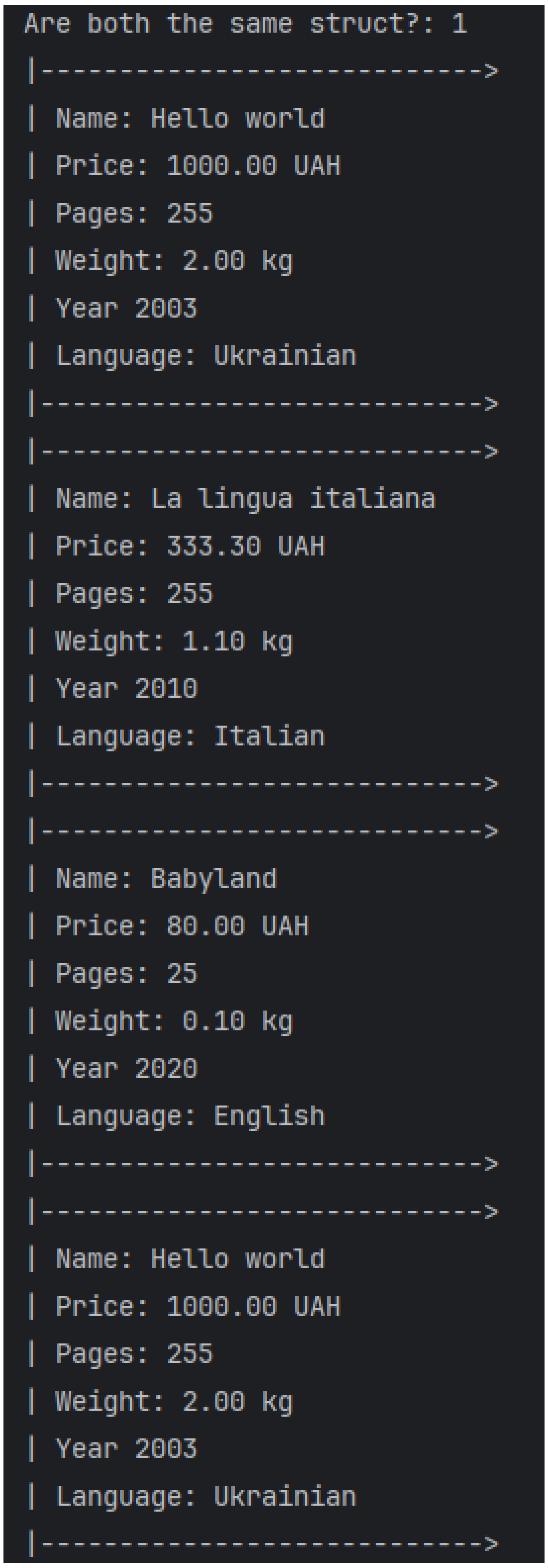
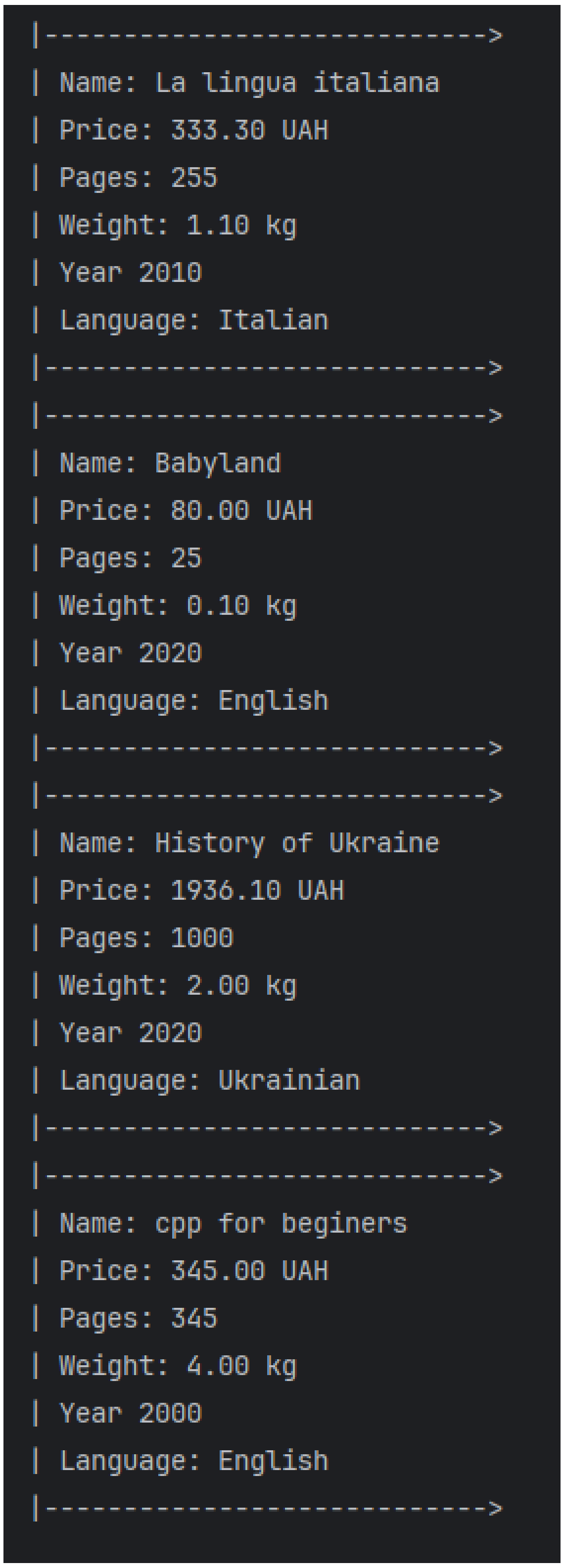
https://github.com/tonujet/c-embedded/blob/main/basics/L3/code/src/main.c

Вивід в консолі:

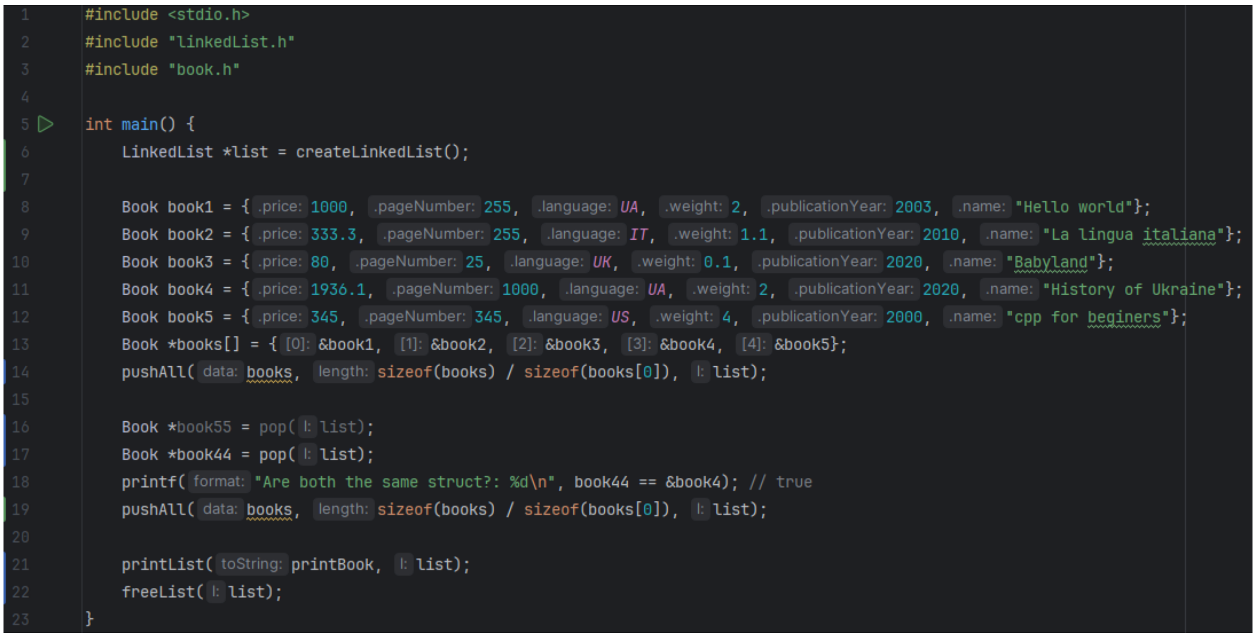


```
Are both the same struct?: 1
|---------------------------->
| Name: Hello world
| Price: 1000.00 UAH
| Pages: 255
| Weight: 2.00 kg
| Year 2003
| Language: Ukrainian
|---------------------------->
|---------------------------->
| Name: La lingua italiana
| Price: 333.30 UAH
| Pages: 255
| Weight: 1.10 kg
| Year 2010
| Language: Italian
|---------------------------->
|---------------------------->
| Name: Babyland
| Price: 80.00 UAH
| Pages: 25
| Weight: 0.10 kg
| Year 2020
| Language: English
|---------------------------->
|---------------------------->
| Name: Hello world
| Price: 1000.00 UAH
