```
//Dostęp do składowych klasy

#include <iostream>
#include <string>
using namespace std;

class Car
{
	string Numer_VIN;

protected:
	string Marka;

public:
	string Model;

	Car(string VIN, string marka, string model)
	{
		Numer_VIN=VIN;
		Marka=marka;
		Model=model;
	}
```

```
        Car():Marka("Honda"),Model("Accord"){}
        Car(string VIN):Marka("Honda"),Model("Accord")
        {Numer_VIN=VIN;}

        string GetMarka()
        {return Marka;}

        int SetMarka(string mark)
        {Marka=mark;
        return 0;}

        string GetVIN()
        {return Numer_VIN;}

        int SetVIN(string numer)
        {Numer_VIN=numer;
        return 0;}
};
```

```
int main(void)
{
	Car car1=Car("VNM1234","Opel","Corsa");
	Car car2;
	Car car3("KLMN1234");
	car2.SetMarka("Polonez");
	car2.Model="Caro";
	car2.SetVIN("MKK8976");
	//car2.Marka="Polonez";
	//car1.Numer_VIN=„ABCD123456";
cout<<car1.GetMarka()<<"\t"<<car1.Model<<"\t"<<car1.GetVIN()<<endl;
cout<<car2.GetMarka()<<"\t"<<car2.Model<<"\t"<<car2.GetVIN()<<endl;
cout<<car3.GetMarka()<<"\t"<<car3.Model<<"\t"<<car3.GetVIN()<<endl;
	system("pause");
	return 0;
}
```

```
Opel    Corsa   VNM1234
Polonez Caro    MKK8976
Honda   Accord  KLMN1234
Press any key to continue . . .
```

```
//Konstruktor i destruktor

#include<iostream>
using namespace std;

class Example
{
public:
	Example()
	{cout<<"Konstruktor\n";}

	//Destruktor
	~Example()
	{cout<<"Destruktor\n";}
};
void main()
{
	Example Examples[5];
	for (int i=0; i<5; i++)
	{
		cout<<i<<endl;
		Examples[i].~Example();
	}
```

```
Example Przyklad;

	system("pause");
}
```

```
Konstruktor
Konstruktor
Konstruktor
Konstruktor
Konstruktor
0
Destruktor
1
Destruktor
2
Destruktor
3
Destruktor
4
Destruktor
Konstruktor
Press any key to continue . . .
```

```
// Tablice obiektow klasy

#include<iostream>
#include<string>
using namespace std;

class Krowa
{
	string Imie;
	int Wiek;

public:

	Krowa()
	{
		cout<<"Imie: \t";
		cin>>Imie;
		cout<<"Wiek : \t";
		cin>>Wiek;
	}
	void ID()
	{cout<<Imie<<"\t"<<Wiek<<endl;}
};
```

```
void main()
{
	Krowa Obora[4];
	for (int i=3; i>-1 ;i--)
	{Obora[i].ID();}
	system("pause");
}
```

```
Imie:   Krasula
Wiek :  10
Imie:   Laciata
Wiek :  5
Imie:   Mecka
Wiek :  6
Imie:   Ruda
Wiek :  3
Ruda    3
Mecka   6
Laciata 5
Krasula 10
Press any key to continue . . .
```

