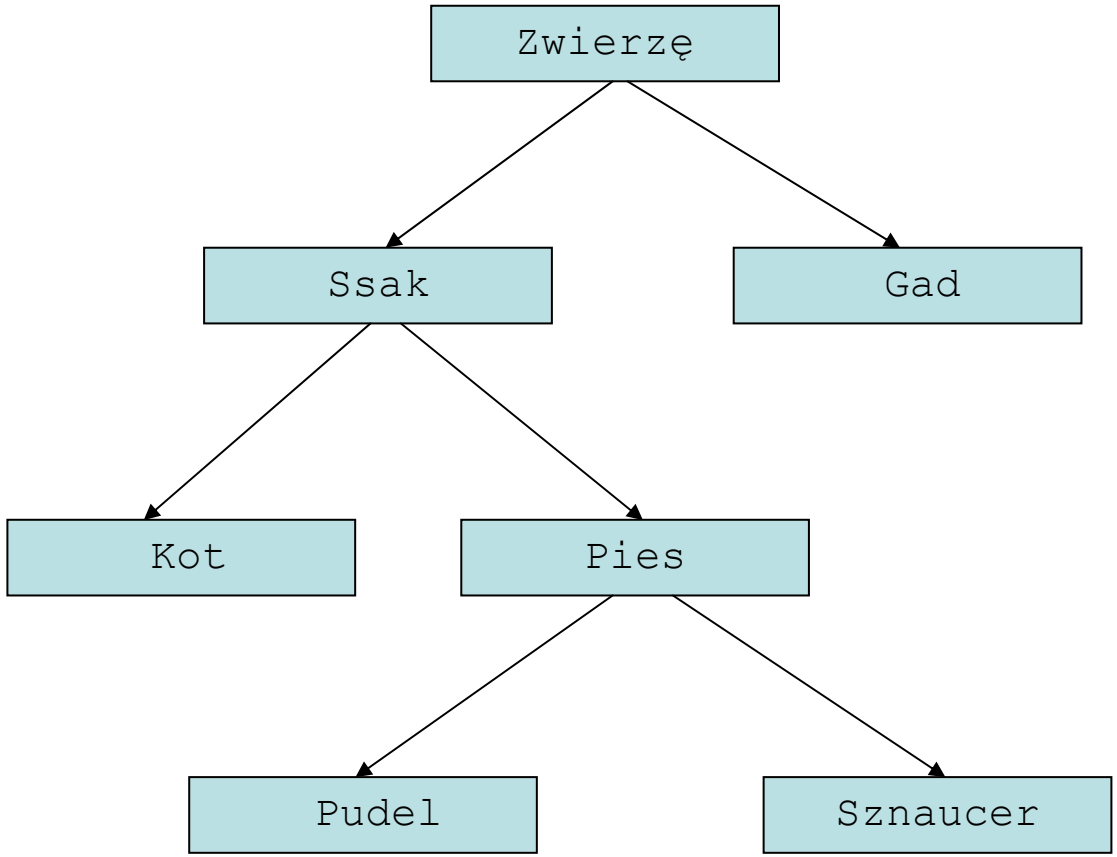
Dziedziczenie



Mechanizm pozwalający na wyprowadzanie obiektów klas pochodnych z klas bazowych

Klasa bazowa – klasa na podstawie które tworzymy nową klasę.

