0x0E. C - Structures, typedef

=> dog.h

#ifndef DOG

#define DOG

   /\*\*

    \* struct dog - dog class

    \*

    \* @name: Name of the dog

    \*

    \* @age: Age of the dog

    \*

    \* @owner: Owner of the dog

    \*/

   struct dog

{

   char \*name;

   float age;

   char \*owner;

};

void init\_dog(struct dog \*d, char \*name, float age, char \*owner);

void print\_dog(struct dog \*d);

typedef struct dog dog\_t;

dog\_t \*new\_dog(char \*name, float age, char \*owner);

void free\_dog(dog\_t \*d);

#endif

=>1-init\_dog.c

#include "dog.h"

/\*\*

\* init\_dog - initializes a structure of type dog

\* @d: pointer to structure

\* @name: pointer to the name of the dog

\* @age: age of dog

\* @owner: pointer to owner of dog

\*

\* Return: void

\*/

void init\_dog(struct dog \*d, char \*name, float age, char \*owner)

{

   if (d)

   {

       d->name = name;

       d->age = age;

       d->owner = owner;

   }

}

=> 2-print\_dog.c

#include <stdio.h>

#include "dog.h"

/\*\*

\* print\_dog - prints a struct dog

\* @d: pointer to structure

\*

\* Return: void

\*/

void print\_dog(struct dog \*d)

{

   if (d != NULL)

   {

       if (d->name != NULL)

           printf("Name: %s\n", d->name);

       else

           printf("Name: (nil)\n");

       printf("Age: %.6f\n", d->age);

       if (d->owner != NULL)

           printf("Owner: %s\n", d->owner);

       else

           printf("Owner: (nil)\n");

   }

}

=>4-new\_dog.c

#include <stdlib.h>

#include "dog.h"

/\*\*

\* new\_dog - creates a new dog

\* @name: name of dog

\* @age: age of dog

\* @owner: owner of dog

\*

\* Return: pointer to new dog

\*/

dog\_t \*new\_dog(char \*name, float age, char \*owner)

{

   unsigned int nl, ol, i;

   dog\_t \*dog;

   if (name == NULL || owner == NULL)

       return (NULL);

   dog = malloc(sizeof(dog\_t));

   if (dog == NULL)

       return (NULL);

   for (nl = 0; name[nl]; nl++)

       ;

   nl++;

   dog->name = malloc(nl \* sizeof(char));

   if (dog->name == NULL)

   {

       free(dog);

       return (NULL);

   }

   for (i = 0; i < nl; i++)

       dog->name[i] = name[i];

   dog->age = age;

   for (ol = 0; owner[ol]; ol++)

       ;

   ol++;

   dog->owner = malloc(ol \* sizeof(char));

   if (dog->owner == NULL)

   {

       free(dog->name);

       free(dog);

       return (NULL);

   }

   for (i = 0; i < ol; i++)

       dog->owner[i] = owner[i];

   return (dog);

}

=>5-free\_dog.c

#include <stdlib.h>

#include "dog.h"

/\*\*

\* free\_dog - frees dogs

\* @d: pointer to dog to free

\*

\* Return: void

\*/

void free\_dog(dog\_t \*d)

{

   if (d)

   {

       free(d->name);

       free(d->owner);

       free(d);

   }

}