Программирование на C++

Минцифры России

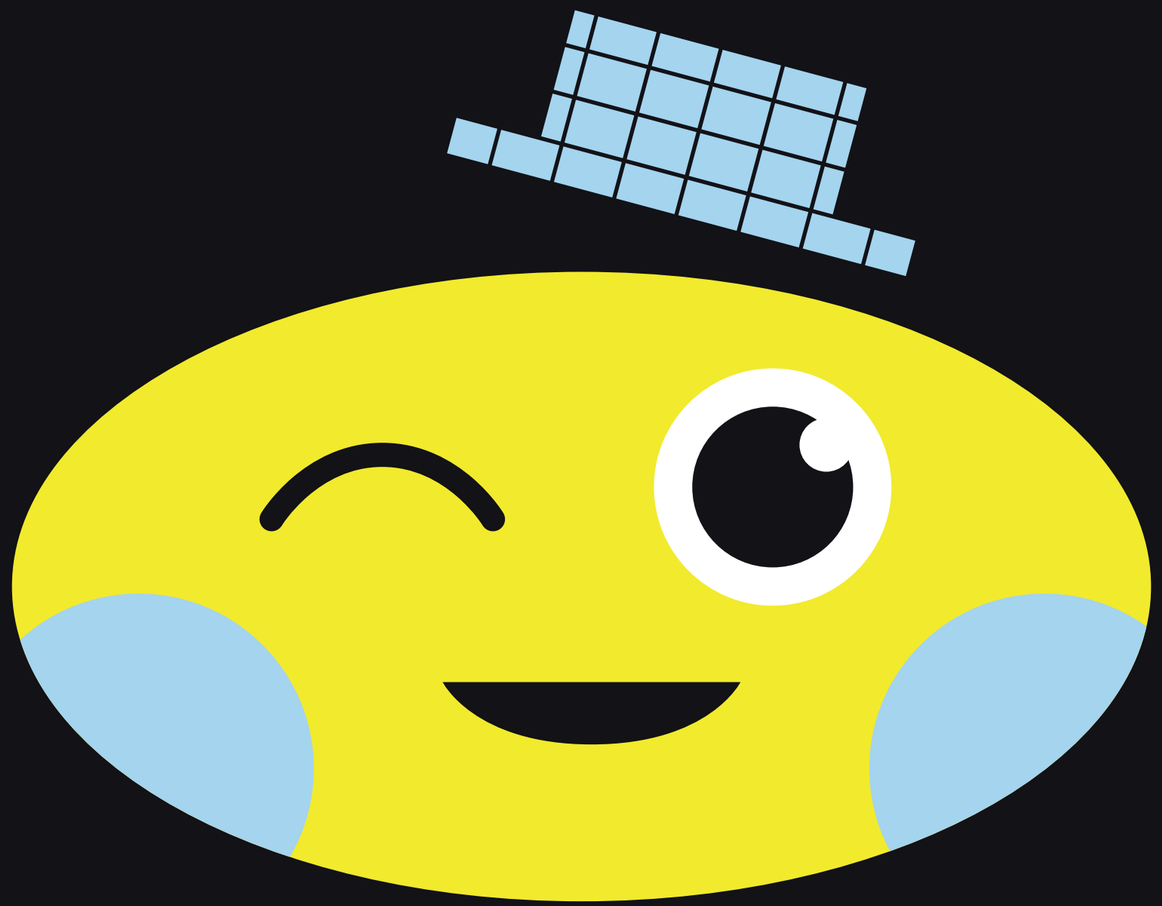
UCHi DOMA

20.35 УНИВЕРСИТЕТ

