

# Программирование на языке С++

## Лекция 9

Структуры

## Постановка задачи

- Хранить в программе описание характеристик некоторого объекта

## Решение I

```
int aliceBirthYear;  
int aliceBirthMonth;  
int aliceBirthDay;  
double aliceHeight;  
double aliceWeight;
```

```
int bobBirthYear;  
int bobBirthMonth;  
int bobBirthDay;  
double bobHeight;  
double bobWeight;
```

# Решение I - Проблемы

- Для каждого человека нужно создавать по пять отдельных переменных – **долго, могут быть опечатки**
- Чтобы передать в функцию, нужно перечислить все аргументы – **можно перепутать порядок**

```
print(aliceBirthYear, aliceBirthMonth,  
      aliceBirthDay, aliceHeight, aliceWeight);
```

- Как вернуть из функции?

## Решение II - Структуры

```
struct human {      // Свой тип данных
    int BirthYear;
    int BirthMonth;
    int BirthDay;
    double Height;
    double Weight;
};    // Точка с запятой обязательно
```

```
human alice, bob;    // Создаём переменные
```

## Решение II - Структуры

```
struct human {  
    int BirthYear;  
    int BirthMonth;  
    int BirthDay;  
    double Height;  
    double Weight;  
} alice, bob;
```

## Решение II - Структуры

```
struct {  
    int BirthYear;  
    int BirthMonth;  
    int BirthDay;  
    double Height;  
    double Weight;  
} alice, bob;
```

# Где можно объявлять структуры?

- Внутри функций

```
void func(){
    struct num{int i;} var;
};
```
- Вне функций

```
struct num{int i;} var;
void func(){
};
```
- Внутри других структур

```
struct num{
    int i;
    struct {int k;} j;
} var;
```



# Что может быть членом структуры?

Если можно создать переменную этого типа, то это может быть членом структуры

Например:

- Примитивные типы: `int`, `double`, `char` ...
- Другие структуры;
- Массивы;
- Строки;
- ...

# Как работать со структурой

```
struct Data{  
    int Year;  
    int Month;  
    int Day;  
};
```

```
Data now;  
now.Year = 2018;  
now.Day = 9;  
now.Month = 11;
```

## Как работать со структурой

```
now.Year = now.Year + 1; // 2019
```

```
cout << now.Day; // 9
```

```
now.Month = now.Day + now.Year; // 2028
```

```
int *p = &now.Month;
```

# Инициализация структуры I

```
struct Employee {  
    short id;  
    int age;  
    double wage;  
};
```

```
// joe.id = 1, joe.age = 32, joe.wage = 60000.0  
Employee joe = { 1, 32, 60000.0 };  
// frank.id = 2, frank.age = 28, frank.wage = 0.0  
Employee frank = { 2, 28 };  
Employee frank { 2, 28 };    // C++11
```

## Инициализация структуры II C++11/C++14

```
struct Rectangle {  
    double length = 1.0;  
    double width = 1.0;  
};
```

```
int main() {  
    Rectangle x; // length = 1.0, width = 1.0  
    x.length = 2.0; // Меняем значение  
    return 0;  
}
```

## Инициализация структуры III C++11/C++14

```
struct Rectangle {  
    double length = 1.0;  
    double width = 1.0;  
};
```

```
int main() {  
    // C++11 – Ошибка; C++14 – Разрешено  
    Rectangle x = {1.0, 1.0};  
  
    return 0;  
}
```

# Присваивание значений структурам I

```
struct Employee {  
    short id;  
    int age;  
    double wage;  
};
```

```
Employee joe;  
joe.id = 1;  
joe.age = 32;  
joe.wage = 60000.0;
```

## Присваивание значений структурам II

```
struct Employee {  
    short id;  
    int age;  
    double wage;  
};
```

```
Employee joe = {1, 20, 3.0}, mike;  
mike = joe; // Копирование значений joe в mike  
  
// Присваивание полям joe новых значений  
joe = {2, 22, 6.3};
```



