

**Laboratorium 6 Pomoc. Właściwości programowania obiektowego: polimorfizm.**  
**Klasy abstrakcyjne**

**Cel laboratorium:**

Zapoznanie z podstawowymi właściwościami programowania obiektowego - polimorfizmem i wykorzystaniem klas abstrakcyjnych

**Polimorfizm:**

- "wielopostaciowość", "**jeden interfejs – wiele metod**" - pozwala **jednemu interfejsowi** na podejmowanie **odpowiedniego działania**; komunikat wysyłany do obiektu może powodować wykonanie **różnych akcji** w zależności od klasy obiektu
- właściwość **wyboru metody na podstawie typu obiektu** w czasie wykonania programu (późne wiązanie) => typ obiektu określa wersję metody na etapie wykonania
- przesłanianie **alternatywnych procedur wspólnym sposobem wywołania**, możliwość **dynamicznego wyboru metody** (późne wiązanie) wykonywanej na obiekcie po skierowaniu do niego komunikatu zawierającego nazwę tej metody
- jest realizowany za pomocą **metod wirtualnych**, tzn. metod wywoływanych za pomocą **późnego wiązania**, czyli dopiero w czasie wykonywania programu

**Metoda wirtualna** w klasie bazowej zakłada możliwość jej redefinicji w klasie potomka:

- Nazwy identyczne, implementacja inna
- Prototyp metody poprzedzony słowem **virtual** w przodku
- Prototyp metody **bez (lub z) słowa virtual** w potomku (mało czytelne)

```
class TBazowa
{public:
    void metoda();//metoda zwykła
};
class TPotomna: public TBazowa
{ public:
    void metoda();//redefiniowanie metody - przesłanianie
};
```

```
class TBazowa
{public:
    virtual void metoda();//metoda wirtualna
};
class TPotomna: public TBazowa
{ public:
    void metoda();//redefiniowanie metody - polimorfizm
};
```

- Destraktor w klasie bazowej powinien być metodą wirtualną
- Obiekt klasy z metodami wirtualnymi jest większy
- Wykonanie programu z metodami wirtualnymi trwa dłużej

Przykład 1:

```
class TSkladowa{
};
```

```
class TBazowa{
public:
    void metoda1() {cout<<"metoda 1 w
TBazowa"<<endl;}; //zwykła
    virtual void metoda2(){cout<<"metoda 2 w TBazowa"<<endl;
        }; //wirtualna
    void wynik(){cout<<"wynik: "; metoda1();metoda2();}
    TBazowa(){cout<<"tworze obiekt TBazowa"<<endl;}
    ~TBazowa(){cout<<"usuwa obiekt TBazowa"<<endl;}; //zle
    //virtual ~TBazowa(){cout<<"usuwa obiekt TBazowa"<<endl;}; //dobrze
};
```

```
class TPotomna: public TBazowa{
public:
    void metoda1() {cout<<"metoda 1 w TPotomna"<<endl;};
    // przesłanianie zwykłej
    void metoda2(){cout<<"metoda 2 w TPotomna"<<endl; }
    //przesłanianie wirtualnej
    TPotomna(){ oS=new TSkladowa;cout<<"tworze obiekt
TPotomna "
        cout<<"tworze obiekt TSkladowa"<<endl;}
    ~TPotomna(){delete oS;cout<<"usuwa obiekt TPotomna "
        cout<<"usuwa obiekt TSkladowa"<<endl;}; //zle
    //virtual ~TPotomna(){delete oS;cout<<"usuwa obiekt TPotomna "
    //cout<<"usuwa obiekt TSkladowa"<<endl;}; //dobrze
protected:
    TSkladowa* oS;
};
```

```
int main(int argc, char** argv) {
cout<<"praca przez zmienna"<<endl;
```

```
    TBazowa oB;
    oB.metoda1();
    oB.metoda2();
    oB.wynik();
```

```
cout<<endl;
```

```
    TPotomna oP;
    oP.metoda1();
    oP.metoda2();
    oP.wynik();
```

```
cout<<endl;
```

