

Wartości zwracane przez funkcję

1. Zadeklarowany typ
2. Wskaźnik
3. Referencja

Problemy przy zwracaniu wskaźnika

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

int* Dodaj_5(int liczba);

int main()
{
    int liczba = 3;
    int* wynik=new int;
    wynik=Dodaj_5(liczba);
    //cout <<"liczba + 5 = " << liczba+5 << endl;
    cout << endl<<"Dodaj_5(liczba) = " <<
    *wynik<<endl;

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

```
int* Dodaj_5(int liczba)
{
int wynik=liczba+5;
return &wynik;
}
```

```
Dodaj_5(liczba) = 8
Press any key to continue .
. .
```

liczba + 5 = 8

```
Dodaj_5(liczba) = 1693240008
Press any key to continue .
. .
```

Wniosek : Nigdy nie należy zwracać adresu zmiennej lokalnej

Warning C4172: returning address of local variable or temporary

```
//Wersja poprawna

#include <iostream>
using namespace std;

int* Dodaj_5(int liczba);

int main()
{
    int liczba = 3;
    int* wynik=new int(0);
    wynik=Dodaj_5(liczba);
    cout <<"liczba + 5 = " << liczba+5 << endl;
    cout << endl<<"Dodaj_5(liczba) = " <<
*wynik<<endl;

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

```
int* Dodaj_5(int liczba)
{
int* wynik=new int(0);
*wynik=liczba+5;
return wynik;
}
```

liczba + 5 = 8

Dodaj_5(liczba) = 8

Press any key to continue . . .

```
//Referencja

#include <iostream>
using namespace std;

int& Dodaj_5(int liczba);

int main()
{
    int liczba = 3;
    int &wynik=liczba;
    wynik=Dodaj_5(liczba);
    cout <<"liczba + 5 = " << liczba+5 << endl;
    cout << endl<<"Dodaj_5(liczba) = " <<
wynik<<endl;

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

```
int& Dodaj_5(int liczba)
{
int &wynik=liczba;
wynik=liczba+5;
return wynik;
}
```

liczba + 5 = 13

Dodaj_5(liczba) = 8

Press any key to continue . . .

```
//Referencja

#include <iostream>
using namespace std;

int& Dodaj_5(int liczba);

int main()
{
    int liczba = 3,liczba_1=9;
    int &wynik=liczba_1;
    wynik=Dodaj_5(liczba);
    cout <<"liczba + 5 = " << liczba+5 << endl;
    cout << endl<<"Dodaj_5(liczba) = " << wynik<<endl;

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



```
int& Dodaj_5(int liczba)
{
int &wynik=liczba;
wynik=liczba+5;
return wynik;
}
```

liczba + 5 = 8

Dodaj_5(liczba) = 8

Press any key to continue . . .

```
//modyfikator const

#include <iostream>
using namespace std;

int Dodaj_5(const int liczba);

int main()
{
    const int liczba = 3;
    cout << endl << "Adres lokalny = " << &liczba <<
endl;
    cout << endl << "Dodaj_5(liczba) = " <<
Dodaj_5(liczba);
    cout << endl << "liczba = " << liczba << endl;
    //liczba=Dodaj_5(liczba);

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

```
int Dodaj_5(const int liczba)
{
    cout << endl << "Adres = " << &liczba << endl;
    int wynik=liczba+5;
    return wynik;
}
```

Adres lokalny = 008DFD88

Adres = 008DFCB4

Dodaj_5(liczba) = 8

liczba = 3

Press any key to continue . . .

Przeciążenie funkcji w C++

Mechanizm pozwalający na użycie tej samej funkcji do zdefiniowania kilku funkcji o takich samych listach parametrów.

Przykład: dodawanie elementów 4 funkcje do różnych typów danych

```
// Przeciazenie funkcji

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

int Dodaj(int a,int b);
double Dodaj(double a,double b);
char Dodaj(char a,char b);
string Dodaj(string a,string b);

void main()
{
    cout<<"Int\t"<<Dodaj (2,3) <<endl;
    cout<<"Double\t"<<Dodaj (2.5,3.6) <<endl;
    cout<<"Char\t"<<Dodaj ('A','B') <<endl;
    cout<<"String\t"<<Dodaj ("abcd","efgh") <<endl;
    system("pause");
    return;
}
```

```
int Dodaj(int a,int b)
{return a+b;}
double Dodaj(double a,double b)
{return a+b;}
char Dodaj(char a,char b)
{return a+b;}
string Dodaj(string a,string b)
{return a+b;}
```

```
Int      5
Double   6.1
Char     â
String   abcdefgh
Press any key to continue . . .
```

