Информатика

Кочанов Марк

13 октября 2018 г.

МИФИ

Структуры

Пример

```c
#include <string.h> // strcpy()

struct Book {
    int id;
    char title[50];
};

int main(void) {
    struct Book book;
    book.id = 6495407;
    strcpy(book.title, "C Programming");

    printf("Book 1 book_id: %d\n", book.id);
    printf("Book 1 title: %s\n", book.title);

    return 0;
}
```

Структуры

Структуры позволяют создавать пользовательские структуры данных. Как правило они используются для описания некоторой сущности из описываемой предметной области.

В рамках языка C структуры ведут себя аналогично встроенным типам данных. Возможна их передача в функцию, возврат структуры/указателя на структуру из функции, создание указателя на структуру, создание статического/динамического массива из структур и тд.

Примеры использования структур:

- длинные числа (знак числа + указатель на массив с разрядами + длина числа)
- дроби (числитель + знаменатель)
- матрицы (число строк + число столбцов + двумерный/одномерный массив с данными)

Передача структур функциям

```c
void print_book(struct Book book) {
    printf("Book book_id : %d\n", book.id);
    printf("Book title : %s\n", book.title);
}

void print_book_by_pointer(struct Book* book) {
    printf("Book id : %d\n", book->id);
    printf("Book title : %s\n", (*book).title);
    // (*book).title is equivalent to book->title
}

struct Book book;
book.id = 6495407;
strcpy(book.title, "C Programming");

print_book(book);
print_book_by_pointer(&book);
```

Передача структур функциям

Как правило, структуры передаются в функции по указателю, так как это позволяет избежать лишнего копирования структуры при её передаче в функцию.

При обращении к полям структура через указатель доступен оператор $->$, который аналогичен разыменованию указателя и последующего обращению к полю структуры.

«Короткий» синтаксис

```
1  struct Book { ... };
2
3  struct Book book;
4  void print_Book(struct Book book);
5  void print_Book_by_pointer(struct Book* book);
```

VS

```
1  // 1st variant
2  struct Book { ...};
3  typedef struct Book Book;
4
5  // 2nd variant
6  typedef struct Book { ... } Book;
7
8  // using
9  Book book;
10 void print_book(Book book);
11 void print_book_by_pointer(Book* book);
```

Задачи

Дроби

Реализовать функции для работы с дробями, заданными структурами.

```
1  typedef struct frac {
2      int num;
3      int denom;
4      // char sign
5  } frac;
6
7  frac frac_create(int num, int denom);
8  frac frac_sum(frac* f1, frac* f2);
9  frac frac_sub(frac* f1, frac* f2);
10 void frac_print(frac* f);
```

Скачать заголовочный файл.

Структура должна хранить дробь в нормализованном виде (числитель и знаменталь не имеют общих делителей кроме 1). По желанию знак дроби может храниться в числителе, либо отдельном поле структуры. Знаменатель должен быть положительным числом. При приведении дробей к общему знаменателю можно использовать код для вычисления НОД (НОК(a, b) = |a b| / НОД(a, b)).