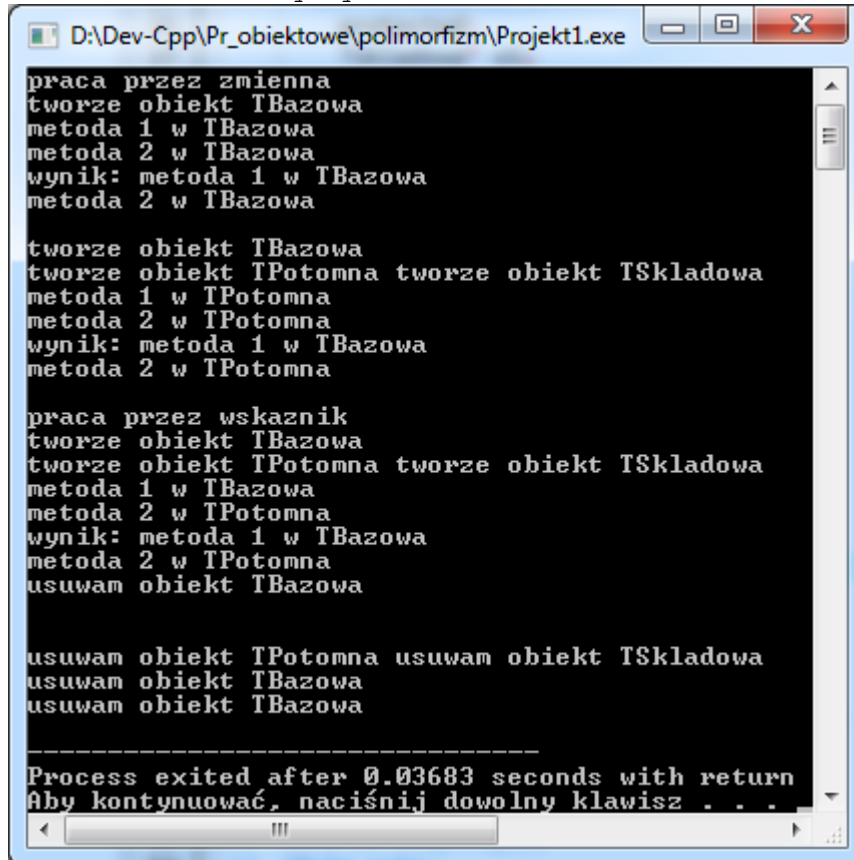
```
cout<<"praca przez wskaznik"<<endl;
```

```
    TBazowa* w; //TPotomna* w;
    w=new TPotomna; //w=&oP;
    w->metoda1(); //wczesne wiązanie metody
    w->metoda2(); //późne wiązanie metody
    w->wynik();
    delete w; // destruktor nie wirtualny - wyciek pamięci - źle
```

```
cout<<endl;
```

```
return 0;}
```

