




Язык программирования C++

Объединение переменных



С помощью массивов можно объединить переменные одного типа. В реальном мире, однако, требуется объединять между собой данные различных типов.

Например, к характеристикам автомобиля относятся:


- марка и тип - это последовательности символов;
- пробег и производительность — целочисленные величины;
- цена — величина вещественная (возможно тип double);

Для объединения разнотипных переменных в языке C ++ используются **структуры**.

Под структурами подразумевают группу переменных, объединенных общим именем.

Объявление структуры:

```
struct имя {  
    тип1 поле1;  
    тип2 поле2;  
    ...  
    типN полеN;  
} список переменных ;
```



Описание структуры – это всего лишь шаблон, по которому впоследствии создаются переменные.

Переменная структуры в программе создаются точно также, как и переменная любого базового типа:

- можно сразу, указав список с названиями после описания структуры:
- можно в любом месте кода.

Пример: для моделирования автомобиля создать новый тип из нескольких элементов.

struct TAutoTyp // объявляем новый тип

```
{  
    char brand[MaxMarka];  
    char model[MaxModel];  
    long km;  
    int kW;  
    float price;  
} ;
```

Новый тип **TAutoTyp** создан.

Можно, например, создать новую переменную такого типа, или новый массив, или объявить указатель с таким типом.

TAutoTyp collection; // Объявлена новая переменная

TAutoTyp auto_parking[100]; // Массив автомобилей

TAutoTyp *parking_map; // Указатель на автомобиль

Все данные, которые будут использоваться при выполнении программы, записываются в поля переменной.

Пример:

collection.km = 128000;

collection.kW = 25;

collection.price = 25000.00;

Структуры можно инициализировать.
Для этого используются фигурные скобки.

```
TAutoTyp JB =  
{"Aston Martin", "DB5", 12000, 90, 12.95};
```

```
TAutoTyp GWB = {0};
```


Доступ к элементам структуры через указатель:

```
TAutoTyp *parking_map;
```

```
parking_map = &collection;
```

```
(*parking_map).price = 12500;
```

ИЛИ

```
parking_map->price = 12500;
```

Важный аспект- расположение объявления структуры.

Существует два варианта:

1. Объявление внутри функции main()
2. Объявление за пределами ф-ии main()

Для программ, состоящих из нескольких функций, способ объявления имеет значение.

Внешнее объявление может быть использовано всеми функциями, которые следуют за структурой.

Внутренне объявление - только той функцией, в которой объявлена структура.

ПРИМЕР

struct marks {

char name[10]; int phys;

int chem; int maths; };

struct exams{

double phys;

double chem;

double maths; };



```
int main()
```

```
{ marks ivanov={"Victor",4,4,3};
```

```
    marks petrov={"Andre",3,4,3} ;
```

```
exams LastYear,ThisY;
```

```
    LastYear.chem=4.33; LastYear.maths=3.82;
```

```
    LastYear.phys=3.5;
```



```
ThisY.chem=(ivanov.chem+petrov.chem)/2;
```


```
ThisY.maths=(ivanov.maths + petrov.maths) /2;
```

```
ThisY.phys =(ivanov.phys+petrov.phys)/2;
```

```
cout<<"mathematics "<<ThisY.chem<<endl;
```

```
return 0;
```

```
}
```



По отношению к структурам можно применять операцию присваивания. Для этого две переменные должны относиться к одной структуре. В результате такого присваивания из одной переменной в другую копируются значения всех полей структуры.

```
marks sidorov={"Vlad",33,3};
```

```
marks novikov;
```

```
novikov=sidorov;
```