# Комбинации типов

Представлены массивы, структуры и указатели. Они могут комбинироваться разнообразными способами.

*struct antarctica\_years\_end*

*{*

*int year;*

*/\* определение других нужных данных \*/*

*};*

**Можно создавать переменные этого типа:**

antarctica\_years\_end sOl, s02, s03; // sOl, s02, s03 - структуры

После этого можно обращаться к членам с использованием операции принадлежности:

sOl.year = 1998;

**Можно создать указатель на такую структуру:**

antarctica\_years\_end \* pa = &s02;

Имея указатель, установленный в допустимый адрес, можно использовать операцию членства через указатель для доступа к членам:

ра->уеаг = 1999;

**Можно создавать массивы структур:**

antarctica\_years\_end trio[3]; // массив из трех структур

Затем с помощью операции принадлежности можно обращаться к членам какого-нибудь элемента:

trio[0].year = 2003; // trio[0] является структурой

Здесь trio — это массив, но trio [0] — структура, и trio [0] .year представляет собой член этой структуры. Поскольку имя массива является указателем, можно также применять операцию членства через указатель:

(trio+l) ->year = 2004; // то же, что и trio[l] .year = 2004;

**Можно создавать массивы указателей**:

const antarctica\_years\_end \* arp[3] = (&s01, &s02, &s03};

Это уже выглядит немного сложнее. Как получить доступ к данным в этом массиве? Если агр — массив указателей, то

агр [1] должен быть указателем, и для доступа к члену можно воспользоваться операцией членства через указатель:

std::cout « arp[l]->year « std::endl;

**Можно создавать указатель на такой массив:**

const antarctica\_years\_end \*\* рра = агр;

Здесь агр представляет собой имя массива; следовательно, он является адресом первого элемента в массиве. Но его первый элемент — указатель, поэтому рра должен быть указателем на указатель на const antarctica\_years\_end, отсюда и \*\*.

Существует немало путей внести путаницу в это объявление. Например, можно было бы забыть о const или о звездочке (а то и о двух), переставить буквы либо еще каким-то образом исказить этот тип структуры. Это может служить примером удобства применения ключевого слова auto из С++11. Компилятор хорошо осведомлен о типе агр, поэтому он может вывести правильный тип самостоятельно:

auto ppb = агр; // автоматическое выведение типа в С++11

В прошлом компилятор использовал свои знания о правильном типе для сообщения об ошибках, которые вы могли допустить в объявлении; теперь же эти знания работают на вас. Как использовать рра для доступа к данным? Поскольку рра — это указатель на указатель на структуру, \*рра представляет собой указатель на структуру, так что его

можно применять с операцией членства через указатель:

std::cout << (\*ppa)->year << std::endl;

std::cout << (\*(ppb+1))->year << std::endl;

Так как рра указывает на первый член агр, \*рра является первым членом, т.е. &s01. Таким образом, (\*рра)->уеаг — это член year в s01. Во втором операторе ppb+1 указывает на следующий элемент, агр[1], т.е. &s02. Круглые скобки нужны для обеспечения корректной ассоциации. Например, \*рра->уеаг будет пытаться применить операцию \* к рра->уеаг, что даст сбой, поскольку член year не является указателем.

**Листинг 4.23. mixtypes. срр**

*// mixtypes.срр -- некоторые комбинации типов*

*#include <iostream>*

*struct antarctica\_years\_end*

*{*

*int year;*

*/\* определение других нужных данных \*/*

*};*

*int main()*

*{*

*antarctica\_years\_end sOl, s02, s03; '*

*sOl.year = 1998;*

*antarctica\_years\_end \* pa = &s02;*

*pa->year = 1999;*

*antarctica\_years\_end trio[3]; // массив из трех структур*

*trio[0].year = 2003;*

*std::cout << trio->year << std::endl;*

*const antarctica\_years\_end \* arp[3] = {&s01, &s02, &s03};*

*std::cout << arp[l]->year << std::endl;*

*const antarctica\_years\_end \*\* ppa = arp;*

*auto ppb = arp; // автоматическое выведение типа в C++11*

*// или можно использовать const antarctica\_years\_end \*\* ppb = arp;*

*std::cout << (\*ppa)->year << std::endl;*

*std::cout « (\*(ppb+1))->year « std::endl;*

*return 0;*

*}*

**Ниже показан вывод:**

*2003*

*1999*

*1998*

*1999*