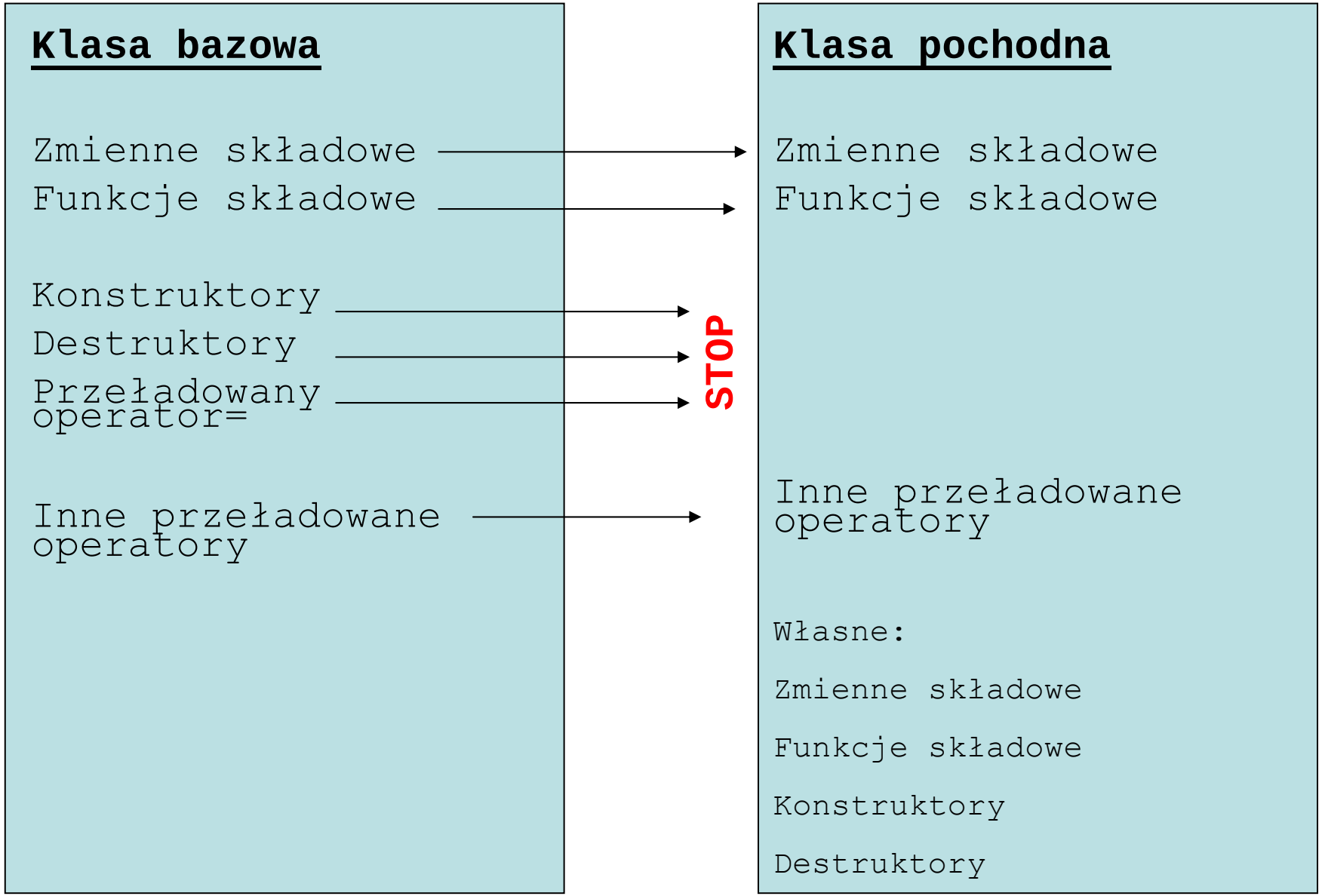
Klasa A → Klasa B →Klasa C

A – bezpośrednia kasa bazowa dla B

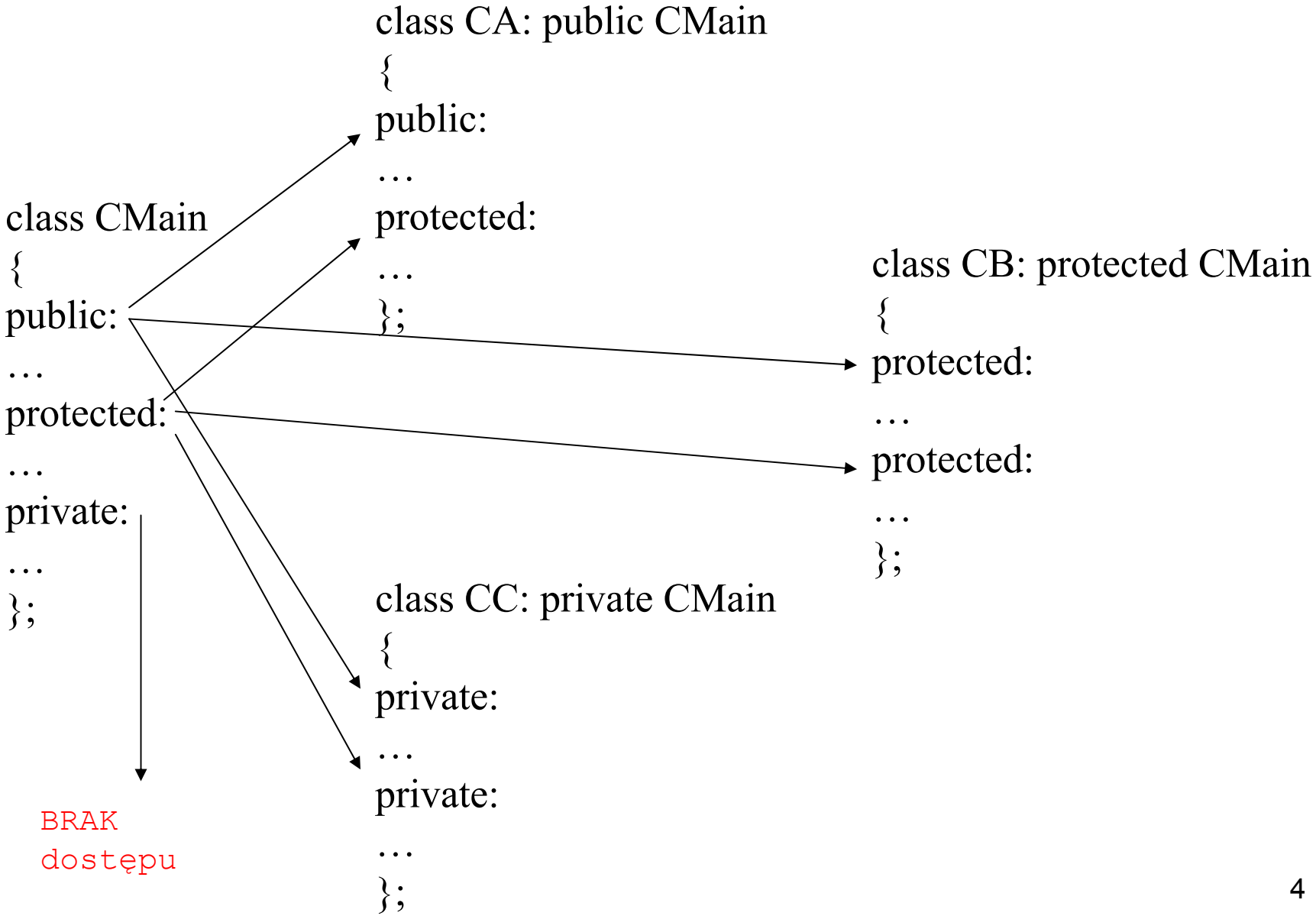
B – bezpośrednia kasa bazowa dla C

A – pośrednia kasa bazowa dla C

Dziedziczenie w klasach



Poziom dostępu do dziedziczonych składowych klasy



```
// Dziedziczenie

#include<iostream>
using namespace std;

class Owad
{
public:
	int waga;
	bool jadowitosc;

	Owad():waga(1),jadowitosc(false)
	{cout<<"Konstruktor domyslny owada\n";}

	Owad (bool j):waga(2),jadowitosc(j)
	{cout<<"Konstruktor owada\n";}

	~Owad()
	{cout<<"Destruktor owada\n";}

	void Ruch()
	{cout<<"Owad lata\n";}
};
```

```
class Komar : public Owad
{
public:
	bool brudas;

	Komar(): Owad(true),brudas(false)
	{cout<<"Konstruktor komara\n";}

	void Aktywnosc()
	{cout<<"Brzeczy i gryzie\n";}

	~Komar ()
	{cout<<"Komar rozgnieciony na scianie\n";}
};
```

```
class Mucha : public Owad
