

Программирование на C++



Минцифры
России

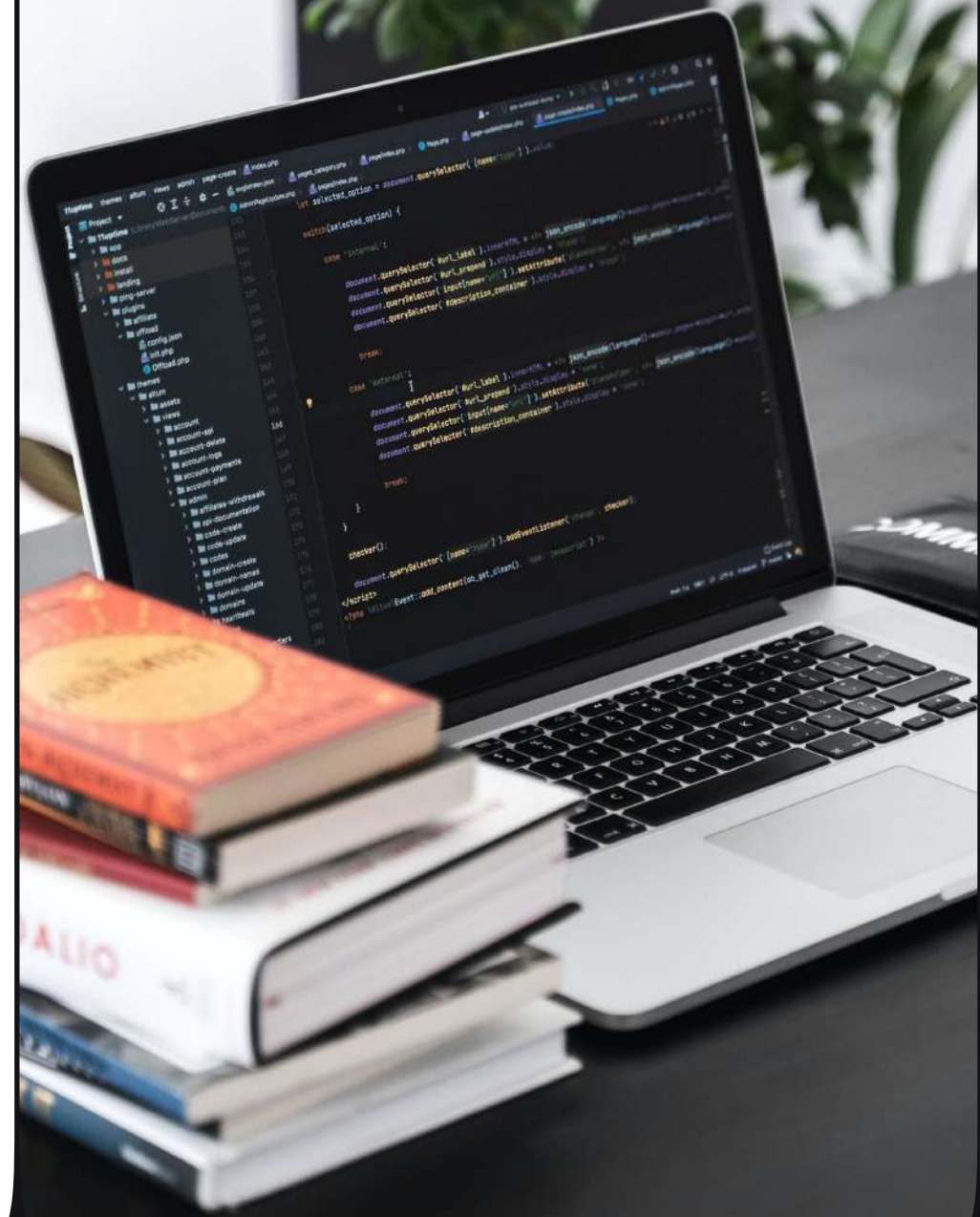
UCHi **DOMA**

20.35
УНИВЕРСИТЕТ

Урок 11 Модуль 2

Указатели на структуры

Полезные материалы



Программирование на C++



Минцифры
России

UCHi **DOMA**

20.35
УНИВЕРСИТЕТ

Цели урока



изучить указатели на структуры



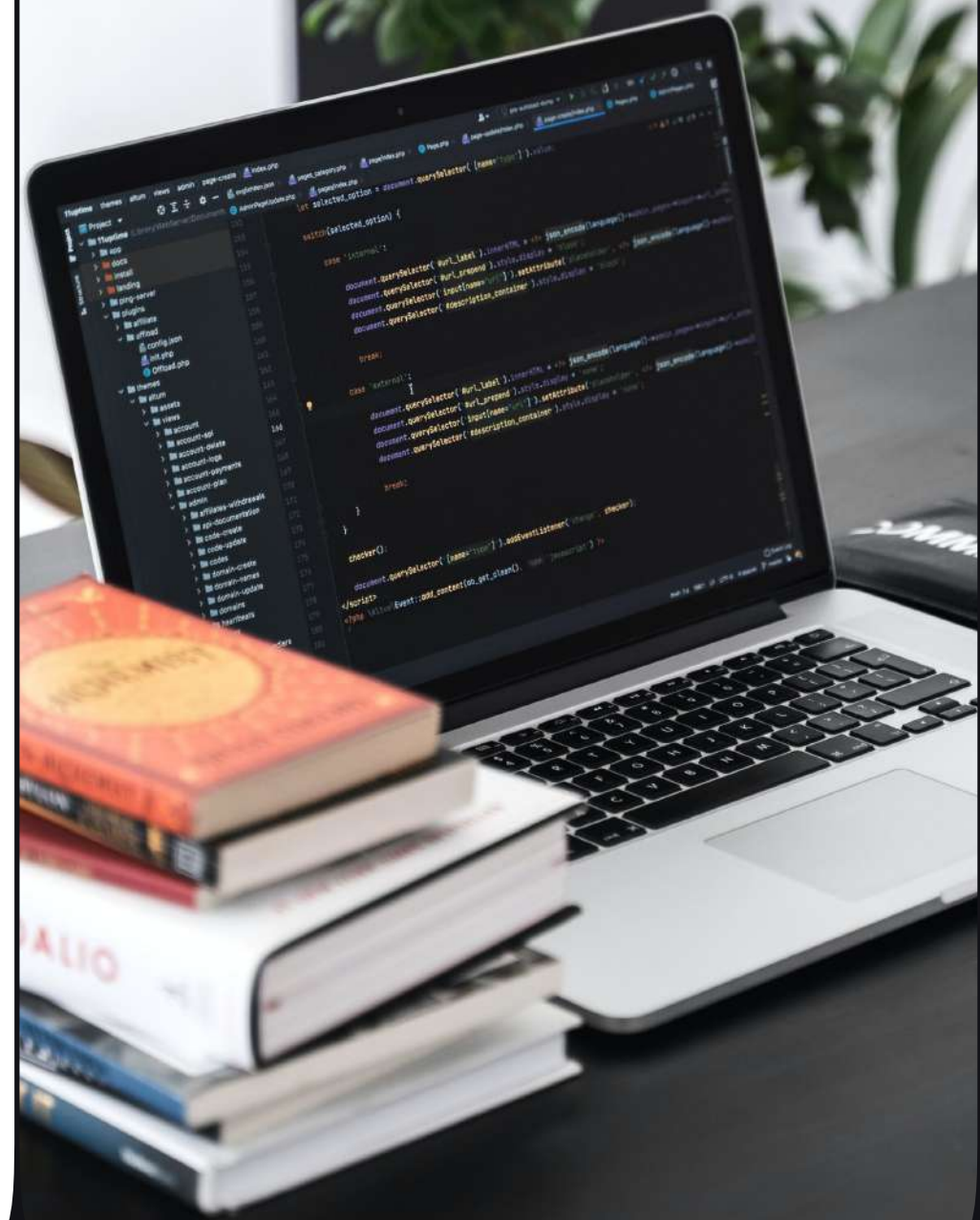
отработать на практике
составление алгоритмов
с указателями на структуры
на Си



Урок 9 Модуль 2

Структуры и функции

Полезные материалы



Указатели на структуры

Доступ к элементам структуры или объединения можно осуществить с помощью указателей. Для этого необходимо инициализировать указатель на адрес структуры или объединения.

Для организации работы с массивом можно использовать указатель. При этом обращение к полям структуры через указатель будет выглядеть как:

указатель->поле

или

(*указатель).поле



указатель — указатель на структуру или объединение;

поле — поле структуры или объединения;