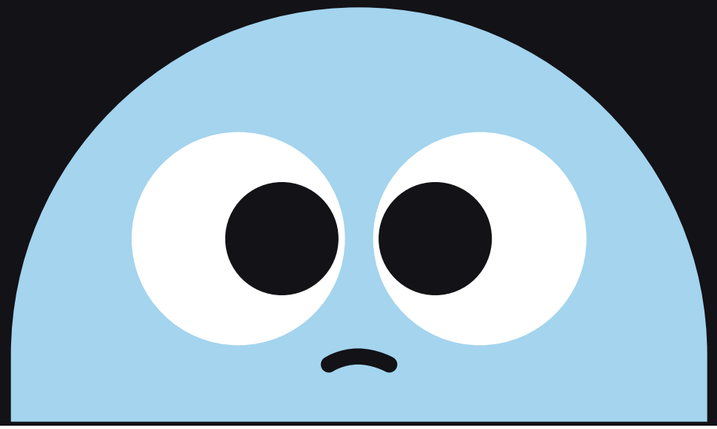
Модуль 2. Урок 7

Указатели как аргументы функций

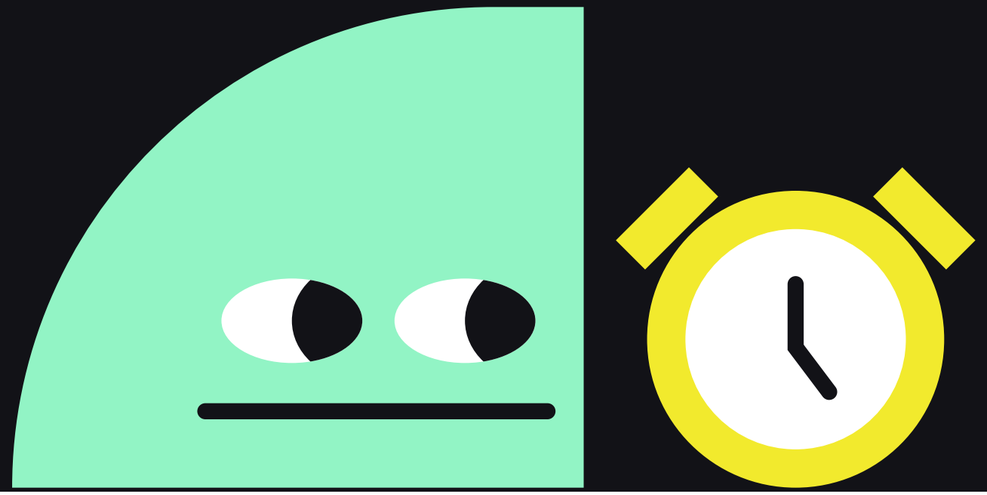
Привет!