| Pages: 255
| Weight: 2.00 kg
| Year 2003
| Language: Ukrainian
|---------------------------->
```



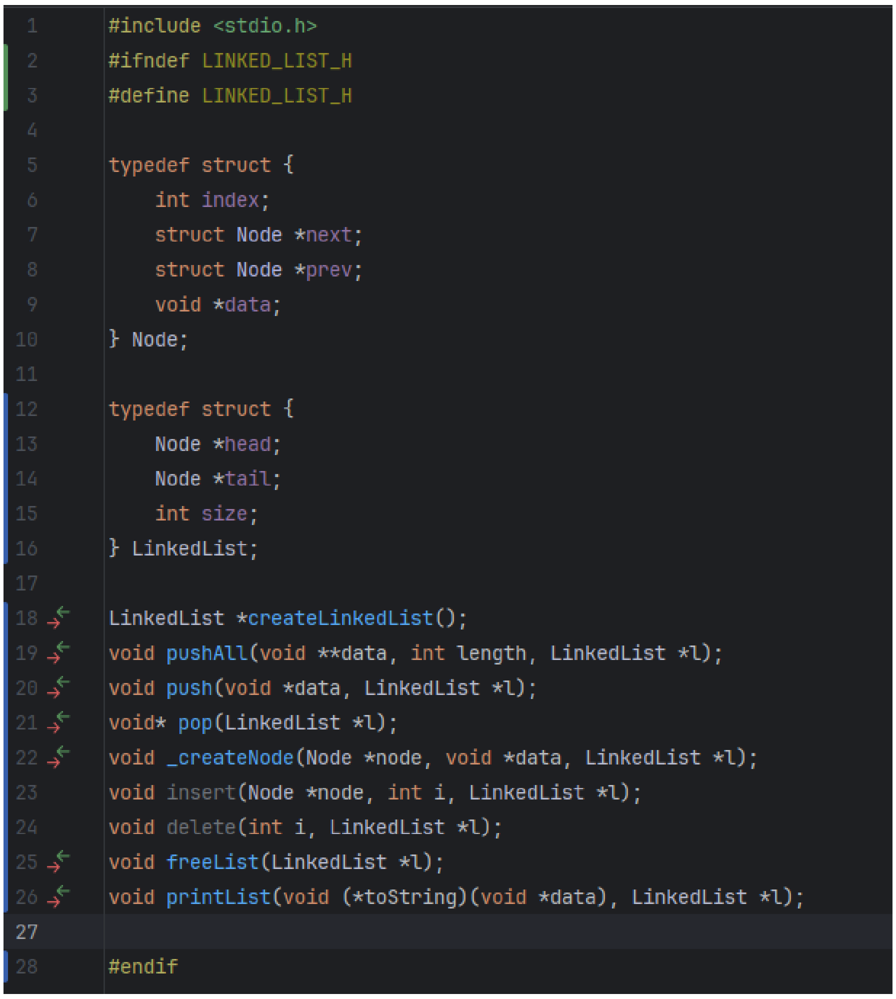
```
|---------------------------->
| Name: La lingua italiana
| Price: 333.30 UAH
| Pages: 255
| Weight: 1.10 kg
| Year 2010
| Language: Italian
|---------------------------->
|---------------------------->
| Name: Babyland
| Price: 80.00 UAH
| Pages: 25
| Weight: 0.10 kg
| Year 2020
| Language: English
|---------------------------->
|---------------------------->
| Name: History of Ukraine
| Price: 1936.10 UAH
| Pages: 1000
| Weight: 2.00 kg
| Year 2020
| Language: Ukrainian
|---------------------------->
|---------------------------->
| Name: cpp for beginers
| Price: 345.00 UAH
| Pages: 345
| Weight: 4.00 kg
| Year 2000
| Language: English
|---------------------------->
