Fundamendez - 2C 2024 - Clase 5

Registros y Strings

Ejercicio 1

Lamentablemente para Garfio, Jack Sparrow encontró su tesoro y se lo robó porque es mejor pirata. Ahora que tiene el tesoro, quiere saber cuál es el valor total del mismo y para eso, creó structs para representarlo.

```
typedef struct joya {
    int valor;
    int antiguedad;
    char color[MAX_COLOR]; // “verde”, “azul”, “rojo”
} joya_t;

typedef struct moneda {
    int valor;
    bool de_oro;
} moneda_t;

typedef struct tesoro {
    moneda_t monedas[MAX_MONEDAS];
    int tope_monedas;
    joya_t joyas[MAX_JOYAS];
    int tope_joyas;
    bool desenterrado;
} tesoro_t;
```

1. Armar una función que reciba un tesoro y devuelva su valor total, sabiendo que:
    - Si una moneda es de oro, su valor se duplica.
    - Si la antigüedad de la joya es menor a 5 años y es azul, su valor se reduce a la mitad.
2. A Jack se le cayeron un par de monedas en el camino, y quiere guardarlas en el tesoro. Hacer un procedimiento que reciba el tesoro por referencia y guarde la moneda en el mismo.

Ejercicio 2

Andy se está por ir a estudiar a la FIUBA y para comprarse una nueva PC (que se banque la máquina virtual) piensa vender toda su colección de juguetes. Para esto necesita calcular el valor total de sus juguetes y la cantidad de cada tipo que tiene.
Los tipos son:

- Peluche
- Electrónicos
- Plástico

Andy, para organizarse, tuvo la idea de hacer un inventario de juguetes para el cual necesita de nuestra ayuda para poder armarlo finalmente.
Nos pidió que:

1. Pueda agregar de a un juguete al inventario dentro del baúl con menos juguetes.
2. Que se muestre la cantidad de juguetes que hay por tipo.
3. Crear un baúl nuevo.
4. Guardar en un vector con los juguetes que se pueden vender teniendo en cuenta que:
    a. Se quieren vender los juguetes que ya tienen más de 10 años.
    b. Los peluches no los quiere vender porque tienen mucho valor sentimental

Por lo que le presentamos a Andy la siguiente biblioteca (o .h…):

```c
#include <stdio.h>
#include <string.h>
#include <stdbool.h>

#define MAX_TIPO 20
#define MAX_JUGUETES 100
#define MAX_NOMBRE 50

typedef struct juguete {
    char nombre[MAX_NOMBRE];
    int anio_comprado;
    int valor;
    char tipo[MAX_TIPO]; // "Electrónico", "Peluche", "Plástico"
} juguete_t;

typedef struct baul{
    juguete_t juguetes[MAX_JUGUETES];
    char etiqueta[MAX_NOMBRE];
    int tope_juguetes;
} baul_t;

typedef struct inventario{
    baul_t baules[MAX_BAULES];
    int tope_baules;
} inventario_t;
```

```c
//Pre: nuevo_juguete debe ser un juguete inicializado. inventario está correctamente inicializado.
//Pos: Se va a agregar el juguete al baul que tenga menos juguetes y se muestra un mensaje con la etiqueta del baul al que fue agregado. Si el inventario está lleno, no se agrega el juguete y se muestra un mensaje indicando que no hay espacio.
void agregar_juguete(inventario_t* inventario, juguete_t nuevo_juguete);

//Pre: inventario debe estar inicializado y los tipos de los juguetes deben ser “Electrónico", "Peluche", o "Plástico"
//Pos: Se cuentan y muestran la cantidad de cada tipo de peluche por pantalla
void contar_juguetes_por_tipo(inventario_t inventario);

//Pre: inventario debe estar correctamente inicializado.
//Pos: Crea un nuevo baúl vacío para guardar juguetes con la etiqueta que recibe.
void agregar_baul(inventario_t* inventario, char etiqueta[MAX_NOMBRE]);

//Pre: inventario debe estar correctamente inicializado, así como también el vector de juguetes a vender vacío con su tope inicializado correctamente.
//Pos: Guarda en el vector de juguetes aquellos que pueden ser vendidos, se tiene en cuenta que se quieren vender los que no son peluches y los que tienen más de 10 años de antigüedad.
void calcular_valor_total(inventario_t inventario, juguete_t jueguetes_a_vender[MAX_JUGUETES], int* tope_juguetes_a_vender);
```

Ahora, solo nos toca armar el código, ¿facilito no???