{
public:
	bool brudas;

	Mucha(): brudas(true)
	{cout<<"Konstruktor muchy\n";}

	void Aktywnosc()
	{cout<<"Lata i brzeczy\n";}

	~Mucha ()
	{cout<<"Mucha dostala w beret\n";}
};
```

```
int main()
{
	cout<<"Najpierw mucha :\n";
	Mucha mucha;
	cout<<mucha.waga<<endl;
	cout<<mucha.jadowitosc<<endl;
	cout<<mucha.brudas<<endl;
	mucha.Ruch();
	mucha.Aktywnosc();
	mucha.~Mucha();
	cout<<endl;
	cout<<"Kolej na komara\n";
	Komar komar;
	cout<<komar.waga<<endl;
	cout<<komar.jadowitosc<<endl;
	cout<<komar.brudas<<endl;
	komar.Ruch();
	komar.Aktywnosc();
	komar.~Komar();
	system("pause");
	return 0;
}
```

```
Najpierw mucha :
Konstruktor domyslny owada
Konstruktor muchy
1
0
1
Owad lata
Lata i brzeczy
Mucha dostala w beret
Destruktor owada

Kolej na komara
Konstruktor owada
Konstruktor komara
2
1
0
Owad lata
Brzeczy i gryzie
Komar rozgnieciony na scianie
Destruktor owada
Press any key to continue . . .