```

Код:



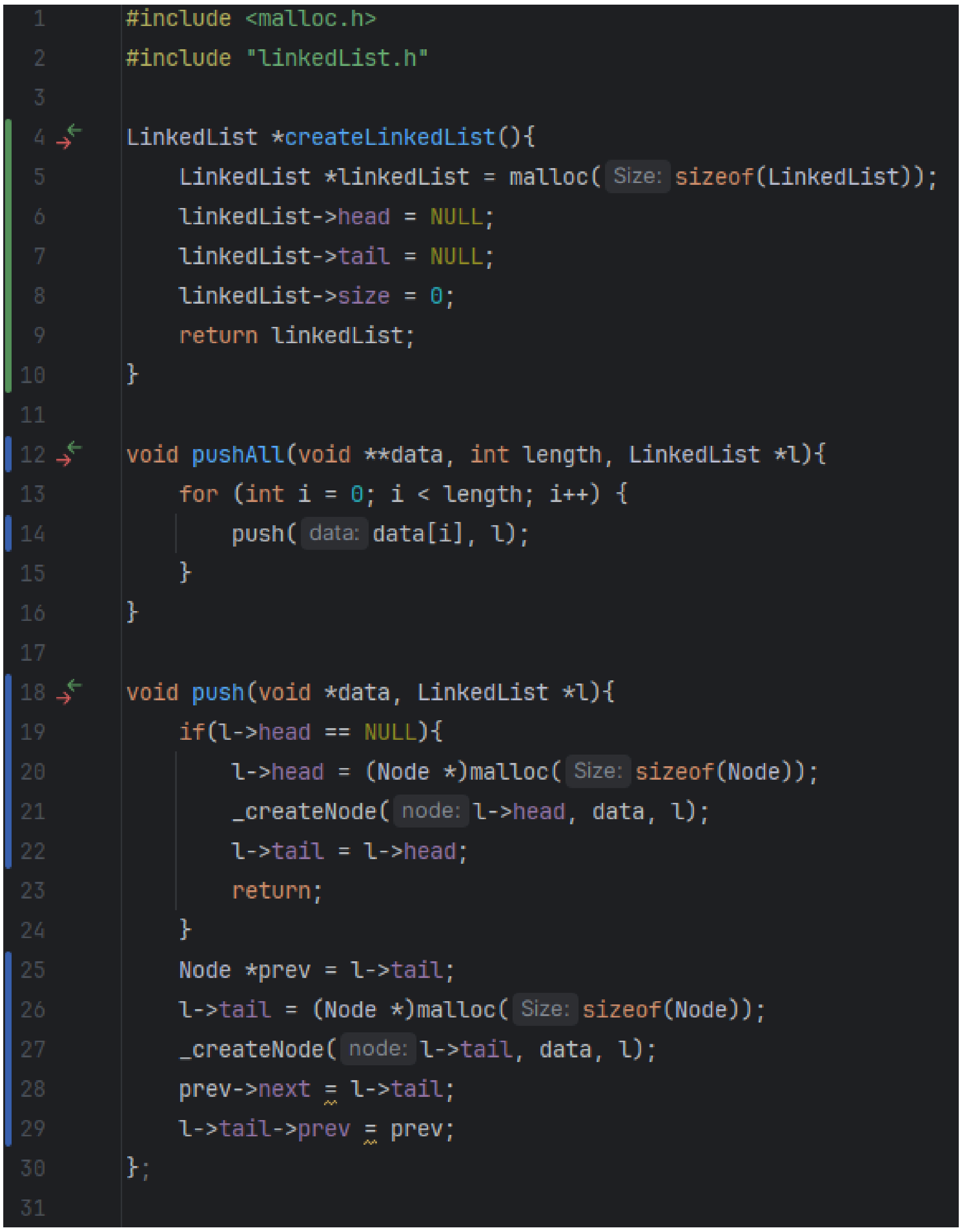
```c
#include <stdio.h>
#include "linkedList.h"
#include "book.h"

int main() {
    LinkedList *list = createLinkedList();

    Book book1 = { .price: 1000, .pageNumber: 255, .language: UA, .weight: 2, .publicationYear: 2003, .name: "Hello world"};
    Book book2 = { .price: 333.3, .pageNumber: 255, .language: IT, .weight: 1.1, .publicationYear: 2010, .name: "La lingua italiana"};
    Book book3 = { .price: 80, .pageNumber: 25, .language: UK, .weight: 0.1, .publicationYear: 2020, .name: "Babyland"};
    Book book4 = { .price: 1936.1, .pageNumber: 1000, .language: UA, .weight: 2, .publicationYear: 2020, .name: "History of Ukraine"};
    Book book5 = { .price: 345, .pageNumber: 345, .language: US, .weight: 4, .publicationYear: 2000, .name: "cpp for beginers"};