Uruchomienie ze zwykłym destrukтором



```

D:\Dev-Cpp\Pr_obiektowe\polimorfizm\Projekt1.exe
praca przez zmienna
tworze obiekt TBazowa
metoda 1 w TBazowa
metoda 2 w TBazowa
wynik: metoda 1 w TBazowa
metoda 2 w TBazowa

tworze obiekt TBazowa
tworze obiekt IPotomna tworze obiekt TSkładowa
metoda 1 w IPotomna
metoda 2 w IPotomna
wynik: metoda 1 w TBazowa
metoda 2 w IPotomna

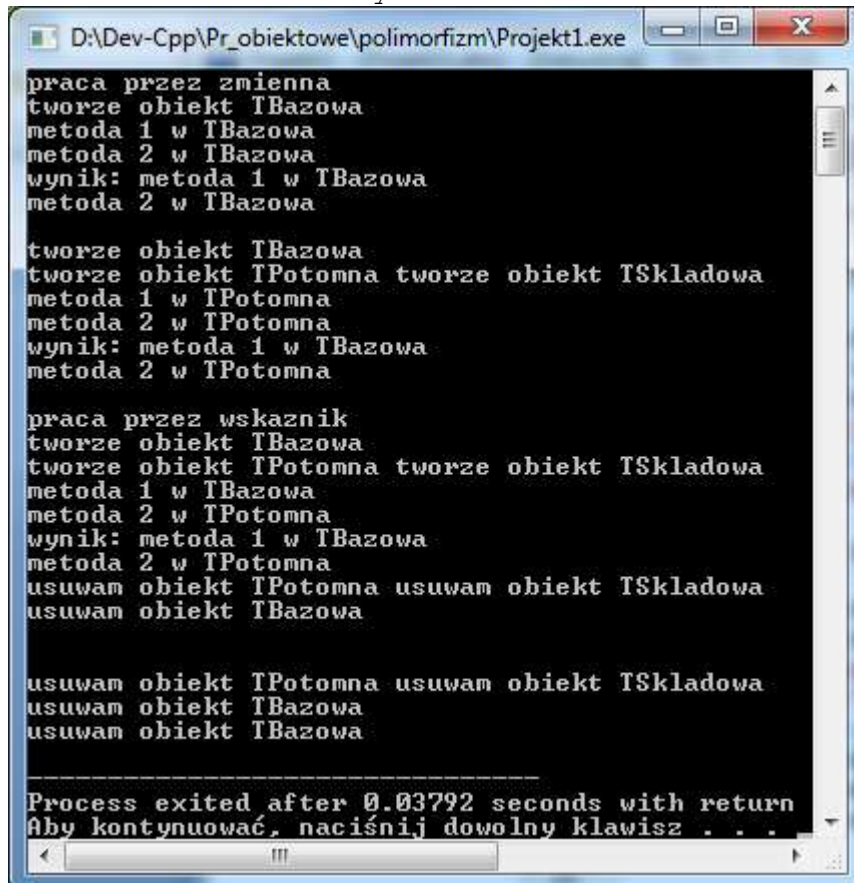
praca przez wskaźnik
tworze obiekt TBazowa
tworze obiekt IPotomna tworze obiekt TSkładowa
metoda 1 w TBazowa
metoda 2 w IPotomna
wynik: metoda 1 w TBazowa
metoda 2 w IPotomna
usuwa obiekt TBazowa

usuwa obiekt IPotomna usuwa obiekt TSkładowa
usuwa obiekt TBazowa
usuwa obiekt TBazowa

-----
Process exited after 0.03683 seconds with return
Aby kontynuować, naciśnij dowolny klawisz . . .

```

uruchomienie z wirtualnym destrukтором



```

D:\Dev-Cpp\Pr_obiektowe\polimorfizm\Projekt1.exe
praca przez zmienna
tworze obiekt TBazowa
metoda 1 w TBazowa
metoda 2 w TBazowa
wynik: metoda 1 w TBazowa
metoda 2 w TBazowa

tworze obiekt TBazowa
tworze obiekt IPotomna tworze obiekt TSkładowa
metoda 1 w IPotomna
metoda 2 w IPotomna
wynik: metoda 1 w TBazowa
metoda 2 w IPotomna

praca przez wskaźnik
tworze obiekt TBazowa
tworze obiekt IPotomna tworze obiekt TSkładowa
metoda 1 w TBazowa
metoda 2 w IPotomna
wynik: metoda 1 w TBazowa
metoda 2 w IPotomna
usuwa obiekt IPotomna usuwa obiekt TSkładowa
usuwa obiekt TBazowa

usuwa obiekt IPotomna usuwa obiekt TSkładowa
usuwa obiekt TBazowa
usuwa obiekt TBazowa

-----
Process exited after 0.03792 seconds with return
Aby kontynuować, naciśnij dowolny klawisz . . .

```

- **Klasa abstrakcyjna** to klasa zawierająca metody czysto wirtualne, służy jako klasa bazowa dla innych klas potomnych: nie posiada instancji (obiektów), można tworzyć do niej wskaźniki, nazwa zaczyna się od litery I
- Aby klasa była abstrakcyjna to musi mieć przynajmniej jedną metodę czysto wirtualną - czyli metodę wirtualną **która nie ma ciała**.
- Metody **czysto wirtualne**, inaczej **abstrakcyjne** (pure virtual) to metody bez implementacji przeznaczone wyłącznie do predefiniowania w klasach potomnych, używane są do tworzenia **interfejsów** (zbiór prototypów metod - nazwy, parametry, typ), wymuszają jednolitość nazw i parametrów metod w klasach potomnych
- Możliwa jest deklaracja wskaźnika na obiekt klasy abstrakcyjnej, utworzenie obiektu klasy potomnej lub przypisanie wskaźnikowi obiektu klasy potomnej

```
class Iklasa //klasa abstrakcyjna
{protected:
    virtual void metoda1()=0; //metoda czysto wirtualna
    virtual void metoda2()=0; // metoda czysto wirtualna
...};
```

```
class Tklasa: public Iklasa {...};
```

```
.....
Iklasa* wsk= new Tklasa; //
wsk->metoda1();
delete wsk;
```