Szablony funkcji w C++

Mechanizm pozwalający na napisanie przepisu, na podstawie którego kompilator będzie tworzył funkcje stosując różne typy parametrów.

```
template<class T> T Dodaj(T x, T len)
template<typename T> T Dodaj(T x, T len)
{
ciało funkcji
return ...;
}
```

```
// Szablony funkcji

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

template<typename T> T Dodaj (T a, T b)
{
    T wynik=a+b;
    return wynik;
}

void main()
{
    cout<<"Int\t"<<Dodaj (2, 3) <<endl;
    cout<<"Double\t"<<Dodaj (2.5, 3.6) <<endl;
    char a1='A', b1='B';
    cout<<"Char\t"<<Dodaj (a1, b1) <<endl;
    //cout<<"Char\t"<<Dodaj ('A', 'B') <<endl;
}
```



```
string a2="abcd",b2="efgh";
    cout<<"String\t"<<Dodaj (a2,b2)<<endl;
    //cout<<"String\t"<<Dodaj ("abcd", "efgh")<<endl;
    system("pause");
    return;
}
```

```
Int      5
Double   6.1
Char     â
String   abcdefgh
Press any key to continue . . .
```

```

// Szablony funkcji

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

template<typename T1, typename T2> int Dodaj (T1 a, T2 b)
{
    int wynik=a+b;
    return wynik;
}

void main()
{
    cout<<"Int\t"<<Dodaj (2, 3.5) <<endl;

    system("pause");
    return;
}

```

Int 5
Aby kontynuowaĆ, naciŚnij
dowolny klawisz . . .

```
//szablon max elementu tablicy

#include<iostream>
using namespace std;

// Szablon funkcji max
template<typename T> T max(T x[], int rozmiar)
{
    T max = x[0];
    for (int i = 1; i < rozmiar; i++)
        if (max < x[i])
            max = x[i];
    return max;
}

int main()
{
    int Tablica_int[] = {3, 21, 34, 15};
    long Tablica_long[]={23, 245, 123, 1, 234, 2345};
```

```

double Tablica_double[]={23.0, 1.4, 2.5, 345.0, 12.3,
2.51};
    char Tablica_char[]{"abdgrcf"};

    int rozmiar_int=sizeof Tablica_int/sizeof
Tablica_int[0];
    int rozmiar_long=sizeof Tablica_long/sizeof
Tablica_long[2];
    int rozmiar_double=sizeof Tablica_double/sizeof
Tablica_double[0];
    int rozmiar_char=sizeof Tablica_char/sizeof
Tablica_char[0];

    cout<< max(Tablica_int, rozmiar_int)<<endl;
    cout<< max(Tablica_long, rozmiar_long)<<endl;
    cout<< max(Tablica_double, rozmiar_double)<<endl;
    cout<< max(Tablica_char, rozmiar_char)<<endl;
    system("pause");
    return 0;
}

```

34

2345

345

r

Press any key to continue .

. .

Makro

Preprocesor – jednostka kompilatora przetwarzająca kod przed jego rzeczywistą kompilacją.

`#define` – do tworzenia funkcji makro – preprocesor podstawia ciąg tekstowy w miejscu, gdzie wystąpi argument `#define`

```
#define MAX(x, y) ((x) > (y) ? (x) : (y))
```

```
#define MIN(x, y) ((x) < (y) ? (x) : (y))
```

```

// Rozwijanie makr

#include<iostream>
using namespace std;

#define CUBE(a) ((a)*(a)*(a))
#define THREE(a) a*a*a

int main()
{
    long x = 5;
    long y = CUBE(x);
    long z = THREE(x);

    cout <<"y: " <<y<<endl;
    cout <<"z: " <<z<<endl;

    long a = 5, b = 7;

    y = CUBE(a+b);
    z = THREE(a+b);

    cout <<"y: " <<y<<endl;
    cout <<"z: " <<z<<endl;
    system("pause");
    return 0;
}

```

y: 125
z: 125
y: 1728
z: 82
Aby kontynuować, naciśnij dowolny
klawisz . . .

