Объекты и классы

Простой класс

```
// smallobj.cpp
// демонстрирует простой небольшой объект
#include <iostream>
using namespace std;
////////////////////////////////////////////////
class smallobj     // определение класса
{
private:
      int somedata;    // поле класса
public:
      void setdata(int d)  // метод класса, изменяющий значение поля
      {
            somedata = d;
      }
      void showdata()   // метод класса, отображающий значение поля
      {
            cout << "Значение поля равно " << somedata << endl;
      }
};
////////////////////////////////////////////////////////
int main()
{
      smallobj s1, s2;  // определение двух объектов класса smallobj
      s1.setdata(1066); // вызовы метода setdata()
      s2.setdata(1776);
      s1.showdata();    // вызовы метода showdata()
      s2.showdata();
      return 0;
}
```

Детали изделия в качестве объектов

```
// objpart.cpp
// детали изделия в качестве объектов
#include <iostream>
using namespace std;
////////////////////////////////////////////////////////
class part  // определение класса
{
private:
      int modelnumber; // номер изделия
      int partnumber;  // номер детали
      float cost;      // стоимость детали
public:
      void setpart(int mn, int pn, float c) // установка данных
      {
            modelnumber = mn;
            partnumber = pn;
            cost = c;
      }
      void showpart()  // вывод данных
      {
            cout << "Модель " << modelnumber;
            cout << ", деталь " << partnumber;
            cout << ", стоимость $" << cost << endl;
      }
};
////////////////////////////////////////////////////////
      int main()
      {
            part part1;  // определение объекта класса part
            part1.setpart(6244, 373, 217.55F);  // вызов метода
            part1.showpart();  // вызов метода
            return 0;
      }
```

Класс как тип данных

```
// englobj.cpp
// длины в английской системе в качестве объектов
#include <iostream>
using namespace std;
////////////////////////////////////////////////////////
class Distance  // длина в английской системе
{
private:
	int feet;
	float inches;
public:
	void setdist(int ft, float in)// установка значений полей
	{
		feet = ft; inches = in;
	}
	void getdist()   // ввод полей с клавиатуры
	{
		cout << "\nВведите число футов: "; cin >> feet;
		cout << "Введите число дюймов: "; cin >> inches;
	}
	void showdist()  // вывод полей на экран
	{
		cout << feet << "\'-" << inches << '\"';
	}
};
	////////////////////////////////////////////////////////
	int main()
	{
		Distance dist1, dist2;    // две длины
		dist1.setdist(11, 6.25); // установка значений для d1
		dist2.getdist();        // ввод значений для dist2
		// вывод длин на экран
		cout << "\ndist1 - "; dist1.showdist();
		cout << "\ndist2 - "; dist2.showdist();
		cout << endl;
		return 0;
	}
```

Пример со счетчиком

```
// counter.cpp
// счетчик в качестве объекта
#include <iostream>
using namespace std;
////////////////////////////////////////////////////////
class Counter
{
private:
	unsigned int count;  // значение счетчика
public:
	Counter() : count(0)  // конструктор
	{ /* пустое тело */
	}
	void inc_count()  // инкрементирование счетчика
	{
		count++;
	}
	int get_count()  // получение значения счетчика
	{
		return count;
	}
};
////////////////////////////////////////////////////////
int main()
{
	Counter c1, c2;  // определение с инициализацией
	cout << "\nc1 =" << c1.get_count(); // вывод
```

```
        cout << "\nc2 =" << c2.get_count();
        c1.inc_count();  // инкрементирование c1
        c2.inc_count();  // инкрементирование c2
        c2.inc_count();  // инкрементирование c2
        cout << "\nc1 =" << c1.get_count(); // вывод
        cout << "\nc2 =" << c2.get_count();
        cout << endl;
        return 0;
}
```

Упражнения

1. Создайте класс Int, имитирующий стандартный тип int. Единственное поле этого класса должно иметь тип int. Создайте методы, которые будут устанавливать значение поля, равным нулю, инициализировать его целым значением, выводить значение поля на экран и складывать два значения типа Int. Напишите программу, в которой будут созданы три объекта класса Int, два из которых будут инициализированы. Сложите два инициализированных объекта, присвойте результат третьему, а затем отобразите результат на экране.