```

```
// Przekazywanie elementów
// do klas

#include<iostream>
#include<string>
using namespace std;
string Rasa[6]=
{"Golden", "Spaniel", "Labrador", "Pudel", "Terier" ,
"Beaggle"};

class Ssak
{public:

Ssak();

Ssak(int age);

void PodajWiek() {cout<<"Mam "<<wiek<<" lat\n";}
void UstalWiek(int age) {wiek = age;}

void Spij() {cout<<"Wlasnie spie\n";}
```

```
protected:
	int wiek;
	int waga;
};

class Pies : public Ssak
{
public:
	Pies();
	Pies(int age);
	Pies(int age, int weight);
	Pies(int age, int weight, int rasa);

	string PodajRase() {return rasa_psa;}
	void WpiszRase(int rasa) {rasa_psa = Rasa[rasa];}

	void Radosc() {cout<<"Macham ogonem\n";}

private:
	string rasa_psa;
};
Ssak::Ssak():wiek(2),waga(5)
{ cout<<"Konstruktor 1 klasy Ssak "<<endl; }
```

```
Ssak::Ssak(int age)
{       wiek=age;
        waga=5;
        cout<<"Konstruktor 2 klasy Ssak "<<endl; }

Pies::Pies():Ssak(),rasa_psa(Rasa[5])
{ cout<<"Konstruktor 1 klasy Pies "<<endl; }

Pies::Pies(int age):Ssak(age), rasa_psa(Rasa[3])
{ cout<<"Konstruktor 2 klasy Pies "<<endl; }

/*Pies::Pies(int age): wiek(age), rasa_psa(Rasa[3]){ }*/

Pies::Pies(int age, int weight): Ssak(age), rasa_psa(Rasa[0])
{ waga=weight;
        cout<<"Konstruktor 3 klasy Pies "<<endl; }

Pies::Pies(int age, int weight, int rasa):Ssak(age),
rasa_psa(Rasa[rasa])
{ waga =weight;
        cout<<"Konstruktor 4 klasy Pies "<<endl; }
```

```
int main()
{
	Pies Reks;
	Pies Azor(5);
	Pies Ciapek(6,0);
	Pies Kastor(15,15,1);
	Pies Maks(4,4);
	Maks.Radosc();
	cout<<"Azor : ";
	Azor.PodajWiek();
	cout<<"Ciapek : ";
	Ciapek.PodajRase();
	Ciapek.PodajWiek();
	Ciapek.Radosc();
	cout<<"Azor : ";
	Azor.Spij();
	cout<<"Kastor :"<<Kastor.PodajRase()<<endl;
	Kastor.PodajWiek();

	system("pause");
	return 0;
}
```

```
Konstruktor 1 klasy Ssak
Konstruktor 1 klasy Pies
Konstruktor 2 klasy Ssak
Konstruktor 2 klasy Pies
Konstruktor 2 klasy Ssak
Konstruktor 3 klasy Pies
Konstruktor 2 klasy Ssak
Konstruktor 4 klasy Pies
Konstruktor 2 klasy Ssak
Konstruktor 3 klasy Pies
Macham ogonem
Azor : Mam 5 lat
Ciapek : Mam 6 lat
Macham ogonem
Azor : Wlasnie spie
Kastor :Spaniel
Mam 15 lat
Press any key to continue . . .