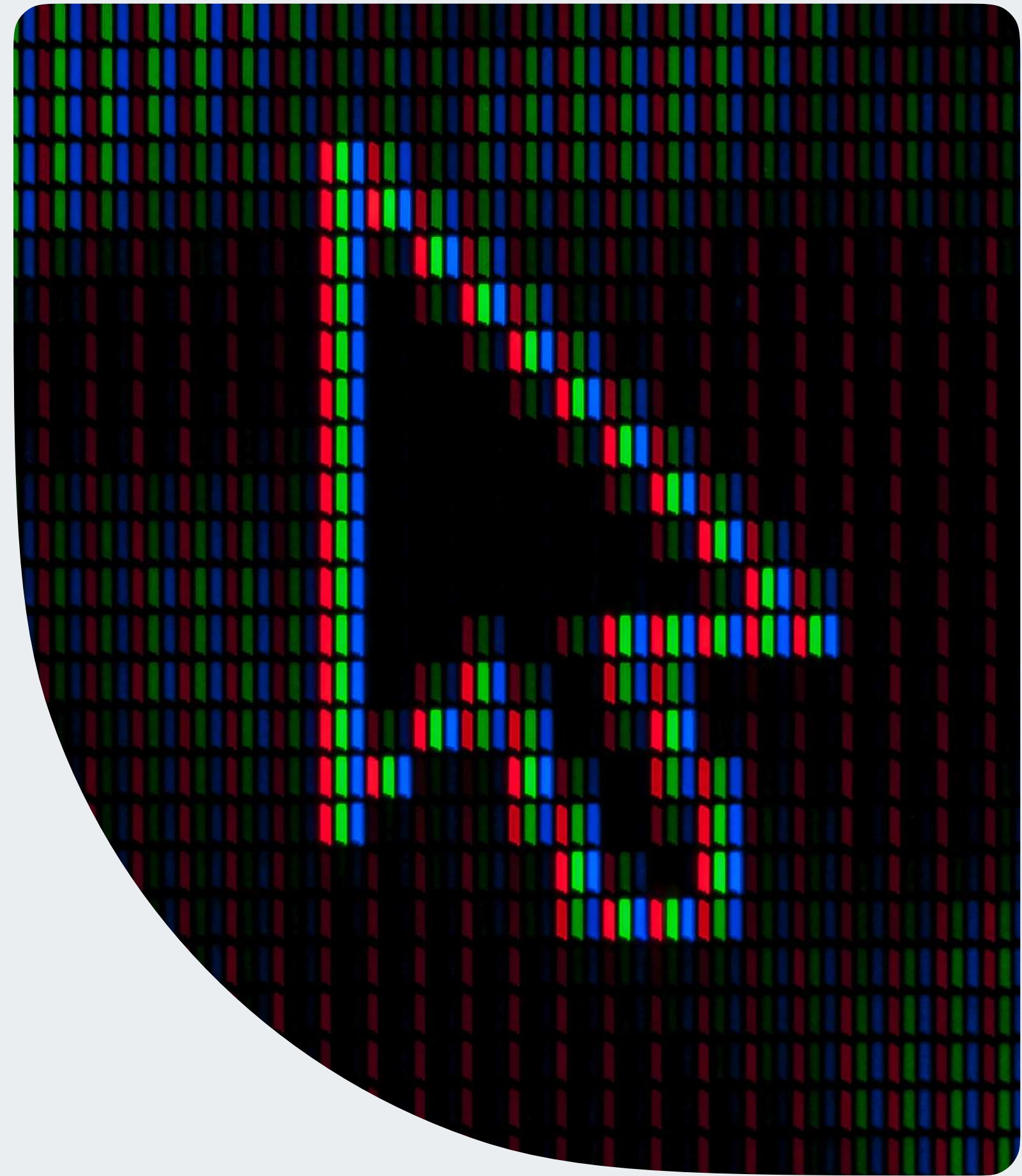
Цели урока



изучить использование структур
как параметров функций



отработать на практике
составление алгоритмов
с передачей структур
как параметров функций на Си



Пример



```
1  #include <stdlib.h>
2  #include <malloc.h>
3  struct book
4  {
5      char title[15];
6      char author[15];
7      int value;
8  };
9  int main()
10 {
11     struct book lib={"Сказки", "Пушкин", 100};
12     struct book *ptr_lib=&lib;
13     printf("%s", ptr_lib->title);
14     return 0;
15 }
```

Результат работы программы:

Сказки

Структура

Структура может использоваться в качестве параметра функции и также может быть возвращаемым объектом функции.

Динамическое выделение памяти для структур



Если заранее неизвестен размер массива, то необходимо Динамически выделять память под массив структур.



Для определения размера структуры в байтах используется операция `sizeof(ИмяСтруктуры)`.

Пример



Задача:

Написать программу для хранения координат точки в трехмерном пространстве и изменения положения точки в пространстве. Программа должна запрашивать начальные координаты, изменение координат и выводить на экран новые координаты.

