Luego se le pedirá al usuario que ingrese dos valores de altura y el programa debe indicar la cantidad de personas con alturas en ese rango. Si el intervalo ingresado por el usuario es inválido indiquelo por stdout.

6. Implemente un programa que le pida al usuario números enteros y los almacene en 4 vectores diferentes según su tipos.
   - Positivos y el cero.
   - Negativos.
   - Pares.
   - Impares.

   El usuario ingresara diez valores y luego el programa deberá imprimir por stdout la cantidad de números almacenados en cada vector y posteriormente los datos almacenados en cada uno de ellos.

7. Implemente un programa que le pida al usuario que ingrese una palabra por stdin y la imprima en mayúscula por stdout. Use la función toupper y fgets.

8. Implemente un programa que le pida al usuario que ingrese una palabra por stdin e informe la cantidad de caracteres que esta posee sin contar el ‘\0’.

9. Implemente un programa que le pida al usuario que ingrese dos palabras por stdin e indique si son iguales o cual aparece primero en el diccionario.

10. Implemente un programa que le pida al usuario que ingrese una palabra y un carácter por stdin. A continuación reemplace este carácter en la palabra por asteriscos para finalmente debe indicar la cantidad de veces que reemplazó el carácter.

11. Responda las siguientes preguntas, use el tag \note de doxygen para colocar sus respuestas.
    - ¿Que funcion cumple el operador &?
    - Dibuje el mapa de memoria.
    - ¿Porque cada vez que ejecuta el programa tienen direcciones distintas?
    - ¿De qué depende la cantidad de bytes que ocupa la dirección de la variable?
    - ¿Hay alguna relación entre el tipo de dato de una variable y el tamaño de su dirección?

```
#include <stdio.h>

int main (void)
{
int varIntA, varIntB;
char varCharA, varCharB;

     printf ("varIntA = %08x; %p\r\n", varIntA, &varIntA);
     printf ("varIntB = %08x; %p\r\n", varIntB, &varIntB);
     printf ("varCharA = %08x; %p\r\n", varCharA, &varCharA);
     printf ("varCharB = %08x; %p\r\n", varCharB, &varCharB);

     return (0);
}
```

12. Genere dos vectores de 5 elementos uno de tipo char y otro de tipo int. Imprima las direcciones de cada elemento de ambos vectores. Escriba sus conclusiones en el código usando el tag \note de doxygen.

13. Indique que imprime por stdout el siguiente código. ¿Que ocurre si sacamos el casteo de vect?

```
#include <stdio.h>
#define CANT ((int)10)    //!< Cantidad de elementos del vector

int main (void)
{
int vect[CANT];

        printf ("0x%08X\r\n", (int)vect);

        return (0);
}
```

14. Implemente las siguientes tres funciones en un archivo llamado myStr.c con su correspondiente myStr.h (Ninguna de estas funciones debe imprimir en pantalla)
    - int myStrCmp (char *s0, char *s1);
        - Esta función compara dos string
        - La función devuelve
            - Cero si las palabras son iguales
            - Un número negativo si la palabra almacenada en s0 aparece primero que la almacenada en s1;
            - Un número positivo si la palabra almacenada en s0 aparece primero que la almacenada en s1;
        - s0 y s1: Son punteros a los strings a comparar.

    - void myStrUpper (char *v);
        - Esta función pasa a mayúscula el string que se pasa como parámetro
        - v: Es un puntero al string que debe pasar a mayúscula.

    - int myStrChr (char *v, char c)
        - Reemplaza el caracter c por un * en el string apuntado por v
        - Devuelve la cantidad de caracteres reemplazados.

    Implemente un main que testee cada función de forma automática.