Перегрузка функций

Язык C++ поддерживает перегрузку функций (в языке C перегрузки нет). Перегрузка означает, что можно определять несколько разных функций с *одинаковыми именами*. В этом случае говорят, что имя функции “перегружено”.

Единственное ограничение состоит в том, что перегруженные функции должны *отличаться набором типов параметров*.

Пример 1. Перегруженная функция dist вычисляет расстояние между точками на прямой, на плоскости и в пространстве.

```
float dist (float x, float y)
{ return fabs(x - y); }

float dist (float x1, float y1, float x2, float y2)
{ return sqrt((x2 - x1)*(x2 - x1) + (y2 - y1)*(y2 - y1)); }

float dist (float x1, float y1, float z1, float x2, float y2, float z2)
{ return sqrt((x2 - x1)*(x2 - x1) + (y2 - y1)*(y2 - y1) + (z2 - z1)*(z2 - z1));}


int main()
{
cout << dist (1.1, 3.3);      // 2.2
cout << dist (0, 0, 3, 4);    // 5
cout << dist (0, 0, 0, 3, 4, 12);  // 13
return EXIT_SUCCESS;
}
```

В данном примере различные версии перегруженной функции имеют *разное количество параметров*, и компилятор различает вызовы по этому признаку.

Например, в первом вызове функции dist (в теле main) указано 2 параметра. Это значит, что используется первая версия перегруженной функции. Во втором вызове указано 4 параметра, поэтому вызывается вторая версия функции dist. И, наконец, в последнем вызове – 6 параметров. Следовательно, используется третья версия рассматриваемой функции.

Пример 2. Следующая перегрузка допустима, поскольку *порядок параметров* в указанных ниже функциях различен.

```
int f(int x, double y)
{
    // тело функции
}

int f(double a, int b)
{
    // тело функции
}

int main()
{
    cout << f(5, 7.2);    // вызывается первая функция
    cout << f(11.4, 8);   // вызывается вторая функция
    return EXIT_SUCCESS;
}
```

Компилятор учитывает эти различия в параметрах и вызывает нужную функцию.

Перегрузка функций может конфликтовать с заданием значений по умолчанию для некоторых параметров.

Пример 3.

```
int f(int, int = 0)
{
    // тело функции
}

int f(int)
{
    // тело функции
}
int main()
{
    cout << f(7);
    // компилятору неясно, какую из двух функций нужно вызывать
    return EXIT_SUCCESS;
}
```

В подобной ситуации компилятору не ясно, какую из двух функций (первую или вторую) вызывать в выражении f(7).

Замечание.

Отличия в типах возвращаемого значения или в именах параметров недостаточно.

Многие библиотечные функции (например, sqrt, pow) являются перегруженными. Например, функция sqrt имеет следующие перегруженные версии:

```
double sqrt (double)  // функция корня для чисел типа double

float sqrt (float)   // функция корня для чисел типа float

int sqrt (int)       // функция корня для чисел типа int
```

Компилятор сам определяет какую функцию выбрать, в зависимости от типа передаваемого параметра.

```
cout << sqrt(4.1);  // вызовется функция для чисел типа double
            //(по умолчанию вещественное число имеет тип double)
cout << sqrt(4.1f);  // вызовется функция для чисел типа float

cout << sqrt(4);   // вызовется функция для чисел типа int
```

Упражнение.

Перегрузить функцию area, вычисляющую полную площадь поверхности куба по ребру, правильной четырехугольной призмы по стороне основания и высоте, прямоугольного параллелепипеда по трем ребрам.