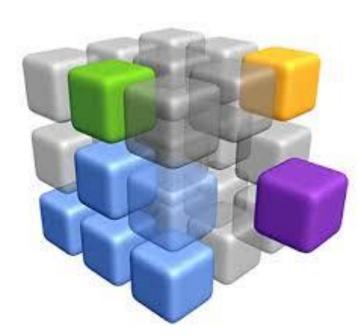
C

TDA



Conceptos

- Modelo
 - Estructura de datos (struct)
 - Comportamiento (funciones)
- "Lifetime":
 - "Create"
 - "Destroy"
- ¿Quien debe alocar la estructura?
 - "Create"
 - Usuario del TDA

<u>Ejemplo - StackFile.h:</u>

#endif

```
#ifndef FILESTACK
#define FILESTACK
#define XOPEN SOURCE 500
#include <stdlib.h>
#include <stdio.h>
typedef struct {
 FILE* file;
  size t item size;
} filestack t;
/* Crea el filestack usando 'filepath' y usando 'item size' bytes para cada elemento */
void filestack create(filestack t *self, const char* filepath, size t item size);
/* Pone un elemento ('value') en el stack */
void filestack put(filestack t* self, void* value);
/* Saca el primer elemento del filestack y lo pone en 'buffer' (asume que apunta a un
lugar válido) */
void filestack pop(filestack t *self, void* buffer);
/* Destruye el filestack y libera los reursos de memoria */
void filestack destroy(filestack t *self);
```

<u>Ejemplo - StackFile.c:</u>

```
#include "filestack.h"
#include <unistd.h>
#include <sys/types.h>
void filestack create(filestack t *self, const char* filepath, size t item size) {
    self->file = fopen(filepath, "a+");
    self->item size = item size;
void filestack put(filestack t* self, void* value) {
    fwrite(value, self->item size, 1, self->file);
void filestack pop(filestack t *self, void* buffer) {
    fseek(self->file, -self->item size, SEEK END);
    fread(buffer, self->item size, 1, self->file);
    fseek(self->file, -self->item size, SEEK END);
    ftruncate(fileno(self->file), ftell(self->file));
void filestack destroy(filestack t *self) {
    fclose(self->file);
```

Ejemplo - main.c:

```
#include <stdio.h>
#include "filestack.h"
int main(int argc, char* argv[])
    filestack t f;
    int i;
    filestack create(&f, "pila.dat", sizeof(int));
    for (int value = 0; value < 10; ++value)</pre>
        filestack put(&f, &value);
    for (i = 0; i < 5; ++i)
        int value = 0;
        filestack pop(&f, &value);
        printf("Value: %d\n", value);
    filestack destroy(&f);
    return 0;
```