Čisté virtuální metody

- pokud není virtuální metoda definována (nemá tělo), tak se jedná o čistou virtuální metodu, která je pouze deklarována
- obsahuje-li objekt čistou virtuální metodu, nemůže být vytvořena jeho instance, může být ale vytvořen jeho ukazatel.

```
deklarace : class base {
....
virtual void fce ( int ) = 0;
}
```

- virtuální metoda nemusí být definována – v tom případě hovoříme o čistě virtuální metodě, musí být deklarována.
- chceme využít jednu třídu jako Bázovou, ale chceme zamezit tomu, aby se s ní pracovalo. Můžeme v konstruktoru vypsat hlášku a existovat. Čistější je ovšem, když na to přijde překladač – tj. použít čistě virtuální metody
- deklarace vypadá: virtual void f (int)=0 (nebo =NULL/nullptr)
- tato metoda se dědí jako čistě virtuální, dokud není definována
- starší překladače vyžadují v odvozené třídě novou deklaraci anebo definici
- obsahuje-li objekt č.v.m. nelze vytvořit jeho instanci, může být ale ukazatel

```
class B {
public: virtual void vf1() {cout << ”bv”; }
        void f()           {cout << ”bn”; }
class C:public B{
        void vf1()    {cout << ”cv”;}   // virtual nepovinné
        void f()     {cout << ”cn”;}}

class D: public B {
        void vf1()     {cout << ”dv”;}
        void f()       {cout << ”dn”;}}

B b; C c;D d;
b.f();  c.f(); d.f();  // vola normální metody třídy
// tisk b c d – protože proměnné jsou typu B C D / překladač
b.vf1(); c.vf1(); d.vf1(); // vola virtuální metody třídy
// tisk b c d – protože proměnné vznikly jako B C D/ runtime
B* bp = &c; // ukazatel na bázovou třídu
// (přiřazení může byt i v ifu, a pak se neví co je dal)
bp->f();  // volá normální metodu třídy B
// tisk b, protože proměnná je typu B / překladač
bp -> vf1(); // vola virtuální metodu
// tisk c – protože proměnná vznikla jako typ c / runtime
```

abstraktní bázové třídy

- Při tvorbě tříd někdy potřebujeme, aby bylo možné pracovat se třídami stejným principem, tj. aby se třídy „zvenku“ chovaly stejně. To je aby jejich část volání měla povinně určité metody. K tomu slouží abstraktní bázový typ, který slouží pouze jako základ pro potomky, ale sám se nevyužívá.
- má alespoň jednu čistou virtuální metodu (C++ přístup)
- neuvažuje se o jejím použití (tvorba objektu)
- obecně třída, která nemá žádnou instanci (objektový přístup)
- slouží jako společná výchozí třída pro potomky
- tvorba rozhraní

```
class X {
  public:
  virtual void f()=0;
  virtual void g()=0;
  void h() ;
}
X b; // nelze

class Y: X {
  void f() {}
}

Y b; opet nelze

class Z:  Y{
  void g(){}
}

Z c; uz jde
c.h() z X
c.f() z Y
c.g() z Z
```