Funkcje zaprzyjaźnione klasy

Funkcje niebędące funkcjami składowymi klasy, mające dostęp do wszystkich jej składowych.

```
//Funkcje zaprzyjaźnione klasy

#include <iostream>
using namespace std;
const double pi=3.1415927;

class Walec
{
	double srednica;
	double wysokosc;
```

```
public:

        Walec(double d_walca=1.0, double h_walca=1.0):
          srednica(d_walca),wysokosc(h_walca)
        {cout<<"Konstruktor"<<endl;       }

        double Objetosc()
        {return pi*srednica*srednica*wysokosc/4;}

        friend double Pole(Walec walec)
                //Lista parametrów nie może być pusta
                {return pi*walec.srednica*walec.wysokosc
                +pi*walec.srednica*walec.srednica/2;}

};
```

```
void main()
{
	Walec walec_1;
	cout<<"Objetosc : "<<walec_1.Objetosc()<<endl;
	cout<<"Powierzchnia : "<<Pole(walec_1)<<endl;
	Walec walec_2(4.0,2.0);
	cout<<endl;
	cout<<"Objetosc : "<<walec_2.Objetosc()<<endl;
	cout<<"Powierzchnia : "<<Pole(walec_2)<<endl;

	system("pause");
}
```

```
Konstruktor
Objetosc : 0.785398
Powierzchnia : 4.71239
Konstruktor

Objetosc : 25.1327
Powierzchnia : 50.2655
Press any key to continue . . .
```

```
//Funkcje zaprzyjaźnione klasy
//Definicja funkcji poza cialem

#include <iostream>
using namespace std;

class Pudelko
	{
	public:

	Pudelko(double a=1.0, double b=1.0, double h=1.0)
	{
		dlugosc=a;
		szerokosc=b;
		wysokosc=h;
	}

	double Objetosc()
	{
		return dlugosc*szerokosc*wysokosc;
	}
```

```
friend double Powierzchnia(Pudelko box);

        private:
        double dlugosc;
        double szerokosc;
        double wysokosc;
};

double Powierzchnia(Pudelko box)
        {
        return 2.0*(box.dlugosc*box.szerokosc+
                box.dlugosc*box.wysokosc+
                box.szerokosc*box.wysokosc);
        }
```

```
int main()
{
	Pudelko pudlo_1;
	cout<<endl<<"Pojemnosc pudelka =
"<<pudlo_1.Objetosc()<<endl;
	cout<<endl<<"Powierzchnia pudelka =
"<<Powierzchnia(pudlo_1)<<endl;
	Pudelko pudlo_2(3,4,2);
	cout<<endl<<"Pojemnosc pudelka =
"<<pudlo_2.Objetosc()<<endl;
	cout<<endl<<"Powierzchnia pudelka =
"<<Powierzchnia(pudlo_2)<<endl;
	system("pause");
	return 0;
}
```

```
Pojemnosc pudelka = 1
Powierzchnia pudelka = 6
Pojemnosc pudelka = 24
Powierzchnia pudelka = 52
Press any key to continue . . .
```

Składowe statyczne klasy

Są to składowe, które przyjmują identyczne wartości we wszystkich obiektach klasy

```
// Składowe statyczne

#include<iostream>
using namespace std;

class Kura
{
public:
        static int liczba_kur;
        //static int liczba_kur=0;

        Kura()
        {
                cout<<"Liczba kur rosnie o 2\n";
                liczba_kur+=2;
        }
};
```

```
class Kogut
{
public:

        static int liczba_kogutow;
        //static int liczba_kogutow=5;
        const static int liczba_ptakow=2;

        Kogut()
        {
                cout<<"Liczba kogutow rosnie o 1\n";
                liczba_kogutow++;
                //liczba_ptakow++;
        }
};

//Inicjalizacja zmiennych statycznych
int Kura::liczba_kur=0;
int Kogut::liczba_kogutow=5;
```

```
void main()
{
	Kura kurnik_1[4];
	for (int i=0; i<4; i++)
	{cout<<i<<"\t"<<"Liczba kur =
"<<kurnik_1[i].liczba_kur<<endl;  }
	Kogut kurnik_2[3];
	for (int i=0; i<3; i++)
	{cout<<i<<"\t"<<"Liczba kogutow =
"<<kurnik_2[i].liczba_kogutow<<endl;
	cout<<i<<"\t"<<"Liczba ptakow =
"<<kurnik_2[i].liczba_ptakow<<endl;}
	system("pause");
}
```

```
Liczba kur rosnie o 2
Liczba kur rosnie o 2
Liczba kur rosnie o 2
Liczba kur rosnie o 2
0       Liczba kur = 8
1       Liczba kur = 8
2       Liczba kur = 8
3       Liczba kur = 8
Liczba kogutow rosnie o 1
Liczba kogutow rosnie o 1
Liczba kogutow rosnie o 1
0       Liczba kogutow = 8
0       Liczba ptakow = 2
1       Liczba kogutow = 8
1       Liczba ptakow = 2
2       Liczba kogutow = 8
2       Liczba ptakow = 2
Press any key to continue . . .
```