Создадим структуру для хранения координат и изменения координат:

```
struct position
{
    int x;
    int y;
    int z;
};
```

Пример



```
1  #include <stdlib.h>
2  #include <malloc.h>
3  struct book
4  {
5      char title[15];
6      char author[15];
7      int value;
8  };
9  int main()
10 {
11     struct book *lib;
12     int i;
13     lib = (struct book*)malloc(3 * sizeof(struct book));
14     for (i = 0; i<3; i++)
15     {
16         printf("Введите название %d книги : ", i + 1);
17         scanf("%s", (lib + i)->title);
18         printf("Введите автора %d книги : ", i + 1);
19         scanf("%s", (lib + i)->author);
20         printf("Введите цену %d книги : ", i + 1);
21         scanf("%d", &(lib + i)->value);
22     }
23     for (i = 0; i<3; i++)
24     {
25         printf("\n %d. %s ", i + 1, (lib + i)->author);
26         printf("%s %d", (lib + i)->title, (lib + i)->value);
27     }
28     return 0;
29 }
```

Пример



Создадим объекты структуры position:

```
struct position point;  
struct position change;
```

Запросим ввод значений с клавиатуры:

```
struct position point;  
struct position change;  
printf("Введите координаты: x y z:");  
scanf("%d%d%d",&point.x,&point.y,&point.z);  
printf("Введите изменение координат: x y z:");  
scanf("%d%d%d",&change.x,&change.y,&change.z);
```


Пример



Библиотека из 3 книг

Результат работы программы:

Введите название 1 книги : Стихи

Введите автора 1 книги : Пушкин

Введите цену 1 книги : 100

Введите название 2 книги : Басни

Введите автора 2 книги : Крылов

Введите цену 2 книги : 150

Введите название 3 книги : Роман

Введите автора 3 книги : Тургенев

Введите цену 3 книги : 200

1. Пушкин Стихи 100

2. Крылов Басни 150

3. Тургенев Роман 200

Пример



Создадим функцию `move`, которая будет получать в качестве параметров два объекта структуры `position` и возвращать уже новый объект структуры `position`.

```
struct position move(struct position p, struct position d)
{
    struct position new_p = {};
    new_p.x = p.x + d.x;
    new_p.y = p.y + d.y;
    new_p.z = p.z + d.z;
    return new_p;
}
```

Пример



Вызовем функцию `move`, передадим ей в качестве параметров начальные координаты и изменение координата. Результат работы функции сохраним в `new_point`

```
struct position new_point = move(point, change);
```

Пример



Выведем новые координаты:

```
printf("x=%d y=%d z=%d \n", new_point.x, new_point.y, new_point.z);
```

Пример



```
1  #include <stdio.h>
2  struct position
3  {
4      int x;
5      int y;
6      int z;
7  };
8  struct position move(struct position p, struct position d)
9  {
10     struct position new_p ={};
11     new_p.x=p.x+d.x;
12     new_p.y=p.y+d.y;
13     new_p.z=p.z+d.z;
14     return new_p;
15 }
16
17 int main(void)
18 {
19     struct position point;
20     struct position change;
21     printf("Введите координаты: x y z:");
22     scanf("%d%d%d",&point.x,&point.y,&point.z);
23     printf("Введите изменение координат: x y z:");
24     scanf("%d%d%d",&change.x,&change.y,&change.z);
25     struct position new_point = move(point, change);
26     printf("x=%d y=%d z=%d \n",new_point.x,new_point.y,new_point.z);
27     return 0;
28 }
```


Практика



Напишите программу, которая будет хранить время в структуре (часы, минуты, секунды).



Программа должна запрашивать ввод времени с клавиатуры, а также изменение времени в минутах.



Программа должна вывести новое время с учетом изменения.

Пример работы программы:

Часы:11

Минуты:59

Секунды:10

Сколько добавить минут:3

12:2:10

Решение:

```
1  #include <stdio.h>
2  struct time
3  {
4      int hour;
5      int minute;
6      int second;
7  };
8  struct time addminutes(struct time, int); // прототип функции
9
10 int main(void)
11 {
12     int minutes;
13     struct time current_time;
14     printf("Часы:");
15     scanf("%d",&current_time.hour);
16     printf("Минуты:");
17     scanf("%d",&current_time.minute);
18     printf("Секунды:");
19     scanf("%d",&current_time.second);
20     printf("Сколько добавить минут:");
21     scanf("%d",&minutes);
22
23     struct time result_time = addminutes(current_time, minutes);
24     printf("%d:%d:%d \n", result_time.hour, result_time.minute, result_time.second);
25     return 0;
26 }
27 struct time addminutes(struct time t, int minutes)
28 {
29     struct time result = {t.hour, t.minute, t.second};
30     int h, d;
31     result.minute += minutes; // прибавляем минуты
32     if(result.minute >= 60) // если минут больше 59, делим на 60
33     {
34         h = result.minute / 60;
35         result.minute -= 60 * h; // от минут вычитаем 60 * h
36         result.hour += h; // к часам прибавляем h
37     }
38     if(result.hour >= 24) // если часов больше 23, делим на 24
39     {
40         d = result.hour / 24; // от часов вычитаем d * 24
41         result.hour -= 24 * d;
42     }
43     return result;
44 }
```