```

```
// protected

#include<iostream>
using namespace std;

class Licznik
{
protected:
	int zmienna;
public:
	Licznik()
	{zmienna=0;}
	void Show()
	{cout<<"zmienna = "<<zmienna<<endl;}
};

class Zmien: public Licznik
{public:
void Dodaj()
{zmienna++;}
void Pokaz()
	{cout<<"zmienna = "<<zmienna<<endl;}
};
```

```
int main()
{
	Licznik obiekt;
	Zmien obiekt1;
	obiekt1.Dodaj();
	obiekt.Show();
	obiekt1.Show();
	obiekt1.Pokaz();
	system("pause");
	return 0;
}
```

```
zmienna = 0
zmienna = 1
zmienna = 1
Press any key to continue . . .
```

```
class Licznik
{
	int zmienna;//Private
public:
	Licznik()
	{zmienna=0;}
	void Show()
	{cout<<"zmienna = "<<zmienna<<endl;}
};
```

```
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_07_04\w_07_04\ffdfdd.cpp(19) : error C2248: 'Licznik::zmienna' : cannot access
private member declared in class 'Licznik'
1>        e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_07_04\w_07_04\ffdfdd.cpp(8) : see declaration of 'Licznik::zmienna'
1>        e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_07_04\w_07_04\ffdfdd.cpp(7) : see declaration of 'Licznik'
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_07_04\w_07_04\ffdfdd.cpp(21) : error C2248: 'Licznik::zmienna' : cannot access
private member declared in class 'Licznik'
1>        e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_07_04\w_07_04\ffdfdd.cpp(8) : see declaration of 'Licznik::zmienna'
1>        e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_07_04\w_07_04\ffdfdd.cpp(7) : see declaration of 'Licznik'
1>Build log was saved at "file://e:\Praca\Dydaktyka\Programowanie obiektowe\Visual
Project\Wykład 2016\W_07_04\W_07_04\Debug\BuildLog.htm"
1>W_07_04 - 2 error(s), 0 warning(s)
========== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped ==========
```

```
// protected 2

#include<iostream>
using namespace std;

class Licznik
{
protected:
	int zmienna;
public:
	Licznik()
	{zmienna=0;}
	void Show()
	{cout<<"zmienna = "<<zmienna<<endl;}
};

class Zmien: protected Licznik
{public:
void Dodaj()
{zmienna++;}
void Pokaz()
	{cout<<"zmienna = "<<zmienna<<endl;}
};
```

```
int main()
{
	Licznik obiekt;
	Zmien obiekt1;
	obiekt1.Dodaj();
	//obiekt.Show();
	//obiekt1.Show();
	obiekt1.Pokaz();
	system("pause");
	return 0;
}
```

```
zmienna = 1
Press any key to continue . . .