Problemy z makro:

1. Makro musi się zmieścić w 1 linii
2. Jest rozwijane w każdym miejscu, gdzie jest wywoływane
3. Kod makra nie pojawia się w kodzie źródłowym – trudne debugowanie
4. Makra nie są bezpieczne ze względu na typy – bezpieczniej jest używać funkcji

Struktura unia klasa

Są to definicje nowego typu danych

- **Unia – kontener do przechowywania danych**
- **Struktura – zmienne składowe i funkcje składowe – dostęp public**
- **Klasa - zmienne składowe i funkcje składowe – dostęp private**

```
// Unia

#include<iostream>
using namespace std;

union Zmienna
{
    char oznaczenie;
    double wartosc;
};

void main()
{
    Zmienna zmienna_1={'a'};
    cout<<"zmienna_1\n";
    cout<<"Oznaczenie = "<<zmienna_1.oznaczenie<<endl;
    cout<<"Wartosc = "<<zmienna_1.wartosc<<endl;
    Zmienna zmienna_2={6.5};
    cout<<"zmienna_2\n";
    cout<<"Oznaczenie = "<<zmienna_2.oznaczenie<<endl;
    cout<<"Wartosc = "<<zmienna_2.wartosc<<endl;
    zmienna_1.wartosc=3.5;
    zmienna_2.oznaczenie='b';
}
```

```

cout<<"zmienna_1\n";
    cout<<"Oznaczenie = "<<zmienna_1.oznaczenie<<endl;
    cout<<"Wartosc = "<<zmienna_1.wartosc<<endl;
    cout<<"zmienna_2\n";
    cout<<"Oznaczenie = "<<zmienna_2.oznaczenie<<endl;
    cout<<"Wartosc = "<<zmienna_2.wartosc<<endl;

    system ("pause");
    return;
}

```

```

zmienna_1
Oznaczenie = a
Wartosc = 4.79244e-322
zmienna_2
Oznaczenie = ♠
Wartosc = 2.96439e-323
zmienna_1
Oznaczenie =
Wartosc = 3.5
zmienna_2
Oznaczenie = b
Wartosc = 4.84184e-322
Press any key to continue . . . 27

```

Struktura

C - definicja nowego typu danych

```
struct KSIAZKA
{
char Tytul[80];
char Autor [80];
char Wydawca [80];
int Rok_wydania;
};
```

Ograniczenia:

1. Nie ma możliwości umieszczenia zmiennej o tej samej nazwie (może być wskaźnik)
2. Ze strukturami nie działają wbudowane operatory

```
KSIAZKA Powiesc; //Deklaracja zmiennej Powiesc typu
//KSIAZKA
```

```
KSIAZKA Podrecznik =  
{  
"C++ dla kazdego";  
"J. Liberty";  
"Helion";  
2009;  
};
```

```
Powiesc.Rok_wydania=2011;// Dostęp do zmiennej
```

```
// Struktura

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

struct Ksiazka
{
    string Autor;
    string Tytul;
    double rok_wydania;

    void Wyszwietl()
    {
        cout<<"Autor : \t"<<Autor<<endl;
        cout<<"Tytul : \t"<<Tytul<<endl;
        cout<<"Rok : \t"<<rok_wydania<<endl;
    }
};
```

```

void main()
{
    Ksiazka ksiazka_1={"H. Sienkiewicz","Potop",2010};
    cout<<"Rozmiar obiektu = "<<sizeof(ksiazka_1)<<endl;
    cout<<"ksiazka_1\n";
    cout<<ksiazka_1.Autor<<endl;
    cout<<ksiazka_1.Tytul<<endl;
    cout<<ksiazka_1.rok_wydania<<endl;
    cout<<"Zastosowanie funkcji \n";
    ksiazka_1.Wyświetl();
    Ksiazka ksiazka_2;
    ksiazka_2.Autor="W. Reymont";
    ksiazka_2.Tytul="Chłopi";
    ksiazka_2.rok_wydania=2012;
    ksiazka_2.Wyświetl();
    cout<<"Rozmiar obiektu = "<<sizeof(ksiazka_2)<<endl;

    system ("pause");
    return;
}

```

Rozmiar obiektu = 72
ksiazka_1
H. Sienkiewicz
Potop
2010
Zastosowanie funkcji
Autor : H. Sienkiewicz
Tytul : Potop
Rok : 2010
Autor : W. Reymont
Tytul : Chłopi
Rok : 2012
Rozmiar obiektu = 72
Press any key to continue . . .


```
// Klasa

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

class Ksiazka
{
    string Autor;
    string Tytul;
    double rok_wydania;

    void Wyszwietl()
    {
        cout<<"Autor : \t"<<Autor<<endl;
        cout<<"Tytul : \t"<<Tytul<<endl;
        cout<<"Rok : \t"<<rok_wydania<<endl;
    }
};
```

```
void main()
{
    Ksiazka ksiazka_1={"H. Sienkiewicz", "Potop", 2010};
    cout<<"Rozmiar obiektu = "<<sizeof(ksiazka_1)<<endl;
    cout<<"ksiazka_1\n";
    cout<<ksiazka_1.Autor<<endl;
    cout<<ksiazka_1.Tytul<<endl;
    cout<<ksiazka_1.rok_wydania<<endl;
    cout<<"Zastosowanie funkcji \n";
    ksiazka_1.Wyświetl();
    Ksiazka ksiazka_2;
    ksiazka_2.Autor="W. Reymont";
    ksiazka_2.Tytul="Chłopi";
    ksiazka_2.rok_wydania=2012;
    ksiazka_2.Wyświetl();
    cout<<"Rozmiar obiektu = "<<sizeof(ksiazka_2)<<endl;

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