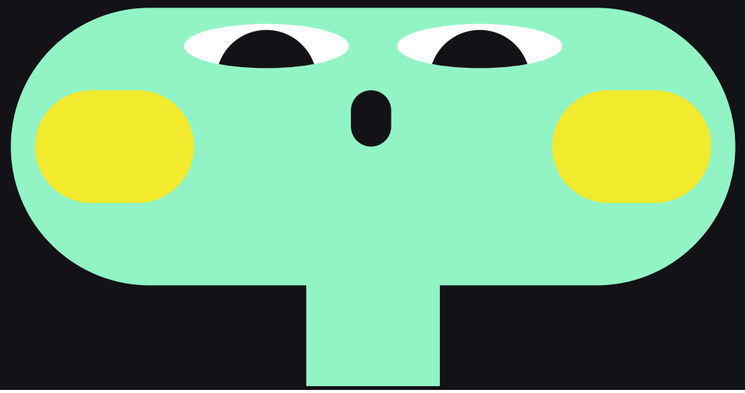
проверка готовности



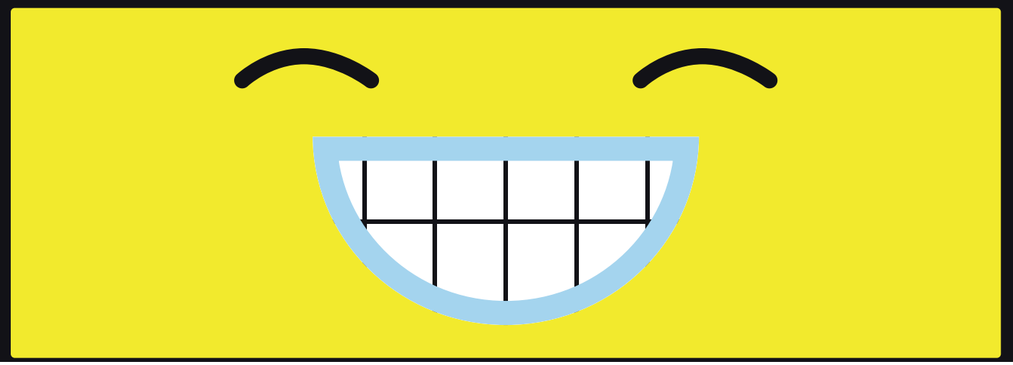
Видим и слышим друг друга без помех



Не опаздываем и не отвлекаемся



Сидим прямо



Улыбаемся, если всё ок

Как домашка?

- Какие были трудности?
- Какие остались вопросы?
- Сколько заданий выполнено?

Разомнёмся

Что будет выведено на экран
в результате работы программы?

```c
#include <stdio.h>
int main()
{
  int a=100;
  int* p_a=&a;
  printf("%d\n",*p_a-a);
  return 0;
}

```

Разомнёмся

Что будет выведено на экран
в результате работы программы?

```c
#include <stdio.h>
int main()
{
  int a=100;
  int* p_a=&a;
  printf("%d\n",*p_a-a);
  return 0;
}

