1. Hacer un programa que defina vectores de 2 componentes de todos los tipos de variables que conoce (**`char, int, float, double`**) y de algunas estructuras (**`struct Complex, struct Punto2D`**) e imprima la dirección de memoria de cada uno de los componentes de cada uno de los vectores, utilizando el especificador de campo **`%p`**. ¿Cuál es la distancia en bytes entre componentes y variables? Comparar con lo que retorna el operador **`sizeof()`**.

2. Implementar la función **`TransponeNN`** que reciba una matriz cuadrada como puntero y la trasponga sobre sí misma. Utilizar el prototipo:

```
void TransponeNN(double *matriz, int n);
```

   Ejemplo de uso:

```
double m[10][10];
...
TransponeNN(m, 10);
```

3. Implementar la función **`CompareStrings`** que compare strings asciibéticamente devolviendo 0 si los strings son iguales, un valor positivo si el primero es mayor y un valor negativo si el segundo es mayor. Utilizar el prototipo:

```
int CompareStrings(const char *str1, const char *str2);
```

4. Implementar la función **`Invierte`** que reciba un string como argumento, lo invierta sobre sí mismo utilizando 2 punteros y devuelva un puntero al resultado. Utilizar el prototipo:

```
char *Invierte(char *str);
```

5. Dada la esctructura:

```
struct Persona {
    char nombre[64];
    char apellido[64];
    int dni;
};
```

   Implementar la función **`CargaPersona`** que reciba un puntero a una **`struct Persona`** e inicialice sus miembros leyéndolos del dispositivo de entrada standard.

```
void CargaPersona(struct Persona *pper);
```

6. ¿Cuáles son las ventajas de mandar como argumento a una función un puntero a una estructura en vez de la estructura misma?