**51. Обектен дизайн : принцип на Лисков.**

Функции които ползват указатели/референции към базов клас, трябва да могат да могат

да ползват обекти от наследени на този клас, без да знаят за това:

Animal \*ptr = new Cat(…);

ptr = new Dog(…);

ptr->speak();

В този случай ще се извика speak() на класът Animal, освен ако той не е виртуален. Ако

е, ще се извика методът на класът Dog наследяващ класа Animal, ако го има.

Това понякога поражда проблеми и за това е препоръчително наследяващия клас само

да разширява базовия клас, без да изменя неговата функционалност/поведение.

Пример: square класът наследява rectangle класът като припокрива ВИРТУАЛНИТЕ

функции:

void square::setWidth(int w) { width = w; height = w; }

void square::setWidth(int w) { width = h; height = h; }

След което ако се извиква следната функция ше се види грешка в логиката:

void f(rectangle \*ptr)

{

ptr->setWidth(5);

ptr->setHeight(10);

ptr->printArea(); //width\*height

}

Ако към функцията се подаде rectangle инстанция, всичко ще бъде наред и ще излезе

50. Но ако подадем square, ще върне 100, което не е очаквания резултат, гледайки го

като rectangle.