    Book *books[] = { [0]: &book1, [1]: &book2, [2]: &book3, [3]: &book4, [4]: &book5};
    pushAll( data: books, length: sizeof(books) / sizeof(books[0]), l: list);

    Book *book55 = pop( l: list);
    Book *book44 = pop( l: list);
    printf( format: "Are both the same struct?: %d\n", book44 == &book4); // true
    pushAll( data: books, length: sizeof(books) / sizeof(books[0]), l: list);

    printList( toString: printBook, l: list);
    freeList( l: list);
}
```

- LinkedList.h

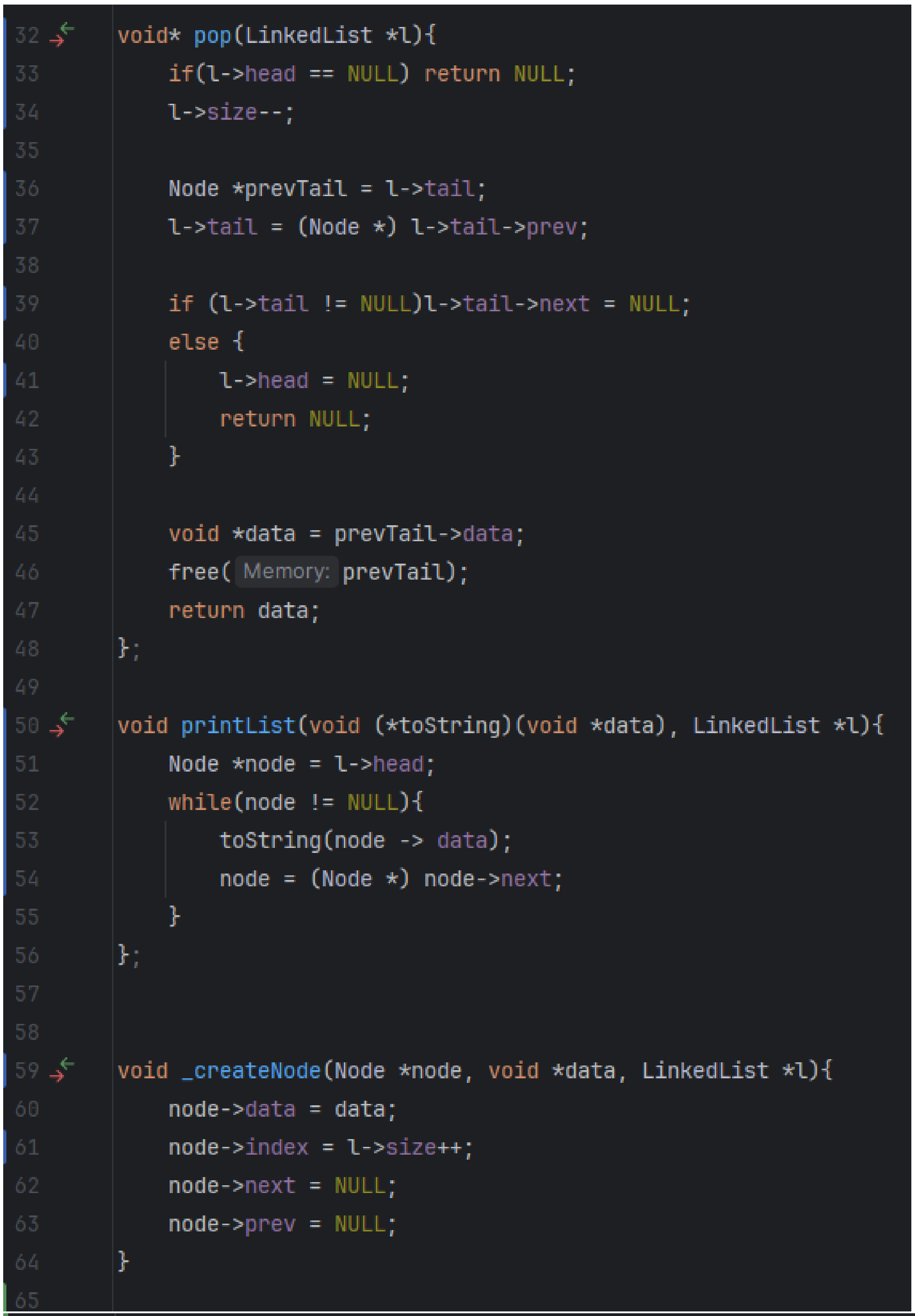


```c
#include <stdio.h>
#ifndef LINKED_LIST_H
#define LINKED_LIST_H

typedef struct {
    int index;
    struct Node *next;
    struct Node *prev;
    void *data;
} Node;

typedef struct {
    Node *head;
    Node *tail;
    int size;
} LinkedList;