```

Результат работы программы

```
0
```

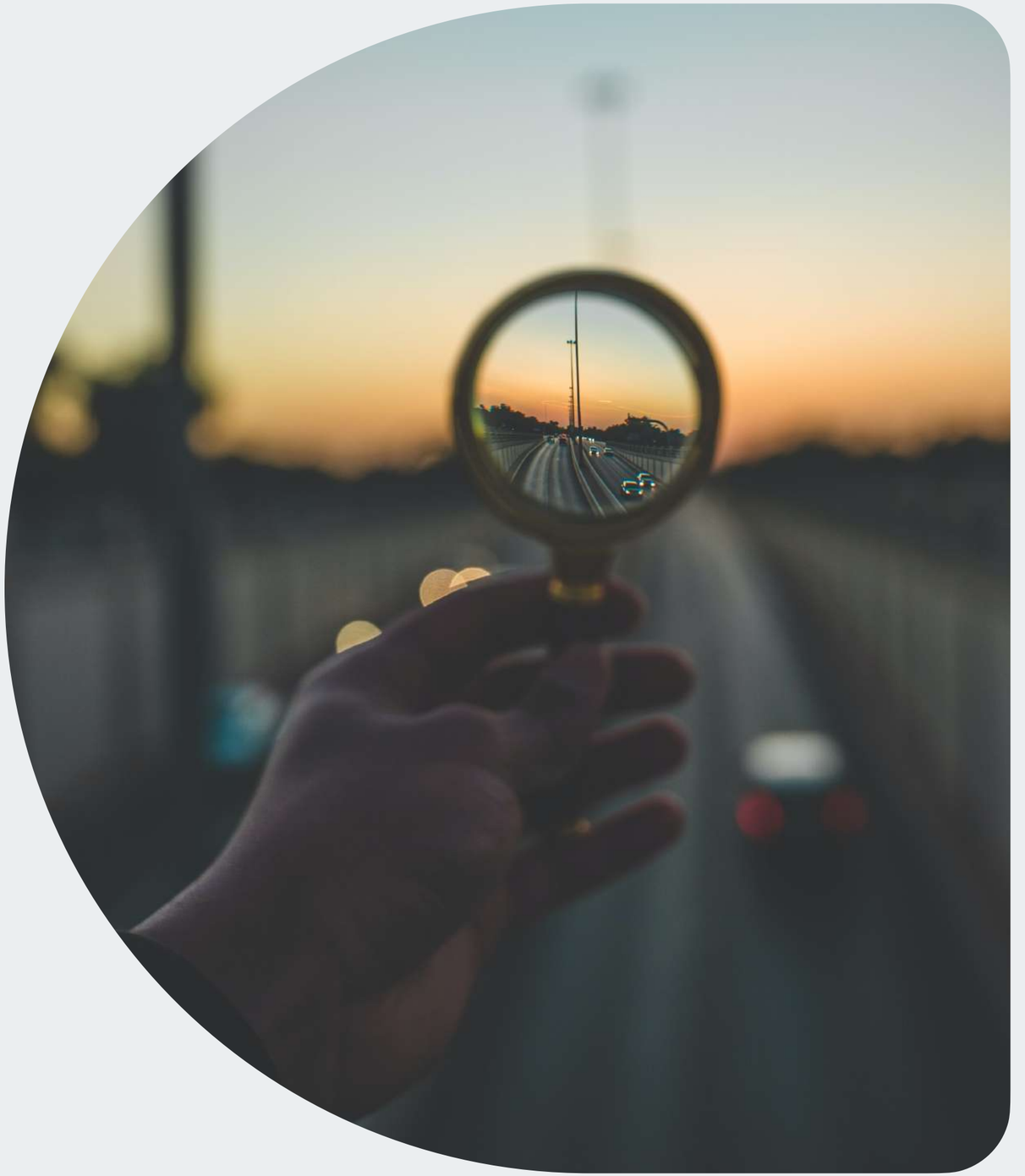
Вопрос

Может ли указатель быть аргументом функции?

Будет ли отличатся результат работы функции при передаче в качестве аргумента переменной и указателя?

Цели урока

- изучить работу с указателями как аргументами функции
- отработать на практике составление алгоритмов с использованием указателей как аргументов функции на Си



Задача

Напишем функцию func() которая будет увеличивать получаемый аргумент в 2 раза

```c
#include <stdio.h>
void func(int);

int main()
{
    int n = 10;
    func(n);
    printf("main function: %d\n",n);
    return 0;
}
void func(int x)
{
    x*=2;
    printf("func function: %d\n",x );
}
```

Задача

Напишем функцию func() которая будет увеличивать получаемый аргумент в 2 раза

```c
#include <stdio.h>
void func(int);

int main()
{
    int n = 10;
    func(n);
    printf("main function: %d\n",n);
    return 0;
}
void func(int x)
{
    x*=2;
    printf("func function: %d\n",x );
}
```

Результат работы программы

```
func function: 20
main function: 10
```

Задача

Напишем туже функцию, только будем передавать в качестве параметра указатель

```c
#include <stdio.h>
void func(int*);

int main()
{
    int n = 10;
    func(&n);
    printf("main function: %d\n",n);
    return 0;
}
void func(int* x)
{
    (*x)*=2;
    printf("func function: %d\n",*x );
}
```

Задача

Напишем туже функцию, только будем передавать в качестве параметра указатель

```c
#include <stdio.h>
void func(int*);

int main()
{
    int n = 10;
    func(&n);
    printf("main function: %d\n",n);
    return 0;
}
void func(int* x)
{
    (*x)*=2;
    printf("func function: %d\n",*x );
}
```

Результат работы программы

```
func function: 20
main function: 20
```

Задача

1

Для изменения значения параметра применяется операция разыменования с последующим умножением на 2:

```
(*x)*=2;
```

2

Это изменяет значение, которое находится по адресу, хранимому в указателе x.

