



Лекция №1

Вводный урок

Продвинутый курс Си






План курса

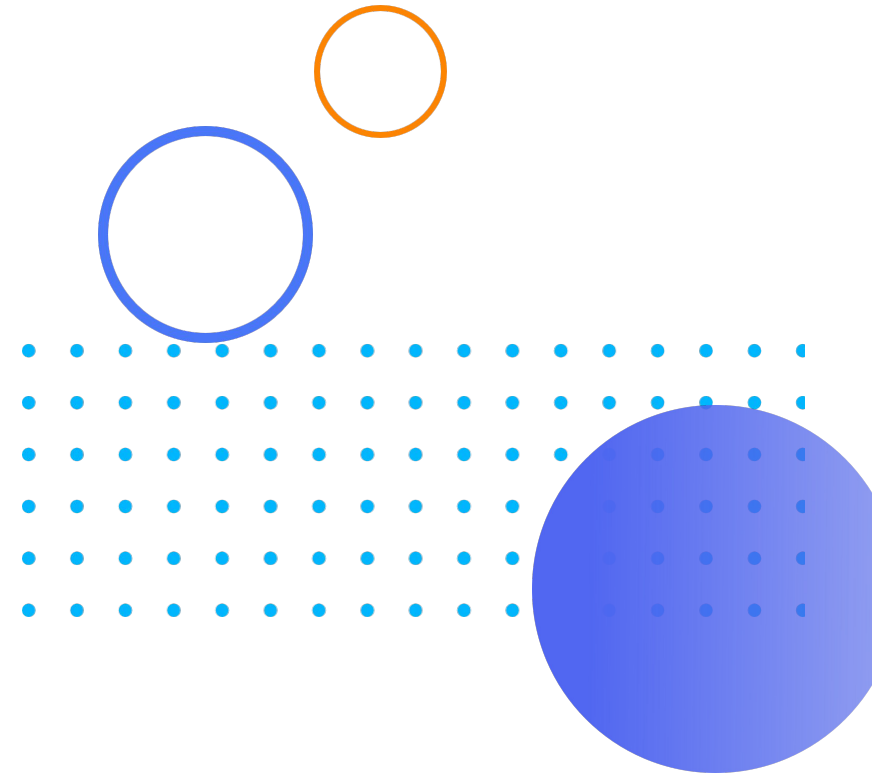
- Вводный урок
- Структуры. Динамические типы
- Библиотеки языка C
- Оптимизация кода
- Алгоритмы
- Компиляция и компиляторы
- Динамические структуры данных
- Курсовая работа