## Передача структуры как параметр в функцию

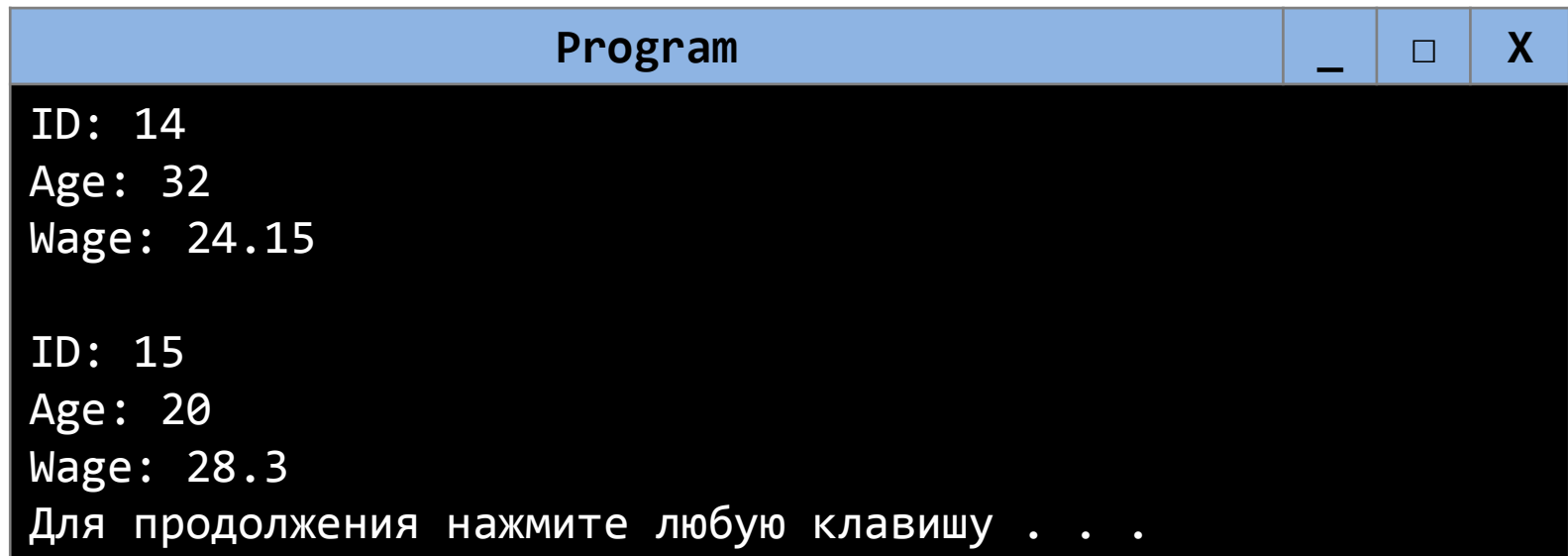
```
struct Employee {  
    short id;  
    int age;  
    double wage;  
};
```

```
void printInformation(Employee employee) {  
    std::cout << "ID: " << employee.id << "\n";  
    std::cout << "Age: " << employee.age << "\n";  
    std::cout << "Wage: " << employee.wage << "\n";  
}
```

## Передача структуры как параметр в функцию

```
int main() {  
    Employee joe = { 14, 32, 24.15 };  
  
    printInformation(joe);  
    std::cout << "\n";  
  
    printInformation({ 15, 20, 28.3 });  
    return 0;  
}
```

# Передача структуры как параметр в функцию



The screenshot shows a standard Windows console window with a blue title bar labeled 'Program'. The window contains white text on a black background. The text displays two sets of data: the first set shows 'ID: 14', 'Age: 32', and 'Wage: 24.15'; the second set shows 'ID: 15', 'Age: 20', and 'Wage: 28.3'. At the bottom, there is a prompt 'Для продолжения нажмите любую клавишу . . .' which translates to 'Press any key to continue . . .'. The window has standard minimize, maximize, and close buttons on the right side of the title bar.

```
Program
ID: 14
Age: 32
Wage: 24.15

ID: 15
Age: 20
Wage: 28.3
Для продолжения нажмите любую клавишу . . .
```

# Передача структуры в функцию через указатель

```
void printInformation(Employee *employee) {  
    std::cout << "ID: " << (*employee).id << "\n";  
    std::cout << "Age: " << (*employee).age << "\n";  
    std::cout << "Wage: " << (*employee).wage << "\n";  
}
```

```
void printInformation(Employee *employee) {  
    std::cout << "ID: " << employee->id << "\n";  
    std::cout << "Age: " << employee->age << "\n";  
    std::cout << "Wage: " << employee->wage << "\n";  
}
```

# Возврат структур из функций

```
struct Point3d {  
    double x, y, z;  
};
```

```
Point3d getZeroPoint() {  
    Point3d temp = { 0.0, 0.0, 0.0 };  
    return temp;  
}
```

```
int main() {  
    Point3d zero = getZeroPoint();  
    return 0;  
}
```

Дополнительные сведения

# Разные типы

```
struct Point3d {  
    double x, y, z;  
};
```

```
struct Vector3d {  
    double x, y, z;  
};
```

```
Point3d p = { 0.0, 0.0, 0.0 };
```

```
Vector3d v;
```

```
v = p; // Ошибка. У v и p разные типы
```

# Массив структур

```
struct Point3d {  
    double x, y, z;  
};
```

```
Point3d p[2] = {{}, {1.0, 2.0, 3.0}};
```

```
p[0].x = 1.0;
```

```
std::cout << p[0].x << ' ' << p[0].y << ' ' << p[0].z;
```



# Вложенные структуры

```
struct Employee {  
    short id;  
    int age;  
    float wage;  
};
```

```
struct Company {  
    Employee CEO;    // CEO – это структура  
    int numberOfEmployees;  
};
```

```
Company myCompany = {{ 1, 42, 60000.0f }, 5 };  
std::cout << myCompany.CEO.id;
```

# Размер структуры и выравнивание I

```
struct Employee {  
    short id;      // sizeof(short) == 2  
    int age;       // sizeof(int) == 4  
    double wage;   // sizeof(double) == 8  
};
```

```
sizeof(Employee); // 16 != ( 2 + 4 + 8 )
```

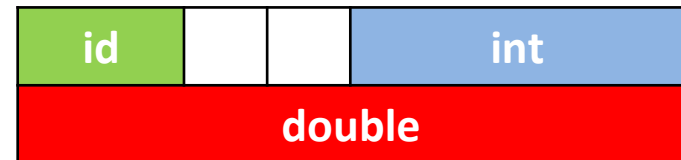
## Размер структуры и выравнивание II

```
struct Employee {  
    short id;      // sizeof(short) == 2  
    double wage;   // sizeof(double) == 8  
    int age;       // sizeof(int) == 4  
};
```

```
sizeof(Employee); // 24 != ( 2 + 4 + 8 )
```

## Размер структуры и выравнивание II

```
struct Employee {  
    short id;  
    int age;  
    double wage;  
};
```



```
struct Employee {  
    short id;  
    double wage;  
    int age;  
};
```