3

Поскольку теперь функция в качестве параметра принимает указатель, то при ее вызове необходимо передать адрес переменной:

```
func(&n);
```

4

В итоге изменение параметра x также повлияет на переменную n:

```
func function: 20
main function: 20
```

Задача

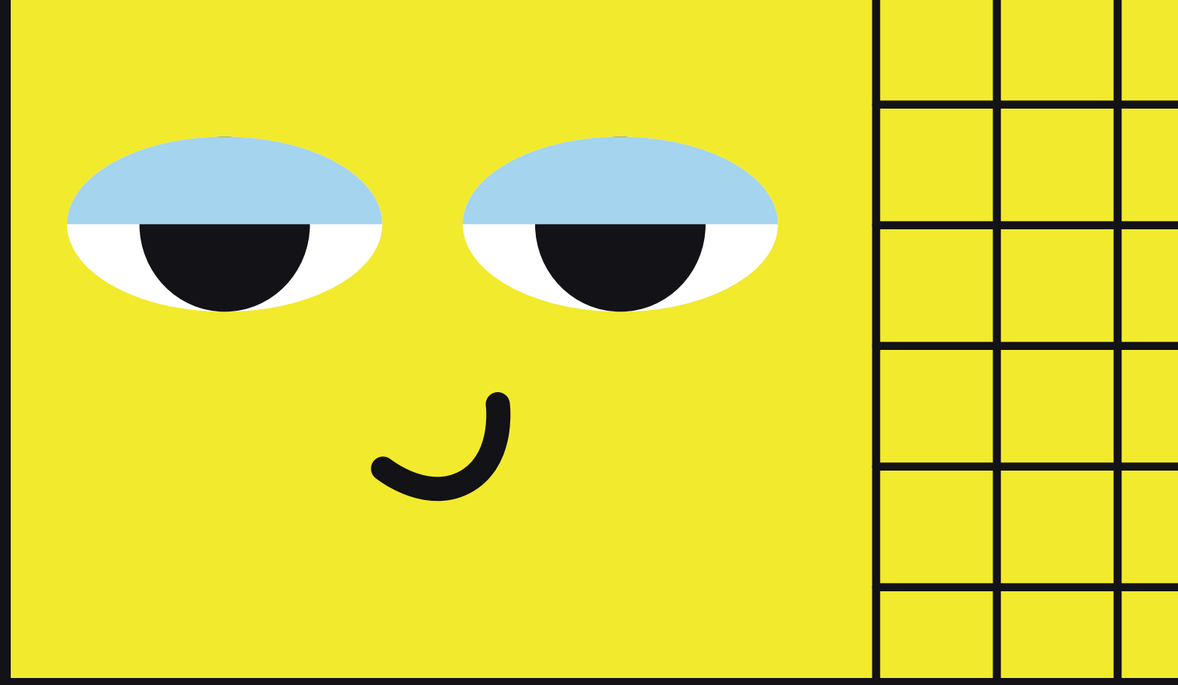
```c
#include <stdio.h>
void func(int*);

int main()
{
    int n = 10;
    int *ptr = &n;
    func(ptr);
    printf("main function: %d\n",n);
    return 0;
}
void func(int* x)
{
    int z = *x;
    x = &z;
    (*x)*=2;
    printf("func function: %d\n",*x );
}
```

Аргумент передается в функцию по значению, то есть функция получает копию адреса, если внутри функции будет изменен адрес указателя, то это не затронет внешний указатель, который передается в качестве аргумента:

Результат работы программы

```
func function: 20
main function: 10
```

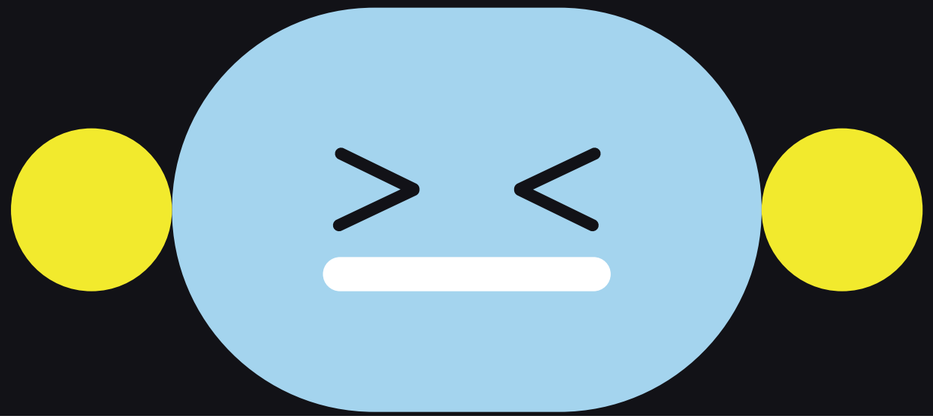


Практика

перерыв физкультминутка

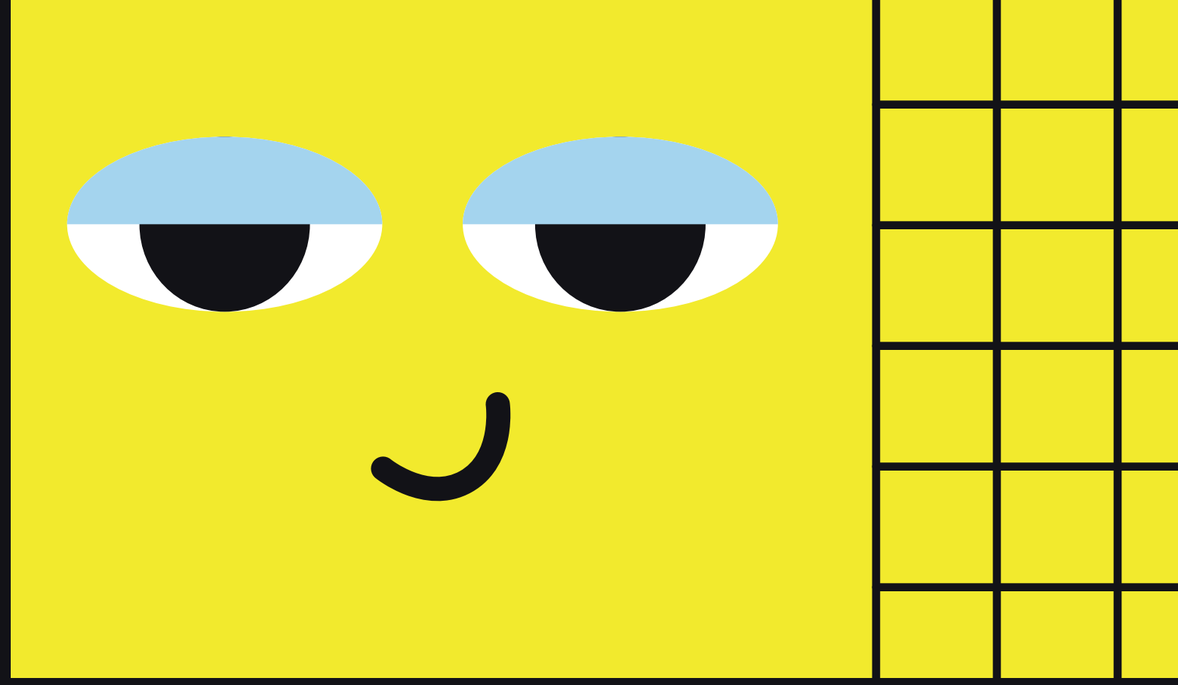
Смотрим вверх–вниз, вправо–влево

Вращаем по кругу туда–обратно



Крепко зажмуриваемся

Быстро моргаем



Практика

Закрепление

Что будет выведено на экран
в результате работы программы?

```c
#include <stdio.h>
void func(int*);

int main()
{
    int n = 10;
    func(&n);
    printf("%d\n",n);
    return 0;
}
void func(int* x)
{
    (*x)++;
}
```