LinkedList *createLinkedList();
void pushAll(void **data, int length, LinkedList *l);
void push(void *data, LinkedList *l);
void* pop(LinkedList *l);
void _createNode(Node *node, void *data, LinkedList *l);
void insert(Node *node, int i, LinkedList *l);
void delete(int i, LinkedList *l);
void freeList(LinkedList *l);
void printList(void (*toString)(void *data), LinkedList *l);

#endif
```

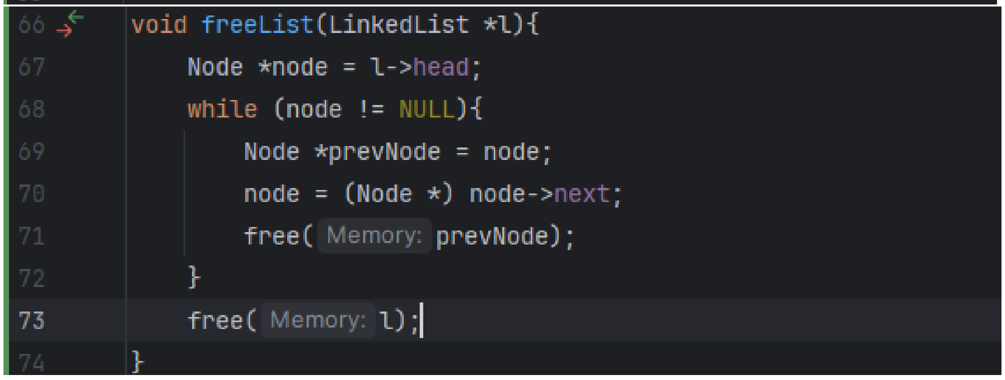
- LinkedList.c



```c
#include <malloc.h>
#include "linkedList.h"

LinkedList *createLinkedList(){
    LinkedList *linkedList = malloc(sizeof(LinkedList));
    linkedList->head = NULL;
    linkedList->tail = NULL;
    linkedList->size = 0;
    return linkedList;
}

void pushAll(void **data, int length, LinkedList *l){
    for (int i = 0; i < length; i++) {
        push(data[i], l);
    }
}