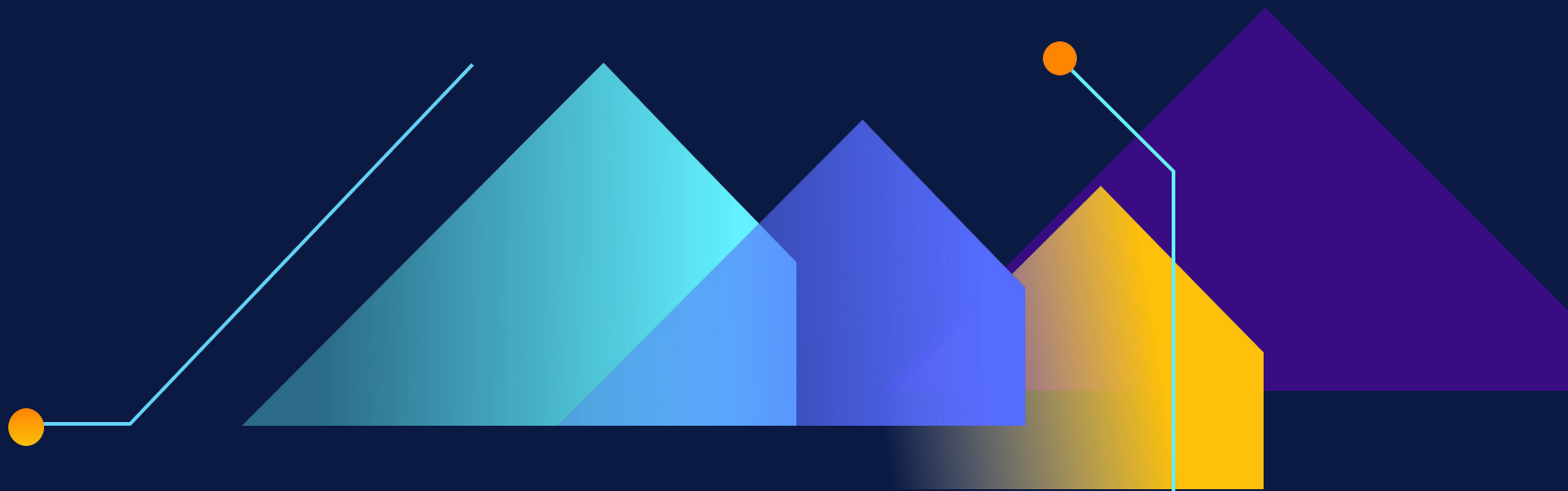
Маршрут

Повторение основного курса С

-  Повторим побитовые операции
-  Повторим структуры с побитовыми полями
-  Повторим массивы, структуры и функции



Побитовые операции





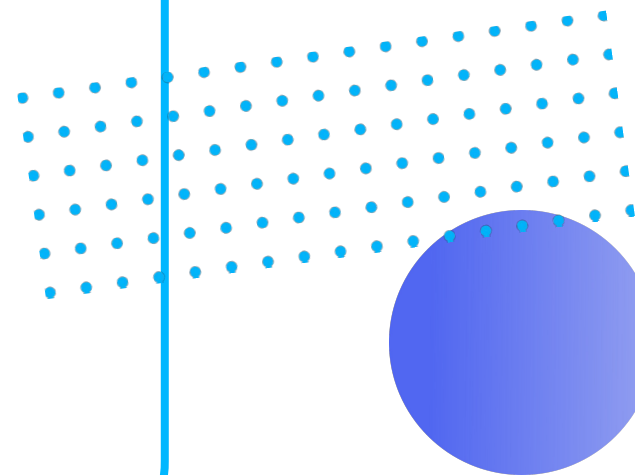
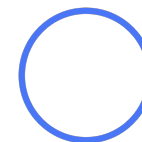
Побитовые операции

Побитовые операции работают над двоичным представлением чисел. Существуют одномерные:

- \sim — побитовая инверсия

и двумерные операции:

- $\&$ — побитовое И
- $|$ — побитовое ИЛИ
- \ll — побитовый сдвиг влево
- \gg — побитовый сдвиг вправо
- \wedge — побитовое исключающее ИЛИ (XOR)



Поразрядные и логические операторы

Внимание! Поразрядные операторы не следует использовать вместо логических. Результатом логических операторов (&&, || и !) является либо 0, либо 1, но побитовые операторы возвращают целочисленное значение.

Кроме того, логические операторы рассматривают любой ненулевой операнд как 1. Рассмотрим следующую программу: результаты & и && различны для одних и тех же операндов.

```
int32_t x = 3, y = 8;  
(x & y) ? printf("True ") : printf("False ");  
(x && y) ? printf("True ") : printf("False ");
```

False True

Побитовый сдвиг

Необходимо помнить, что сдвиг на величину, превышающую размер правого операнда, или сдвиг на отрицательное число разрядов не определён.

При сдвиге вправо результат операции зависит от знаковости операнда. Если операнд имеет беззнаковый тип, то слева выдвигаются нулевые биты.

Такой сдвиг иногда называют арифметическим.

Когда операнд знаковый, то слева выдвигается самый старший бит — бит знака. В том случае, когда этот бит равен 1, то выдвигается единица, а если бит знака равен 0, то выдвигается ноль. Такой вид сдвига называют логическим.

Внимание! Сдвиг влево и вправо на 1 эквивалентен умножению и делению на 2.

Примеры побитового сдвига

```
uint8_t u = 0xF5; //беззнаковый тип  
u >>= 1; // сдвиг вправо на 1 бит  
printf("u = %" PRIx8 "\n", u);
```

u = 7a

```
int8_t u = 0xF5; //знаковый тип  
u >>= 1; // сдвиг вправо на 1 бит  
printf("u = %" PRIx8 "\n", u);
```

u = fa

Операция исключающее ИЛИ - XOR

Побитовый оператор XOR — весьма полезный оператор, он используется во многих задачах. Простым примером может быть следующая задача: дан массив чисел, где все элементы встречаются четное количество раз, кроме одного числа, найдите нечётное встречающееся число. Эту проблему можно эффективно решить, просто выполнив операцию XOR для всех чисел.

```
int find_odd_element(int32_t arr[], size_t n) {  
    int32_t res = 0;  
    for (size_t i = 0; i < n; i++)  
        res ^= arr[i];  
    return res;  
}  
  
int main(void) {  
    int32_t arr[] = { 17, 17, 24, 97, 24, 24, 24 };  
    size_t n = sizeof(arr) / sizeof(arr[0]);  
    printf("The element is %" PRId32,  
          find_odd_element(arr, n));  
    return 0;  
}
```