Закрепление

Что будет выведено на экран
в результате работы программы?

```c
#include <stdio.h>
void func(int*);

int main()
{
    int n = 10;
    func(&n);
    printf("%d\n",n);
    return 0;
}
void func(int* x)
{
    (*x)++;
}
```

Результат работы программы

```
11
```

Закрепление

Что будет выведено на экран
в результате работы программы?

```c
#include <stdio.h>
void func(int*);

int main()
{
    int n = 10;
    int* ptr=&n;
    func(&n);
    printf("%d\n",n);
    return 0;
}
void func(int* x)
{
    (*x)++;
}
```

Закрепление

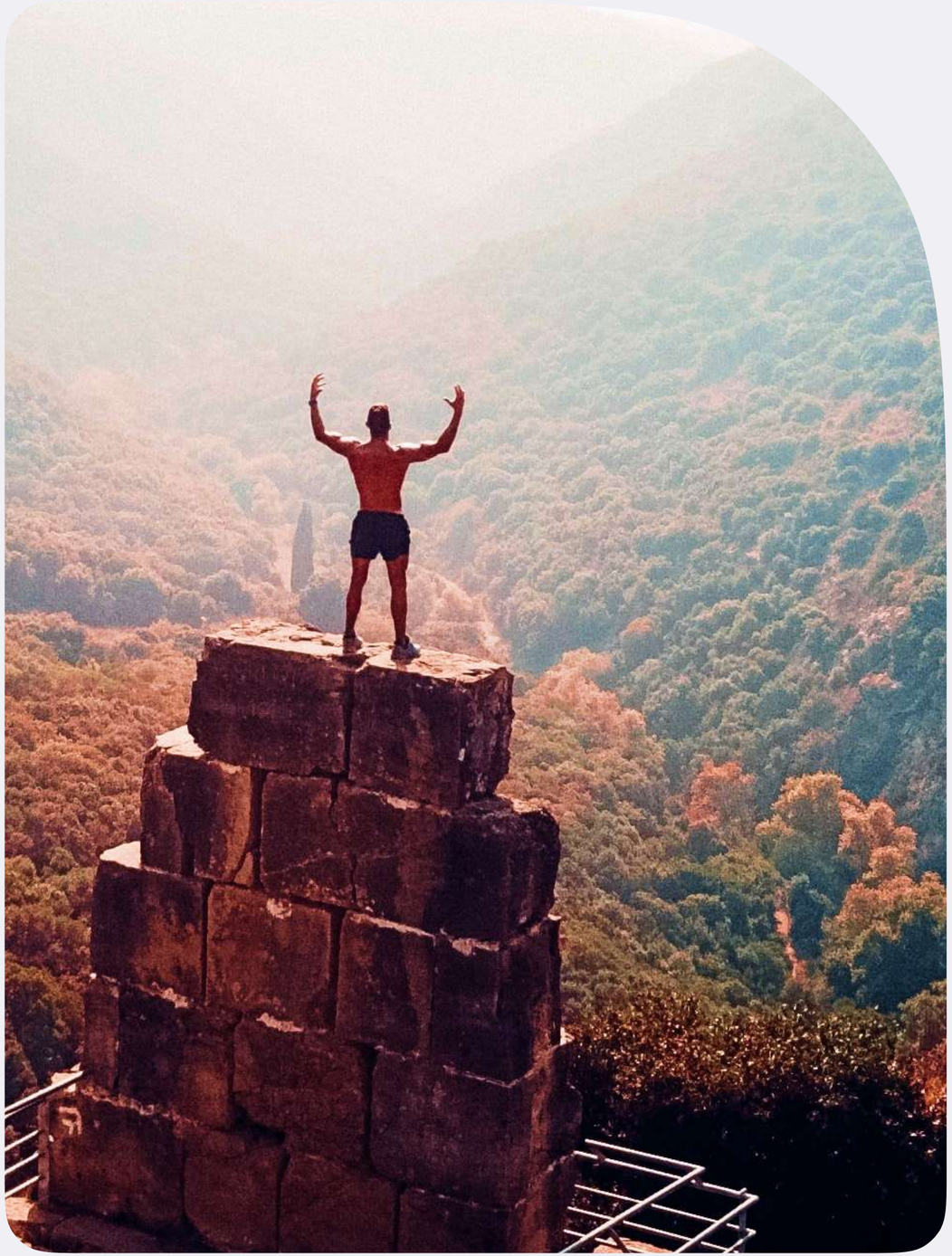
Что будет выведено на экран
в результате работы программы?

```c
#include <stdio.h>
void func(int*);

int main()
{
    int n = 10;
    int* ptr=&n;
    func(&n);
    printf("%d\n",n);
    return 0;
}
void func(int* x)
{
    (*x)++;
}
```