void push(void *data, LinkedList *l){
    if(l->head == NULL){
        l->head = (Node *)malloc(sizeof(Node));
        _createNode(l->head, data, l);
        l->tail = l->head;
        return;
    }
    Node *prev = l->tail;
    l->tail = (Node *)malloc(sizeof(Node));
    _createNode(l->tail, data, l);
    prev->next = l->tail;
    l->tail->prev = prev;
};

```

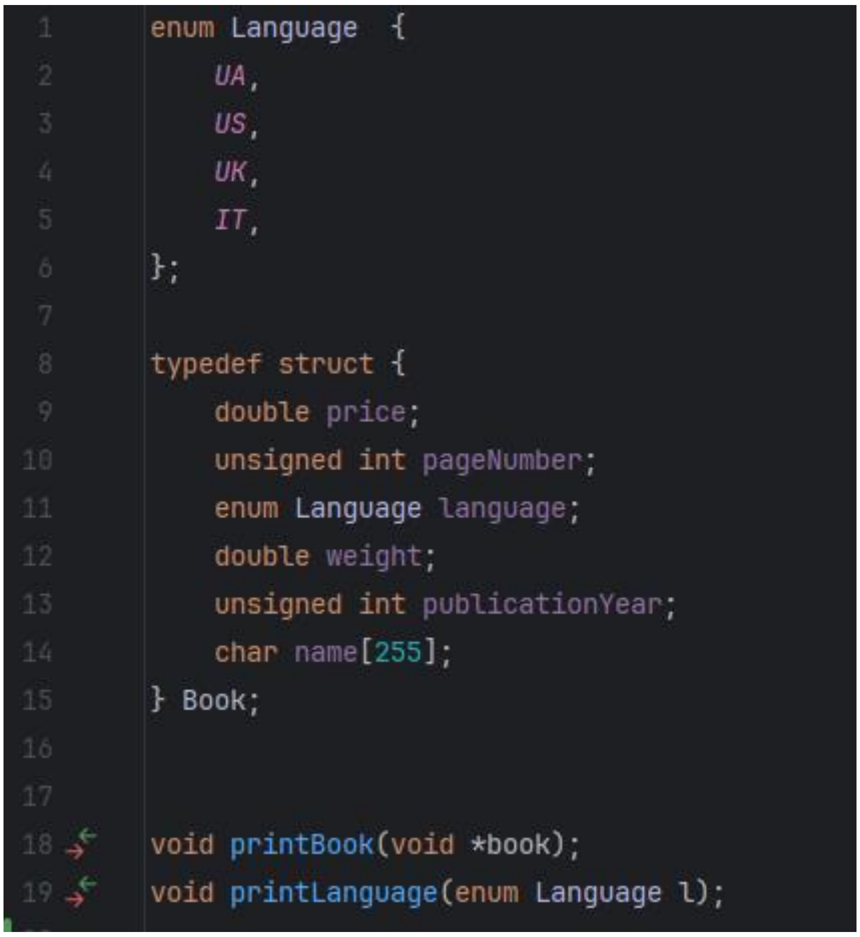


```c
void* pop(LinkedList *l){
    if(l->head == NULL) return NULL;
    l->size--;

    Node *prevTail = l->tail;
    l->tail = (Node *) l->tail->prev;

    if (l->tail != NULL)l->tail->next = NULL;
    else {
        l->head = NULL;
        return NULL;
    }

    void *data = prevTail->data;
    free(prevTail);
    return data;
};

void printList(void (*toString)(void *data), LinkedList *l){
    Node *node = l->head;