The element is
97

Побитовое И

Операция побитовое И может быть использована для быстрой проверки чётности числа.

```
int32_t x = 17;  
(x & 1) ? printf("Odd") : printf("Even");
```

Odd

Примеры

```
uint32_t a = 60;    /* 60 = 0011 1100 */
uint32_t b = 13;    /* 13 = 0000 1101 */
int32_t c = 0;

c = a & b;          /* 12 = 0000 1100 */
printf("Line 1 c = %d\n", c );

c = a | b;          /* 61 = 0011 1101 */
printf("Line 2 c = %d\n", c );

c = a ^ b;          /* 49 = 0011 0001 */
printf("Line 3 c = %d\n", c );

c = ~a;             /* -61 = 1100 0011 */
printf("Line 4 c = %d\n", c );

c = a << 2;          /* 240 = 1111 0000 */
printf("Line 5 c = %d\n", c );

c = a >> 2;          /* 15 = 0000 1111 */
printf("Line 6 c = %d\n", c );
```

Line 1 c = 12

Line 2 c = 61

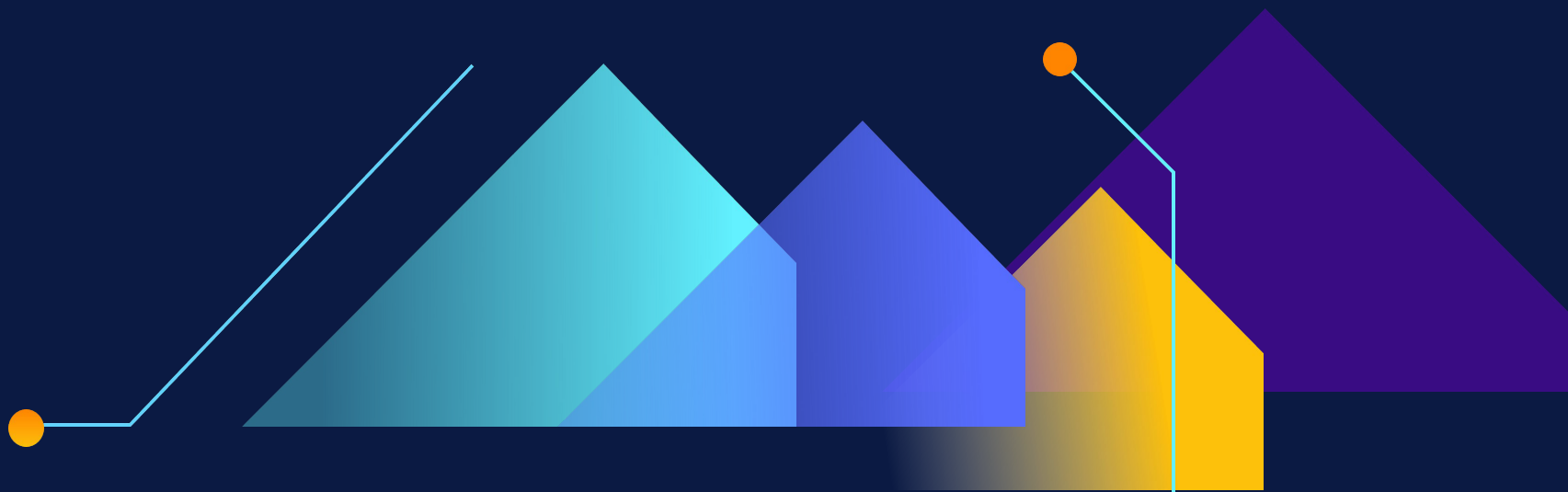
Line 3 c = 49

Line 4 c = -61

Line 5 c = 240

Line 6 c = 15

Структуры с битовыми полями



Пример

Предположим есть некоторое количество переменных, упакованных в структуру для хранения значений True/False.

