Программирование и основы алгоритмизации

Типы данных, определяемые разработчиком

03.03.2022 г., 10.03.2022 г.

Гришмановский Павел Валерьевич,
кафедра автоматики и компьютерных систем, Политехнический институт, СурГУ

Описание новых типов данных

Новые типы данных могут быть описаны на основе уже существующих типов данных – встроенных или описанных ранее

Способы описания новых типов данных:

- **`typedef`** – переименование типов
- **`enum`** – перечисления
- **`struct`** – структуры
- **`union`** – объединения (смеси)

Переименование типов

Вводит имя, использование которого эквивалентно описанию некоторого типа данных

Вариант 1 (простой):

```
typedef <описание_типа> <имя_типа>;
```

Вариант 2 (как это работает на самом деле):

если перед любым описанием дописать **typedef**, то целевой идентификатор станет не именем программного объекта, а именем для соответствующего типа

Перечисления

Перечисление задает множество именованных целочисленных значений

Перечисление эквивалентно типу **`int`** – имеет такой же размер, формат внутреннего представления, те же операции над значениями и т.п.

```
enum <тэг>
  { <имя1> [= <к.вып.1>], <имя2> [= <к.вып.2>]... };
```

Именем типа будет являться сочетание **`enum`** *`<тэг>`*, но для удобства использования перечисление можно переименовать непосредственно при его описании (*`<тэг>`* и *`<имя_типа>`* могут совпадать):

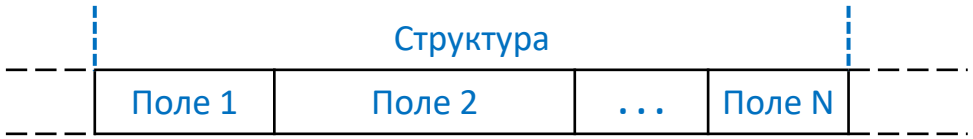
```
typedef
  enum <тэг>
    { <имя1> [= <к.вып.1>], <имя2> [= <к.вып.2>]... }
  <имя_типа>;
```

Структуры

Структура – это совокупность элементов данных, называемых **полями**, которые могут иметь разные типы и располагаются в памяти непосредственно друг за другом.

Каждое поле подобно переменной, но имеет имя, уникальное в пределах структуры, и существует только как часть экземпляра структуры

```
struct <тэг>
{
   <тип1> <имя1>[, ...];
   <тип2> <имя2>[, ...];
   ...
};
```



Именем типа будет являться сочетание **struct** *<тэг>*, но для удобства использования структуру можно переименовать непосредственно при описании (*<тэг>* и *<имя_типа>* могут совпадать):

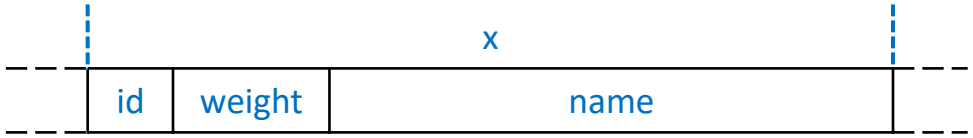
```
typedef struct <тэг> { <описание_полей> } <имя_типа>;
```

Структуры

Тип структуры (**struct** *<тэг>* или *<имя_типа>* в случае переименования) используется так же, как любой другой тип данных:

```
struct Good
{
  int    id;
  double weight;
  char   name[30];
};

struct Good x;
```



Переменная `x` содержит в себе три поля (состоит из): `id`, `weight`, `name`

Обращение к полям выполняется при помощи специальных операций – селекторов:

- прямой селектор – при обращении по имени экземпляра
- косвенный селектор – при обращении по указателю на экземпляр

```
x.id = 123;
x.weight = 2.45;
```

```
struct Good * p;
p = &x;
p->id = 123;
p->weight = 2.45;
```

Структуры

Инициализация структуры выполняется посредством агрегата:

- поля заполняются в порядке их описания
- типы элементов агрегата должны соответствовать типам полей
- последние поля неявно заполняются нулями

```
struct Good
{
  int    id;
  double weight;
  char   name[30];
};

struct Good x = { 1, 2.345, "Unknown" };

struct Good a[5] = {  { 22, 18.05, "Owen"  },
                      { 23, 0.132, "Phone" },
                      { 78, 87             }  };
```

22	18.05	"Owen"
23	0.132	"Phone"
78	87.0	""
0	0.0	""
0	0.0	""

Структуры

Операции над структурами:

- адрес
- размер (sizeof)
- селекторы
- присваивание структур одного типа