    while(node != NULL){
        toString(node -> data);
        node = (Node *) node->next;
    }
};


void _createNode(Node *node, void *data, LinkedList *l){
    node->data = data;
    node->index = l->size++;
    node->next = NULL;
    node->prev = NULL;
}

```



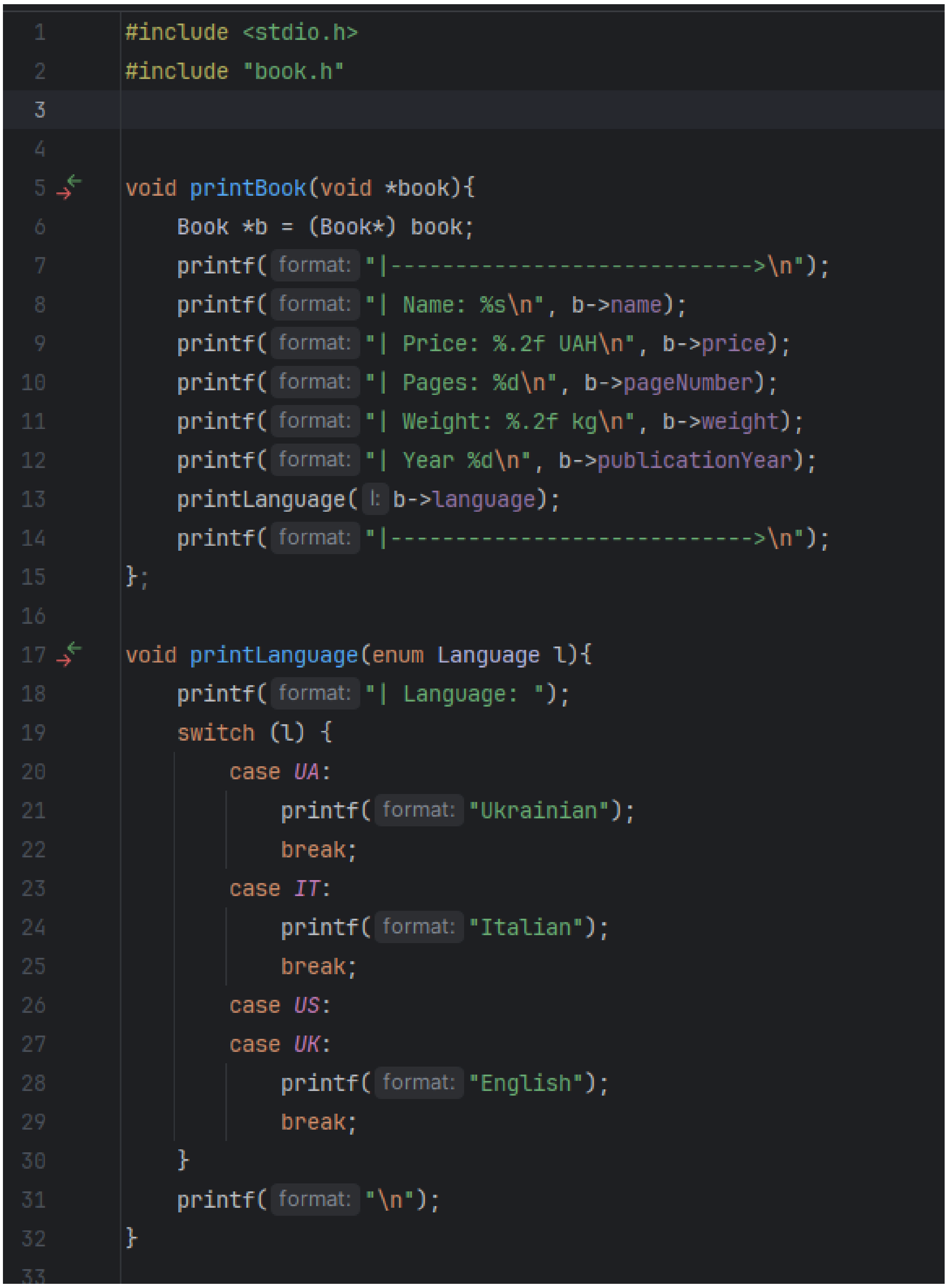
```c
void freeList(LinkedList *l){
    Node *node = l->head;
    while (node != NULL){
        Node *prevNode = node;
        node = (Node *) node->next;
        free(prevNode);
    }
    free(l);
}
```

- Book.h



```c
enum Language  {
    UA,
    US,
    UK,
    IT,
};

typedef struct {
    double price;
    unsigned int pageNumber;
    enum Language language;
    double weight;
    unsigned int publicationYear;
    char name[255];
} Book;


void printBook(void *book);
void printLanguage(enum Language l);
```

- Book.c



```c
#include <stdio.h>
#include "book.h"


void printBook(void *book){
    Book *b = (Book*) book;
    printf("|--------------------------->\n");
    printf("| Name: %s\n", b->name);
    printf("| Price: %.2f UAH\n", b->price);
    printf("| Pages: %d\n", b->pageNumber);
    printf("| Weight: %.2f kg\n", b->weight);
    printf("| Year %d\n", b->publicationYear);
    printLanguage(b->language);
    printf("|--------------------------->\n");
};

void printLanguage(enum Language l){
    printf("| Language: ");
    switch (l) {
        case UA:
            printf("Ukrainian");
            break;
        case IT:
            printf("Italian");
            break;
        case US:
        case UK:
            printf("English");
            break;
    }
    printf("\n");
}
```