Такая структура будет занимать в памяти 8 байт, хотя на самом деле использоваться будет только два бита для хранения 0 и 1.

```
/* Определение структуры */  
struct {  
    uint32_t width;  
    uint32_t height;  
} st1;
```

Пример

При использовании полей внутри структуры, можно задать их размер. Такая структура будет занимать в памяти только 4 байта, но только 2 бита будет фактически использовано. Можно разместить в ней до 32-ух однобитовых полей, и это никак не скажется на размере выделенной для неё памяти.

```
/* Определение структуры с побитовыми полями */  
struct {  
    uint32_t width : 1;  
    uint32_t height: 1;  
} st2;
```

Пример оптимизации структуры для хранения даты

```
struct date {  
    uint32_t day : 5; // значение от 0 до 31  
    uint32_t month : 4; // значение 0 до 15  
    uint32_t year;  
};  
  
int main()  
{  
    struct date dt = { 31, 12, 2021 };  
    printf("Size is %lu\n", sizeof(dt));  
    printf("Date is %u/%u/%u\n", dt.day, dt.month, dt.year);  
    return 0;  
}
```

Size is 8 bytes
Date is
31/12/2021

Пример с ошибкой

Внимание! Если вы зададите значение, которое не помещается в поле с данным размером, то оно будет сохранено с ошибкой.

```
struct date dt = { 31, 12, 2021 };  
dt.month = 16;  
printf("Date is %u/%u/%u", dt.day, dt.month,  
dt.year);
```

Date is
31/0/2021

Внимание! В языке Си нет возможности создать структуру вида побитовый массив.

Как в памяти хранится вещественное число?

```
#include <stdio.h>
#include <inttypes.h>

union floatbit {
    float value;
    struct {
        uint32_t mant : 23;
        uint32_t exp : 8;
        uint32_t sign : 1;
    } bit;
} f;

int main()
{
    f.value = 4.0;
    printf("Memory size is %lu\n", sizeof(f));
    printf("f.value = %f\n", f.value );
    printf("sign = %x\n", f.bit.sign);
    printf("exp = %x\n", f.bit.exp);
    printf("mantissa = %x\n", f.bit.mant);
    return 0;
}
```

Memory size is
4
f.value =
4.000000
sign = 0
exp = 81
mantissa = 0

Задачи

По пройденному материалу

- ① Поменяйте знак с отрицательного переменной `x` типа `int` на положительный через побитовые операции

```
int32_t a, sign;  
scanf("%d", &a);  
sign = a >> 31; // записываем маску  
a = a ^ sign; // если число отрицательное то инверсия  
a = a + (sign & 0x1); // если число было отрицательное то +1  
printf("%d\n", a);
```

Задачи

По пройденному материалу

- ② Проверьте без использования арифметических операторов и операторов сравнения, равны ли два числа

```
_Bool isEqual(int a, int b) {  
    return !(a^b);  
}
```

Задачи

По пройденному материалу

③ Поменяйте знак переменной x типа float

```
void changeSign(float *f) {  
    union {  
        float f;  
        uint32_t u;  
    } tmp;  
    tmp.f = *f;  
    tmp.u = tmp.u ^ 0x80000000;  
    *f = tmp.f;  
}
```

Задачи

По пройденному материалу

- ④ Инвертируйте 5 младших битов переменной `x` типа `uint32_t`, остальные биты оставьте без изменений

Задачи

По пройденному материалу

- ⑤ Напишите функцию, которая возвращает истину, если знаки двух заданных 32 битных чисел разные. Использовать операции сравнения запрещено

```
_Bool difSign(int32_t a, int32_t b) {  
    return (a>>31) ^ (b>>31);  
}
```

Задачи

По пройденному материалу

- ⑥ Напишите функцию, которая вычитает единицу в случае, если число чётное, или не меняет его.
Использовать арифметические операции запрещено

```
int div1IfEven(int a) {  
    return a - !(a&0x1);  
}
```

Задачи

По пройденному материалу

- ⑦ Напишите функцию, которая сбрасывает установленный крайний правый бит целого числа

```
int32_t unset_rightmost (uint32_t n)
{
    return n & (n - 1);
}
```