Результат работы программы

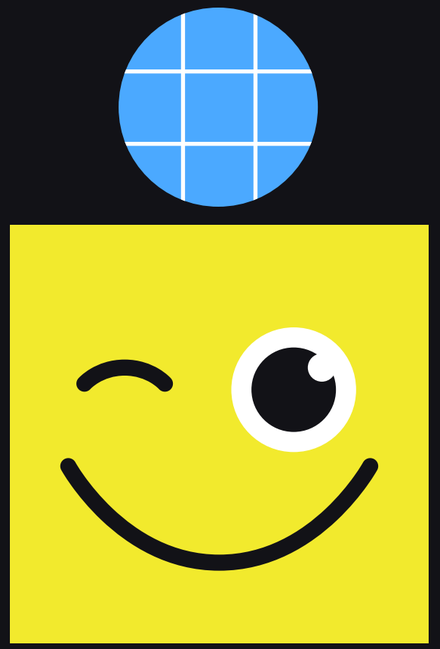
```
11
```



Подведём итоги

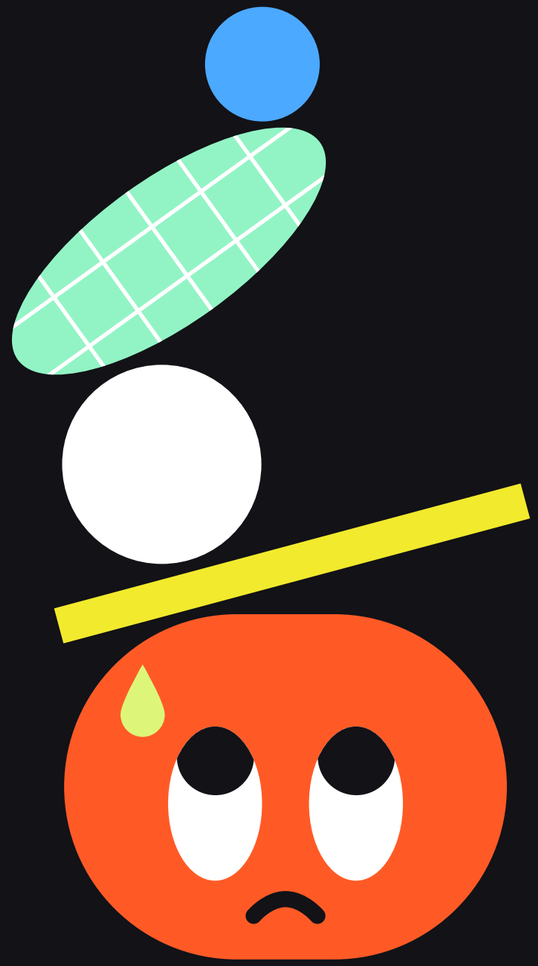
- изучили работу с указателями как аргументами функции
- отработали на практике составление алгоритмов с использованием указателей как аргументов функции на Си

итоги урока



1 легко!

2 сложно, но можно



3 трудно

Домашнее задание

До встречи!