Наследяване

Определения

- Наследяване – създаване на нов клас (наречен клас-наследник) на базата на вече съществуващ клас, наречен родителски.
- Класът-наследник наследява всички данни и методи на родителския клас, но може да промени някои от тях, както и да добави нови.
- Множеството от всички родителски класове и техните наследници се нарича йерархия от класове.

```
class Tochka
{ public:
    float x, y;
};

class Krug: public Tochka
{ public:
    float r;
    float Obikolka();
    float Lice();
};

class Prusten: public Krug
{ public:
    float r_otvor;
    float Obikolka();
    float Lice();
};
```

Определения

- Абстракция – събиране на общите свойства на обектите в подходящи родителски класове, така че да е възможно изграждането на йерархия от класове.
- Капсулиране – проектиране на всеки клас така, че да може да бъде използван самостоятелно и да не е възможна нежелана промяна на вътрешните му състояния.
- Скриване на информация – става чрез указване на нивото на достъп до елементите на класа и начина на наследяване на класа.

```
class Tochka
{ public:
   float x, y;
};

class Krug: private Tochka
{public:
   float r;
   float Obikolka();
   float Lice();
};

class Prava: public Tochka
{ private: bool Savpadat;
  public:
   float x2, y2;
   float Dulzina();
};
```

Видимост на елементите на класа

- В C++ е възможно един клас да наследява повече от един родителски класа
- Ако в класът-наследник се дефинират данни или методи със същото име като в някои от родителските класове, те ги скриват и при извикването им се използват тези от наследения клас.
- Ако клас наследява **public** даден родител, то методите и данните на родителския клас са със същата видимост и в класа наследник.
- Ако клас наследява **private** даден родител, то методите и данните на родителския клас са private в класа наследник.
- Методи и данни, дефинирани като **protected** в класа, са видими само в методите на класа и класовете, които са негови наследници, но не и извън тях.

Предимства на ООП

- по-точно отразяват реалният свят – и там имаме класове и обекти
- кодът става по-прегледен и ясен
- улеснява се откриването на грешки – за всеки обект е указано какви методи и данни може да има
- повторно използване на код - общите неща се дефинират в родителския клас
- ускоряват разработката на програми - чрез готови йерархии от класове

Край