Задачи

По пройденному материалу

- ⑧ Напишите логическую функцию. Определите, является ли введённое целое число степенью 4

```
_Bool isPowerOfFour(uint32_t n)
{
    return n != 0 && ((n & (n - 1)) == 0) && !(n & 0xAAAAAAAA);
}
```

Задачи

По пройденному материалу

- ⑨ Напишите функцию, которая осуществляет побитовый циклический сдвиг влево целого беззнакового 32 разрядного числа

```
int leftRotate(uint32_t n, uint32_t rotate)
{
    return (n << rotate) | (n >> (32 - rotate));
}
```

Задачи

По пройденному материалу

- ⑩ Напишите функцию, которая осуществляет побитовый циклический сдвиг вправо целого беззнакового 32 разрядного числа

```
int rightRotate(uint32_t n, uint32_t rotate)
{
    return (n >> rotate) | (n << (32 - rotate));
}
```

Задачи

По пройденному материалу

- ⑪ Опишите функцию, которая умножает вещественное число `float` на 4. Операцию умножения использовать запрещено.

```
union floatbit {  
    float value;  
    struct {  
        uint32_t mant : 23;  
        uint32_t exp : 8;  
        uint32_t sign : 1;  
    } bit;  
} f;
```

```
float mult4(float f) {  
    union floatbit tmp;  
    tmp.value = f;  
    tmp.bit.exp += 2;  
    return tmp.value;  
}
```

Задачи

По пройденному материалу

- ⑫ Дано положительное целое число n , подсчитайте общее количество установленных битов в двоичном представлении всех чисел от 1 до n

Задачи

По пройденному материалу

13

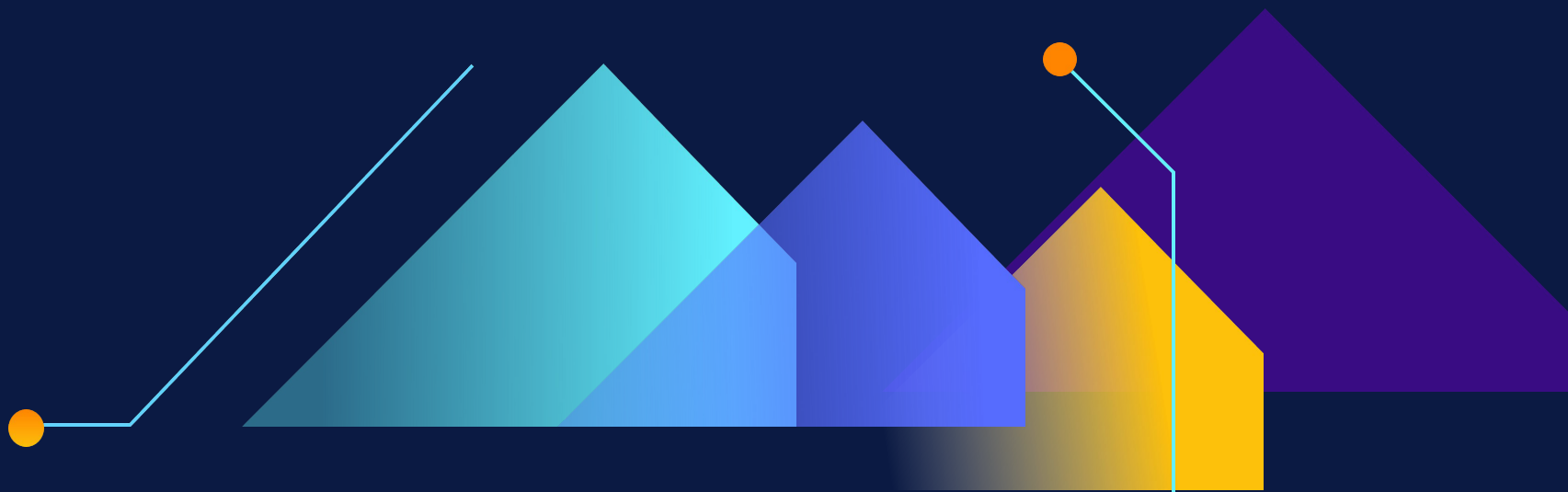
Дано целое число без знака, переверните все его биты и верните число с перевернутыми битами.