```

```
// Dziedziczenie public

#include<iostream>
using namespace std;

class A
{
	char* zm_priv_A;
protected:
	char* zm_prot_A;
public:
	char* zm_publ_A;

	A():zm_priv_A("PRIVATE_A"),
		zm_prot_A("PROTECTED_A"),
		zm_publ_A("PUBLIC_A"){}
	void Pokaz_priv_A()
	{cout<<zm_priv_A<<endl;}
	void Pokaz_prot_A()
	{cout<<zm_prot_A<<endl;}
	void Pokaz_publ_A()
	{cout<<zm_publ_A<<endl;}
};
```

```
class B : public A
        {
        char* zm_priv_B;
protected:
        char* zm_prot_B;
public:
        char* zm_publ_B;

        B():A(),
                zm_priv_B("PRIVATE_B"),
                zm_prot_B("PROTECTED_B"),
                zm_publ_B("PUBLIC_B"){}

        void Pokaz_priv_B()
        {cout<<zm_priv_B<<endl;}
        void Pokaz_prot_B()
        {cout<<zm_prot_B<<endl;}
        void Pokaz_publ_B()
        {cout<<zm_publ_B<<endl;}
        /*void Pokaz_priv_AB(A a)
        {cout<<a.zm_priv_A<<endl;}
        void Pokaz_prot_AB(A a)
        {cout<<a.zm_prot_A<<endl;}*/
```

```
        void Pokaz_publ_AB(A a)
        {cout<<a.zm_publ_A<<endl;}
};

void main()
{
        A obiekt_A;
        cout<<"Rozmiar obiektu klasy A = "<<sizeof(obiekt_A)<<"
bajtow\n";
        B obiekt_B;
        cout<<"Rozmiar obiektu klasy B = "<<sizeof(obiekt_B)<<"
bajtow\n";
        cout<<"\nFunkcje z klasy A\n";
        obiekt_A.Pokaz_priv_A();
        obiekt_A.Pokaz_prot_A();
        obiekt_A.Pokaz_publ_A();
        cout<<"\nFunkcje z klasy B\n";
        obiekt_B.Pokaz_priv_A();
        obiekt_B.Pokaz_prot_A();
        obiekt_B.Pokaz_publ_A();
        obiekt_B.Pokaz_priv_B();
        obiekt_B.Pokaz_prot_B();
        obiekt_B.Pokaz_publ_B();
```

```
        cout<<"\nDostep z klasy B do A\n";
        //obiekt_B.Pokaz_priv_AB(obiekt_A);
        //obiekt_B.Pokaz_prot_AB(obiekt_A);
        obiekt_B.Pokaz_publ_AB(obiekt_A);
        cout<<"\nBezposredni dostep do skladowych\n";
        cout<<obiekt_B.zm_publ_A<<endl;
        cout<<obiekt_B.zm_publ_B<<endl;

        system("pause");
}
```

```
Rozmiar obiektu klasy A = 12 bajtow
Rozmiar obiektu klasy B = 24 bajtow

Funkcje z klasy A
PRIVATE_A
PROTECTED_A
PUBLIC_A

Funkcje z klasy B
PRIVATE_A
PROTECTED_A
PUBLIC_A
PRIVATE_B
PROTECTED_B
PUBLIC_B

Dostep z klasy B do A
PUBLIC_A

Bezposredni dostep do skladowych
PUBLIC_A
PUBLIC_B
Press any key to continue . . .
```

```
// Dziedziczenie protected

#include<iostream>
using namespace std;

class A
{
	char* zm_priv_A;
protected:
	char* zm_prot_A;
public:
	char* zm_publ_A;

	A():zm_priv_A("PRIVATE_A"),
		zm_prot_A("PROTECTED_A"),
		zm_publ_A("PUBLIC_A"){}
	void Pokaz_priv_A()
	{cout<<zm_priv_A<<endl;}
	void Pokaz_prot_A()
	{cout<<zm_prot_A<<endl;}
	void Pokaz_publ_A()
	{cout<<zm_publ_A<<endl;}
};
```

```
class B : protected A
        {
        char* zm_priv_B;
protected:
        char* zm_prot_B;
public:
        char* zm_publ_B;

        B():A(),
                zm_priv_B("PRIVATE_B"),
                zm_prot_B("PROTECTED_B"),
                zm_publ_B("PUBLIC_B"){}

        void Pokaz_priv_B()
        { //cout<<zm_priv_A<<"\t"<<zm_priv_B<<endl;
        cout<<zm_priv_B<<endl;}
        void Pokaz_prot_B()
        {cout<<zm_prot_A<<"\t"<<zm_prot_B<<endl;}
        void Pokaz_publ_B()
        {cout<<zm_publ_A<<"\t"<<zm_publ_B<<endl;}
```

```
        /*void Pokaz_priv_AB(A a)
        {cout<<a.zm_priv_A<<endl;}