Funkcje składowe statyczne klasy

Są to funkcje, które działają tylko na składowych statycznych

```
// Funkcje statyczne

#include<iostream>
using namespace std;

class Kura
{
public:

	static int liczba_kur;
	int numer;

	Kura()
	{
		liczba_kur++;
	}
```

```
        Kura(int k):numer(k)
        {}
        static void Sale(int p)
        {
                cout<<"Liczba kur spadla o "<<p<<endl;
                liczba_kur-=p;
                //numer=0;
        }

};

int Kura::liczba_kur=0;

void main()
{
        Kura kurnik_1[10];
        Kura kurnik_2[4];
        for (int i=0; i<4; i++)
        {
                kurnik_2[i]=Kura(10+i);
        }
```

```
cout<<"liczba kur = "<<kurnik_1[0].liczba_kur<<endl;
	cout<<"liczba kur = "<<kurnik_2[0].liczba_kur<<endl;

	kurnik_2[0].Sale(6);
	cout<<"liczba kur = "<<kurnik_1[0].liczba_kur<<endl;
	cout<<"liczba kur = "<<kurnik_2[0].liczba_kur<<endl;

	system("pause");
}
```

```
liczba kur = 14
liczba kur = 14
Liczba kur spadla o 6
liczba kur = 8
liczba kur = 8
Press any key to continue . . .
```

```
// Wskazniki do obiektow klasy

#include<iostream>
#include<string>
using namespace std;

class Pracownik
{
public:
	string Imie;
	string Nazwisko;
	double Pensja;

	Pracownik(string imie,
		string nazwisko,
		double placa)
		:Imie(imie),Nazwisko(nazwisko), Pensja(placa)
	{}
	void Dane()
	{cout<<Imie<<"\t"<<Nazwisko<<"\t"<<Pensja<<endl;	}
};
```

```
void main()
{
	Pracownik prac_1("Jan","Kowalski",3500);
	cout<<prac_1.Imie<<endl;
	prac_1.Dane();
	cout<<sizeof(prac_1)<<"\t";
	cout<<sizeof(prac_1.Imie)<<"\t";
	cout<<sizeof(prac_1.Pensja)<<"\n";
	Pracownik* ptr;
	ptr=&prac_1;
	cout<<ptr<<endl;
	cout<<ptr->Imie<<"\t";
	cout<<ptr->Nazwisko<<"\t";
	cout<<ptr->Pensja<<endl;
	ptr->Dane();

	system("pause");
}
```

```
Jan
Jan     Kowalski        3500
72      32      8
005AF764
Jan     Kowalski        3500
Jan     Kowalski        3500
Press any key to continue . . .
```

```
// Tworzenie obiektow klasy
//za pomoca wskaznikow

#include<iostream>
#include<string>
using namespace std;

class Pracownik
{
public:
	string Imie;
	string Nazwisko;
	double Pensja;

	Pracownik(string imie,
		string nazwisko,
		double placa)
		:Imie(imie),Nazwisko(nazwisko), Pensja(placa)
	{}
	void Dane()
	{cout<<Imie<<"\t"<<Nazwisko<<"\t"<<Pensja<<endl;	}
};
```

```
void main()
{
	Pracownik* prac_1=new Pracownik("Jan","Kowalski",3500);
	//Pracownik* prac_2=new Pracownik;
	cout<<sizeof(prac_1)<<endl;
	cout<<prac_1->Imie<<endl;
	prac_1->Dane();
	Pracownik* ptr;
	ptr=prac_1;

	/*delete prac_1;
	w tym momencie program pada*/

	cout<<ptr<<endl;
	cout<<ptr->Imie<<"\t";
	cout<<ptr->Nazwisko<<"\t";
	cout<<ptr->Pensja<<endl;
	ptr->Dane();
	delete prac_1;


	system("pause");