Например: 11 (двоичный вид 1011) преобразуется в 13 (двоичный вид 1101)

```
uint32_t reverseBits(uint32_t num)
{
    uint32_t NO_OF_BITS = sizeof(num) * 8;
    uint32_t reverse_num = 0;
    int32_t i;
    for (i = 0; i < NO_OF_BITS; i++)
    {
        if((num & (1 << i)))
            reverse_num |= 1 << ((NO_OF_BITS - 1) - i);
    }
    return reverse_num;
}
```

```
uint32_t bitRevers(uint32_t n) {
    uint32_t r=0;
    while(n) {
        r <<= 1;
        r |= n&0x1;
        n >>= 1;
    }
    return r;
}
```

Массивы, структуры и функции

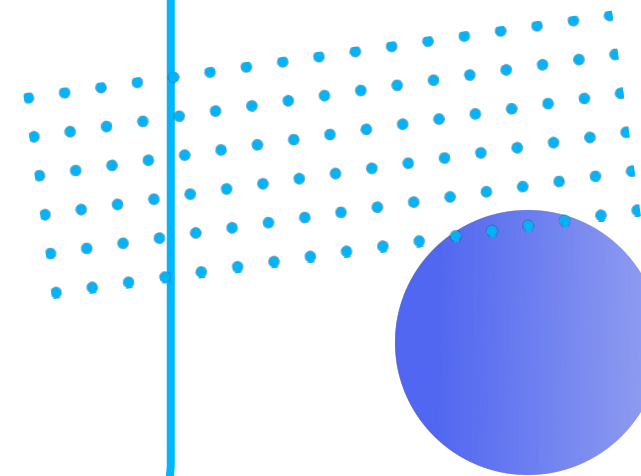




Массивы

Указателями в Си являются переменные, которые хранят адрес. Указатель всегда имеет одинаковый размер. Указатель может хранить адрес:

- Переменной, в том числе указателя
- Массива или строки
- Структуры
- Функции



Примеры

```
int i=123;  
int *pi; //указатель на переменную  
pi = &i;  
int **ppi; // указатель на указатель  
ppi = &pi;  
printf("**ppi = %d\n", **ppi);
```

```
int ar[5];  
    int *pa; //указатель на массив  
    pa = &ar[0]; // pa = ar;
```

Примеры

```
struct s {  
    int i;  
    float f;  
} st;  
struct s *ps;  
ps = &st; //указатель на структуру  
printf("ps -> i = %d\n", ps->i);
```

```
int ar[3][5];  
int (*pa)[5]; //указатель массив из 5-и элементов  
pa = ar+1; //адрес 1-ой строки
```

Структуры

При обращении к полю структуры по указателю на структуру можно использовать

оператор ' \rightarrow '. $ps \rightarrow i$ эквивалент $(*ps).i$;

Структуры можно передавать в функции и возвращать из функции как по значению, так и по ссылке. При передаче по значению происходит копирование всей структуры на стек. В случае, если структура занимает много места в памяти, то оптимально передавать в функцию её адрес, чтобы избежать дополнительного расхода памяти.

Пример

```
struct student{
    int id;
    char name[20];
    int group;
};

void func(struct student
record){
    printf(" Id is: %d \n",
record.id);
    printf(" Name is: %s \n",
record.name);
    printf(" Group is: %d \n",
record.group);
}
```

```
void pfunc(struct student
*record)
{
    printf(" Id is: %d \n",
record->id);
    printf(" Name is: %s \n",
record->name);
    printf(" Group is: %d \n",
record->group);
}
```

Пример

```
{  
    struct student record = {1,  
"Vasiliy", 102};  
  
    func(record) ;  
    pfunc(&record) ;  
    return 0 ;  
}
```

Id is: 1

Name is: Vasiliy

Group is: 102

Id is: 1

Name is: Vasiliy

Group is: 102

Пример

При описании структурного типа память не выделяется. Выделение памяти происходит только после объявления переменных.

```
struct student // Описание нового типа. Память не выделяется.  
{  
    int id;  
    char name[20];  
    int group;  
};  
struct student st; // Описание переменной. Выделяется память  
под нее.
```

Пример описания структурного типа

Внимание! При описании структурного типа делать инициализацию нельзя.

```
// ТАК НЕЛЬЗЯ
struct student
{
    char name[20] = "Ivan";
    int group;
};
```

```
// Так можно
struct student
{
    char name[20];
    int group;
};
struct student st = {"Ivan",
104};
```