        void Pokaz_prot_AB(A a)
        {cout<<a.zm_prot_A<<endl;}*/
        void Pokaz_publ_AB(A a)
        {cout<<a.zm_publ_A<<endl;}
};

void main()
{
        A obiekt_A;
        B obiekt_B;
        cout<<"Funkcje z klasy A\n";
        obiekt_A.Pokaz_priv_A();
        obiekt_A.Pokaz_prot_A();
        obiekt_A.Pokaz_publ_A();
        cout<<"\nFunkcje z klasy B\n";
        //obiekt_B.Pokaz_priv_A();
        //obiekt_B.Pokaz_prot_A();
        //obiekt_B.Pokaz_publ_A();
        obiekt_B.Pokaz_priv_B();
        obiekt_B.Pokaz_prot_B();
        obiekt_B.Pokaz_publ_B();
```

```
        cout<<"\nDostep z klasy B do A\n";
        //obiekt_B.Pokaz_priv_AB(obiekt_A);
        //obiekt_B.Pokaz_prot_AB(obiekt_A);
        obiekt_B.Pokaz_publ_AB(obiekt_A);
        cout<<"\nBezposredni dostep do skladowych\n";
        //cout<<obiekt_B.zm_publ_A<<endl;
        cout<<obiekt_B.zm_publ_B<<endl;

        system("pause");
}
```

```
Funkcje z klasy A
PRIVATE_A
PROTECTED_A
PUBLIC_A

Funkcje z klasy B
PRIVATE_B
PROTECTED_A     PROTECTED_B
PUBLIC_A        PUBLIC_B

Dostep z klasy B do A
PUBLIC_A

Bezposredni dostep do skladowych
PUBLIC_B
Press any key to continue . . .
```

```
// Dziedziczenie private

#include<iostream>
using namespace std;

class A
{
	char* zm_priv_A;
protected:
	char* zm_prot_A;
public:
	char* zm_publ_A;

	A():zm_priv_A("PRIVATE_A"),
		zm_prot_A("PROTECTED_A"),
		zm_publ_A("PUBLIC_A"){}
	void Pokaz_priv_A()
	{cout<<zm_priv_A<<endl;}
	void Pokaz_prot_A()
	{cout<<zm_prot_A<<endl;}
	void Pokaz_publ_A()
	{cout<<zm_publ_A<<endl;}
};
```

```
class B : A
        {
        char* zm_priv_B;
protected:
        char* zm_prot_B;
public:
        char* zm_publ_B;

        B():A(),
                zm_priv_B("PRIVATE_B"),
                zm_prot_B("PROTECTED_B"),
                zm_publ_B("PUBLIC_B"){}

        void Pokaz_priv_B()
        { //cout<<zm_priv_A<<"\t"<<zm_priv_B<<endl;
        cout<<zm_priv_B<<endl;}
        void Pokaz_prot_B()
        {cout<<zm_prot_A<<"\t"<<zm_prot_B<<endl;}
        void Pokaz_publ_B()
        {cout<<zm_publ_A<<"\t"<<zm_publ_B<<endl;}
```

```
        /*void Pokaz_priv_AB(A a)
        {cout<<a.zm_priv_A<<endl;}
        void Pokaz_prot_AB(A a)
        {cout<<a.zm_prot_A<<endl;}*/
        void Pokaz_publ_AB(A a)
        {cout<<a.zm_publ_A<<endl;}
};

void main()
{
        A obiekt_A;
        B obiekt_B;
        cout<<"Funkcje z klasy A\n";
        obiekt_A.Pokaz_priv_A();
        obiekt_A.Pokaz_prot_A();
        obiekt_A.Pokaz_publ_A();
        cout<<"\nFunkcje z klasy B\n";
        //obiekt_B.Pokaz_priv_A();
        //obiekt_B.Pokaz_prot_A();
        //obiekt_B.Pokaz_publ_A();
        obiekt_B.Pokaz_priv_B();
        obiekt_B.Pokaz_prot_B();
        obiekt_B.Pokaz_publ_B();
```

```
cout<<"\nDostep z klasy B do A\n";
        //obiekt_B.Pokaz_priv_AB(obiekt_A);
        //obiekt_B.Pokaz_prot_AB(obiekt_A);
        obiekt_B.Pokaz_publ_AB(obiekt_A);
        cout<<"\nBezposredni dostep do skladowych\n";
        //cout<<obiekt_B.zm_publ_A<<endl;
        cout<<obiekt_B.zm_publ_B<<endl;

        system("pause");
}
```

```
Funkcje z klasy A
PRIVATE_A
PROTECTED_A
PUBLIC_A

Funkcje z klasy B
PRIVATE_B
PROTECTED_A     PROTECTED_B
PUBLIC_A        PUBLIC_B

Dostep z klasy B do A
PUBLIC_A

Bezposredni dostep do skladowych
PUBLIC_B
Press any key to continue . . .