### Przykład 2

```
class Izwierze //klasa abstrakcyjna - tworzenie interfejsu
{protected:
    string nazwa;
public:
    virtual string dajGlos()=0; //czysto wirtualna
    virtual void jedz(string karma, float ilosc)=0
    void setNazwa(string n="nieznane"){nazwa=n;};
    string getNazwa(){return nazwa;};
};
```

```
class Tpies: public Izwierze{
public:
    string dajGlos(){return "hau hau"; }
    void jedz(string karma, float ilosc){
        for (int i=0; i<3;i++){cout<<"karmienie "
<<i+1<<": "<<karma<<" "<<ilosc<<" gram"<<endl;}}
};
```

```
class Twaz:public Izwierze{
public:
    string dajGlos(){return "ssssssssssssssssss";};
    void jedz(string karma, float ilosc){cout<<"w tym
tygodniu: "<<karma<<" "<<ilosc<<" kg"<<endl; }
};
```

```
int main(int argc, char** argv) {
```

```
    //Izwierze obiekt;//nie mozna
```

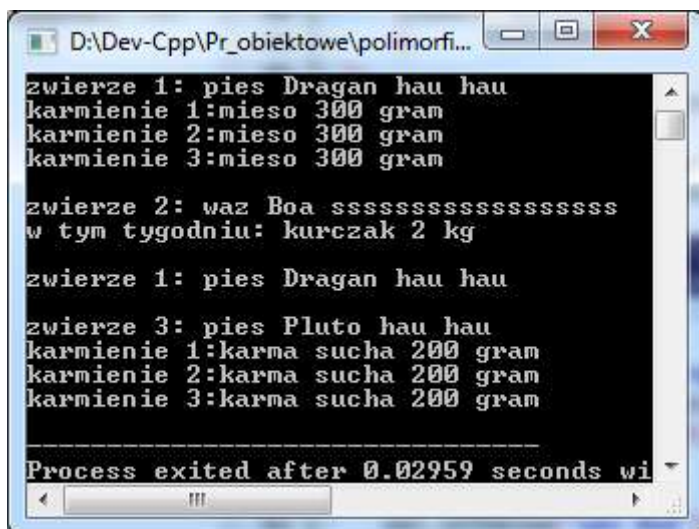
```
    Tpies pies;
    pies.setNazwa("pies Dragan");
    cout<<"zwierze 1: "<<pies.getNazwa()<<" ";
    cout<<pies.dajGlos()<<endl;
    pies.jedz("mieso", 300);
    cout<<endl;
```

```
    Twaz waz;
    waz.setNazwa("waz Boa");
    cout<<"zwierze 2: "<<waz.getNazwa()<<" ";
    cout<<waz.dajGlos()<<endl;
    waz.jedz("kurczak", 2);
    cout<<endl;
```

```
    Izwierze* wsk;
    wsk=&pies; //mozna
    cout<<"zwierze 1: "<<wsk->getNazwa()<<" ";
    cout<<wsk->dajGlos()<<endl;
    cout<<endl;
```

```
    wsk= new Tpies;//mozna
    wsk->setNazwa("pies Pluto");
    cout<<"zwierze 3: "<<wsk->getNazwa()<<" ";
    cout<<wsk->dajGlos()<<endl;
    wsk->jedz("karma sucha", 200);
    delete wsk;
```

```
    return 0;
}
```



```
D:\Dev-Cpp\Pr_obiektowe\polimorfi...
zwierze 1: pies Dragan hau hau
karmienie 1:mieso 300 gram
karmienie 2:mieso 300 gram
karmienie 3:mieso 300 gram

zwierze 2: waz Boa sssssssssssssssssss
w tym tygodniu: kurczak 2 kg

zwierze 1: pies Dragan hau hau

zwierze 3: pies Pluto hau hau
karmienie 1:karma sucha 200 gram
karmienie 2:karma sucha 200 gram
karmienie 3:karma sucha 200 gram

-----
Process exited after 0.02959 seconds wi
```