Передача функции в функцию

Указатель на функцию задаётся следующим образом:

```
int func(int n) {  
    printf("Hello func %d\n",n);  
    return n+1;  
}  
  
int main()  
{  
    int (*fp)(int);  
    fp = func;  
    fp(5);  
    return 0;  
}
```

Hello func 5

Создание синонима типа typedef и #define

При описании структурных типов или массивов, удобно использовать **typedef**. Это не директива препроцессора, — она обрабатывается компилятором, а не препроцессором.

```
typedef struct {  
    int id;  
    char name[20];  
    int group;  
} student;  
student s;  
s.id = 1;
```

```
typedef int iarr[10];  
iarr a, b, c[5];  
// тоже самое int a[10],  
b[10], c[10][5];
```

Создание синонима типа typedef и #define

Необходимо отличать **typedef** от **#define**. Директива препроцессора, в отличие от typedef, обрабатывается перед трансляцией в машинный код. Поэтому такое возможно, но так лучше не делать.

```
/*Так можно*/  
#define float double;  
float f = 1.23;
```

```
/*Так НЕЛЬЗЯ*/  
typedef double float;  
float f = 1.23;
```

Задачи

По пройденному материалу

- ① Опишите функцию `void array2struct(int ar[], struct pack_array *ps)`, которая упаковывает массив из 32-ух элементов, содержащий только 0 и 1, в структуру вида

```
struct pack_array {  
    uint32_t array; // массив из 0 и 1  
    uint32_t count0 : 8; // количество 0 в массиве  
    uint32_t count1 : 8; // количество 1 в массиве  
}  
  
void array2struct(int ar[], struct pack_array *ps);
```

Задачи

По пройденному материалу

- ① Опишите функцию `void array2struct(int ar[], struct pack_array *ps)`, которая упаковывает массив из 32-ух элементов, содержащий только 0 и 1, в структуру вида

```
struct pack_array {  
    uint32_t array; // массив из 0 и 1  
    uint32_t count0 : 8; // количество 0 в массиве  
    uint32_t count1 : 8; // количество 1 в массиве  
}  
  
void array2struct(int ar[], struct pack_array *ps);
```

Задачи

По пройденному материалу

- ② Опишите функцию `void struct2array(int ar[], struct pack_array *ps)`, которая распаковывает структуру в массив из 32-ух элементов, в структуру вида

Задачи

По пройденному материалу

- ③ Опишите функцию, которой на вход передаётся вещественное число в типе float, она возвращает порядок в виде десятичного целого числа.
- ```
int
extractExp(float
f) .
```

```
int extractExp(float f) {
 union {
 float f;
 struct {
 uint32_t mantissa :
23;
 uint32_t exp : 8;
 uint32_t s : 1;
 } field;
 } fl;
 fl.f = f;
 return fl.field.exp;
}
```

# Текстовый материал



Текст...

# Таблица

---

|  |  |  |  |  |  |
|--|--|--|--|--|--|
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |



# Домашнее задание



Что установить/ скачать?

....



Обратная связь

....

# Задание

---

1. 1.

2. 2.

Комментарий:

...



**Дедлайн:** конец курса

Советуем регулярно выполнять ДЗ  
(наверстать пСропуски тяжело)

# Вопросы для самопроверки

---

По пройденному материалу

① ..... ?

...

② ..... ?

...

# Правила

|        |                  |                      |
|--------|------------------|----------------------|
| Шрифты | Заголовок слайда | Source Sans Pro 35px |
|        | Подтема          | Inter Bold 20px      |
|        | Основной текст   | Inter 20px           |
|        | Комментарии      | Inter 16px           |

|       |                                                                                     |                         |                                                                                     |                        |                                                                                       |                         |                                                                                       |                          |
|-------|-------------------------------------------------------------------------------------|-------------------------|-------------------------------------------------------------------------------------|------------------------|---------------------------------------------------------------------------------------|-------------------------|---------------------------------------------------------------------------------------|--------------------------|
| Цвета |  | #00B7FF<br>R0 G183 B255 |  | #0B1A43<br>R11 G26 B67 |  | #FFB600<br>R255 G182 B0 |  | #566CFE<br>R86 G108 B254 |
|       |                                                                                     |                         |                                                                                     |                        |                                                                                       |                         |                                                                                       |                          |

# Иконки