}
```

```
4
Jan
Jan     Kowalski        3500
00CA4B28
Jan     Kowalski        3500
Jan     Kowalski        3500
Press any key to continue . . .
```

```
// Wskaźniki do klas

#include<iostream>
#include<string>
using namespace std;

string Rodzaj[4]=   {"TV      ", "wieza    ", "DVD      ",
"kino     "};
string Producent[4]={"Phillips", "Sony     ", "Sanyo    ",
"Samsung  "};

class Sprzet_AV
{
      string rodzaj;
      string firma;

public:
      int numer_fabr;

      void GetFirm();
      void GetSort();
```

```
Sprzet_AV(int a1, int a2, int a3)
      :rodzaj(Rodzaj[a1]),firma(Producent[a2]),numer_fabr(a3)
{}
      };

void Sprzet_AV::GetFirm()
{cout<<firma<<"\t";}

void Sprzet_AV::GetSort()
{cout<<rodzaj<<"\t";}

int main()
{
      Sprzet_AV* magazyn[4];
      cout<<"Rozmiar obiektu : "<<sizeof(magazyn[0])<<endl;
      cout<<"Adresy obiektow :\n";
      for (int i=0; i<4; i++)
      {cout<<&magazyn[i]<<"\t";}
      cout<<endl;
      Sprzet_AV* ptr;
      int wybor_firmy, wybor_sprzetu, numer,i;
```

```
for (i=0;i<4;i++)
       {
              cout<<"Wybor firmy :\n";
              cout<<"0 - Phillips 1 - Sony 2 - Sanyo 3 -
Samsung : ";
              cin>>wybor_firmy;
              cout<<"Wybor sprzetu :\n";
              cout<<"0 - TV 1 - wieza 2 - DVD 3 - kino : ";
              cin>>wybor_sprzetu;
              cout<<"Numer fabryczny : ";
              cin>>numer;
              magazyn[i]=new
Sprzet_AV(wybor_sprzetu,wybor_firmy,numer);
       }
       ptr=new Sprzet_AV(2,1,5678);
       ptr->GetFirm();
       ptr->GetSort();
       cout<<ptr->numer_fabr<<endl;
```

```
for (i=0;i<4;i++)
        {
                ptr=magazyn[i];
                ptr->GetFirm();
                ptr->GetSort();
                cout<<ptr->numer_fabr<<endl;
        }

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

```
Rozmiar obiektu : 4
Adresy obiektow :
0012FF48        0012FF4C        0012FF50        0012FF54

0 - Phillips 1 - Sony 2 - Sanyo 3 - Samsung : 0
Wybor sprzetu :
0 - TV 1 - wieza 2 - DVD 3 - kino : 0
Numer fabryczny : 1234
Wybor firmy :
0 - Phillips 1 - Sony 2 - Sanyo 3 - Samsung : 1
Wybor sprzetu :
0 - TV 1 - wieza 2 - DVD 3 - kino : 1
Numer fabryczny : 2345
Wybor firmy :
0 - Phillips 1 - Sony 2 - Sanyo 3 - Samsung : 3
Wybor sprzetu :
0 - TV 1 - wieza 2 - DVD 3 - kino : 3
Numer fabryczny : 3456
Wybor firmy :
0 - Phillips 1 - Sony 2 - Sanyo 3 - Samsung : 2
Wybor sprzetu :
0 - TV 1 - wieza 2 - DVD 3 - kino : 2
Numer fabryczny : 4567
Sony            DVD             5678
Phillips        TV              1234
Sony            wieza           2345
Samsung         kino            3456
Sanyo           DVD             4567
Press any key to continue . . .
```