2. Представьте пункт для взимания платежей за проезд по автостраде. Каждая проезжающая машина должна заплатить за проезд 50 центов, однако часть машин платит за проезд, а часть проезжает бесплатно. В кассе ведется учет числа проехавших машин и суммарная выручка от платы за проезд. Создайте модель такой кассы с помощью класса tollBooth. Класс должен содержать два поля. Одно из них, типа unsigned int, предназначено для учета количества проехавших автомобилей, а второе, имеющее тип double, будет содержать суммарную выручку от оплаты проезда. Конструктор должен инициализировать оба поля нулевыми значениями. Метод payingCar() инкрементирует число машин и увеличивает на 0.50 суммарную выручку. Другой метод, nopayCar(), увеличивает на единицу число автомобилей, но оставляет без изменения выручку. Метод display() выводит оба значения на экран. Там, где это возможно, сделайте методы константными. Создайте программу, которая продемонстрирует работу класса. Программа должна предложить пользователю нажать одну клавишу для того, чтобы сымитировать заплатившего автолюбителя, и другую клавишу, чтобы сымитировать недобросовестного водителя. Нажатие клавиши Esc должно привести к выдаче текущих значений количества машин и выручки и завершению программы.

3. Создайте класс с именем time, содержащий три поля типа int, предназначенные для хранения часов, минут и секунд. Один из конструкторов класса должен инициализировать поля нулевыми значениями, а другой конструктор — заданным набором значений. Создайте метод класса, который будет выводить значения полей на экран в формате 11:59:59, и метод, складывающий значения двух объектов типа time, передаваемых в качестве аргументов. В функции main() следует создать два инициализированных объекта (подумайте, должны ли они быть константными) и один неинициализированный объект. Затем сложите два инициализированных значения, а результат присвойте третьему объекту и выведите его значение на экран. Где возможно, сделайте методы константными.

4. Создайте класс employee. Класс должен включать поле типа int для хранения номера сотрудника и поле типа float для хранения величины его оклада. Методы класса должны позволять пользователю вводить и отображать данные класса. Напишите функцию main(), которая запросит пользователя

ввести данные для трех сотрудников и выведет полученную информацию на экран.

5. Взяв в качестве основы структуру из упражнения 5 главы 4, создайте класс date. Его данные должны размещаться в трех полях типа int: month, day и year. Метод класса getdate() должен принимать значение для объекта в формате 12/31/02, а метод showdate() — выводить данные на экран.

6. Расширьте содержание класса employee из упражнения 4, включив в него класс date и перечисление etype. Объект класса date будет использоваться для хранения даты приема сотрудника на работу. Перечисление будет использовано для хранения статуса сотрудника: лаборант, секретарь, менеджер и т. д. Последние два поля данных должны быть закрытыми в определении класса employee, как и номер и оклад сотрудника. Вам будет необходимо разработать методы getemploy() и putemploy(), предназначенные соответственно для ввода и отображения информации о сотруднике. Возможно, при создании методов вам понадобится ветвление switch для работы с перечисляемым типом etype. Напишите функцию main(), которая попросит пользователя ввести данные о трех сотрудниках, а затем выведет эти данные на экран.

7. Создайте класс fraction. Данные класса должны быть представлены двумя полями: числителем и знаменателем. Методы класса должны получать от пользователя значения числителя и знаменателя дроби в форме 3/5 и выводить значение дроби в этом же формате. Кроме того, должен быть разработан метод, складывающий значения двух дробей. Напишите функцию main(), которая циклически запрашивает у пользователя ввод пары дробей, затем складывает их и выводит результат на экран. После каждой такой операции программа должна спрашивать пользователя, следует ли продолжать цикл.