```

```
// Dziedziczenie private protected public
#include<iostream>
#include<string>
using namespace std;

class Armia
{
	string aaaa, Adres;
protected:
	string Data;
public:
	string Imie;
	string Nazwisko;

	Armia()
	{
		cout<<„Armia\nImie : ";
		cin>>aaaa;
		Imie=aaaa;

		cout<<"Nazwisko : ";
		cin>>aaaa;
		Nazwisko=aaaa;
```

```
                cout<<"Data urodzenia : ";
                cin>>aaaa;              Data=aaaa;

                cout<<"Adres : ";
                cin>>aaaa;              Adres=aaaa;
                cout<<endl;
        }
        string GetDate() const
        {return this->Data;}
};

class Ladowe: public Armia
{
string aaaa;
protected:
        string Przydzial;
public:
        string stopien;
        Ladowe()
        {
                cout<<„Ladowe\nStopien : ";
                cin>>aaaa;              stopien=aaaa;
```

```
                cout<<"Przydzial : ";
                cin>>aaaa;              Przydzial=aaaa;
                cout<<endl;
        }
};

class SilySpecjalne :protected Armia
{
string aaaa;
string Przydzial;
protected:
        string stopien;
public:
        string Pseudo;
        SilySpecjalne()
        {
                cout<<„Sily specjalne\nStopien : ";
                cin>>aaaa;              stopien=aaaa;

                cout<<"Przydzial : ";
                cin>>aaaa;              Przydzial=aaaa;
```

```
                cout<<"Pseudo : ";
                cin>>aaaa;              cout<<endl;
                Pseudo=aaaa;
        }  };

class Wywiad: Armia
{
        string aaaa;
        string Kryptonim, stopien, Przydzial;
public:
        Wywiad()
        {
                cout<<„Wywiad\nStopien : ";
                cin>>aaaa;              stopien=aaaa;

                cout<<"Przydzial : ";
                cin>>aaaa;              Przydzial=aaaa;

                cout<<"Kryptonim : ";
                cin>>aaaa;              cout<<endl;
                Kryptonim=aaaa;
        }};
```

```
int main()
{
	Armia zolnierz;
	Ladowe czolgista;
	SilySpecjalne snajper;
	Wywiad agent;
	cout<<zolnierz.Imie<<endl;
	cout<<zolnierz.Nazwisko<<endl;
	//zolnierz.Data;
	//zolnierz.Adres;
	cout<<zolnierz.GetDate()<<endl;
	cout<<czolgista.Imie<<endl;
	cout<<czolgista.Nazwisko<<endl;
	cout<<czolgista.stopien<<endl;
	cout<<snajper.Pseudo<<endl;
	//agent.Kryptonim;

	system("pause");
	return 0;
}
```

```
Armia :
Imie : Jan
Nazwisko : Kowalski
Data urodzenia : xxxx
Adres : cccc

Armia :
Imie : Janek
Nazwisko : Kos
Data urodzenia : zzz
Adres : vvv

Ladowe
Stopien : sierzant
Przydzial : Rudy-102

Armia :
Imie : Piotr
Nazwisko : Nowak
Data urodzenia : vvv
Adres : sss
```

```
Sily specjalne
Stopien : porucznik
Przydzial : snajper
Pseudo : Killer

Armia :
Imie : Hans
Nazwisko : Kloss
Data urodzenia : ccc
Adres : sss

Wywiad
Stopien : hauptmann
Przydzial : Abwehra
Kryptonim : J-23

Jan
Kowalski
xxxx
Janek
Kos
sierzant
Killer
Press any key to continue . . .
```