**//МЕТОД ДОСТУПА PROTECTED l4**

#include <iostream.h>

#include<string>

class CBox

{

public:

CBox(double lv = 1., double wv = 1.0, double hv = 1.0)

: m\_Length(lv), m\_Width(wv), m\_Height(hv)

{

cout << endl << "Constructor CBox";

}

//Деструктор CBox-для отслеживания вызова

**~CBox()**

**{ cout <<"Destructor CBox"<<endl; }**

protected:

double m\_Length;

double m\_Width;

double m\_Height;

};

class CCandyBox : public CBox

{

public:

char \*m\_Contents;

**//функция производного класса, вычисляющая объем**

**double Volume()const**

**{ return m\_Length\* m\_Width\* m\_Height; }**

//Конструктор для установки размеров и содержимого с явным вызовом //конструктора CBox

CCandyBox(double lv, double wv, double hv, char \* str = "Candy")

:CBox(lv, wv, hv)

{

cout << endl << "Constructor2 CCandyBox" << endl;

m\_Contents = new char[strlen(str) + 1];

strcpy(m\_Contents, str);

}

//Конструктор для установки только содержимого, автм. вызывает CBox по //умолчанию

CCandyBox(char\* str = "Candy")

{

cout << endl << "Constructor1 CCandyBox"<<endl;

m\_Contents = new char[strlen(str) + 1];

strcpy(m\_Contents, str);

}

**~CCandyBox()**

**{**

**delete[] m\_Contents;**

**cout <<"Destructor CCandyBox"<<endl;**

**}**

};

//Использование спецификатора доступа protected

int main()

{

CCandyBox myCandyBox;

CCandyBox myToffeeBox(1.0, 2.0, 3.0, "Stickjaw Toffee");

cout << "Volume of myCandyBox equals " << myCandyBox.Volume() << endl

<< "Volume of myToffeeBox equals " << myToffeeBox.Volume()<< endl;

//cout <<endl << myToffeeBox. m\_Length; //ошибка

return 0;

}

 



Результат

Constructor CBox

Constructor1 CCandyBox

Constructor CBox

Constructor2 CCandyBox

Volume of myCandyBox equals 1

Volume of myToffeeBox equals 6

Destructor CCandyBox

Destructor CBox

Destructor CCandyBox

Destructor CBox