```
1>klasa.cpp
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(23) : error C2552: 'ksiazka_1' : non-aggregates
cannot be initialized with initializer list
1>          'Ksiazka' : Types with private or protected data members are not
aggregate
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(26) : error C2248: 'Ksiazka::Autor' : cannot access
private member declared in class 'Ksiazka'
1>          e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(9) : see declaration of 'Ksiazka::Autor'
1>          e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(8) : see declaration of 'Ksiazka'
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(27) : error C2248: 'Ksiazka::Tytul' : cannot access
private member declared in class 'Ksiazka'
1>          e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(10) : see declaration of 'Ksiazka::Tytul'
1>          e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(8) : see declaration of 'Ksiazka'
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(28) : error C2248: 'Ksiazka::rok_wydania' : cannot
access private member declared in class 'Ksiazka'
1>          e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(11) : see declaration of 'Ksiazka::rok_wydania'
1>          e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(8) : see declaration of 'Ksiazka'
```

```
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(30) : error C2248: 'Ksiazka::Wyswietl' : cannot access
private member declared in class 'Ksiazka'
1>         e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(13) : see declaration of 'Ksiazka::Wyswietl'
1>         e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(8) : see declaration of 'Ksiazka'
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(32) : error C2248: 'Ksiazka::Autor' : cannot access private
member declared in class 'Ksiazka'
1>         e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(9) : see declaration of 'Ksiazka::Autor'
1>         e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(8) : see declaration of 'Ksiazka'
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(33) : error C2248: 'Ksiazka::Tytul' : cannot access private
member declared in class 'Ksiazka'
1>         e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(10) : see declaration of 'Ksiazka::Tytul'
1>         e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(8) : see declaration of 'Ksiazka'
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(34) : error C2248: 'Ksiazka::rok_wydania' : cannot access
private member declared in class 'Ksiazka'
1>         e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(11) : see declaration of 'Ksiazka::rok_wydania'
1>         e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(8) : see declaration of 'Ksiazka'
1>e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(35) : error C2248: 'Ksiazka::Wyswietl' : cannot access
private member declared in class 'Ksiazka'
1>         e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(13) : see declaration of 'Ksiazka::Wyswietl'
```

```
1> e:\praca\dydaktyka\programowanie obiektowe\visual project\wykład
2016\w_03_03\w_03_03\klasa.cpp(8) : see declaration of 'Ksiazka'
1>Build log was saved at "file:///e:\Praca\Dydaktyka\Programowanie obiektowe\Visual
Project\Wykład 2016\W_03_03\W_03_03\Debug\BuildLog.htm"
1>W_03_03 - 9 error(s), 0 warning(s)
===== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped =====
```

```
// Klasa

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

class Ksiazka
{
    string Autor;
    string Tytul;
    double rok_wydania;

public:

    void Wyszwietl()
    {
        cout<<"\nAutor : \t"<<Autor<<endl;
        cout<<"Tytul : \t"<<Tytul<<endl;
        cout<<"Rok : \t"<<rok_wydania<<endl;
    }
}
```

```

void Wpisz()
{
    cout<<"Wpisywanie danych :\nAutor : \t";
    cin>>Autor;
    cout<<"Tytul : \t";
    cin>>Tytul;
    cout<<"Rok : \t";
    cin>>rok_wydania;
}

};

void main()
{
    Ksiazka ksiazka_1;
    cout<<"Rozmiar obiektu = "<<sizeof(ksiazka_1)<<endl;
    cout<<"ksiazka_1\n";
    ksiazka_1.Wpisz();
    ksiazka_1.Wyswietl();
}

```

```
Ksiazka ksiazka_2;  
    //ksiazka_2.Autor="W. Reymont";  
    //ksiazka_2.Tytul="Chlopi";  
    //ksiazka_2.rok_wydania=2012;  
    ksiazka_2.Wyświetl();  
    cout<<"Rozmiar obiektu = "<<sizeof(ksiazka_2)<<endl;  
  
    system ("pause");  
    return;  
}
```


Rozmiar obiektu = 72

ksiazka_1

Wpisywanie danych :

Autor : Sienkiewicz

Tytul : Potop

Rok : 2009

Autor : Sienkiewicz

Tytul : Potop

Rok : 2009

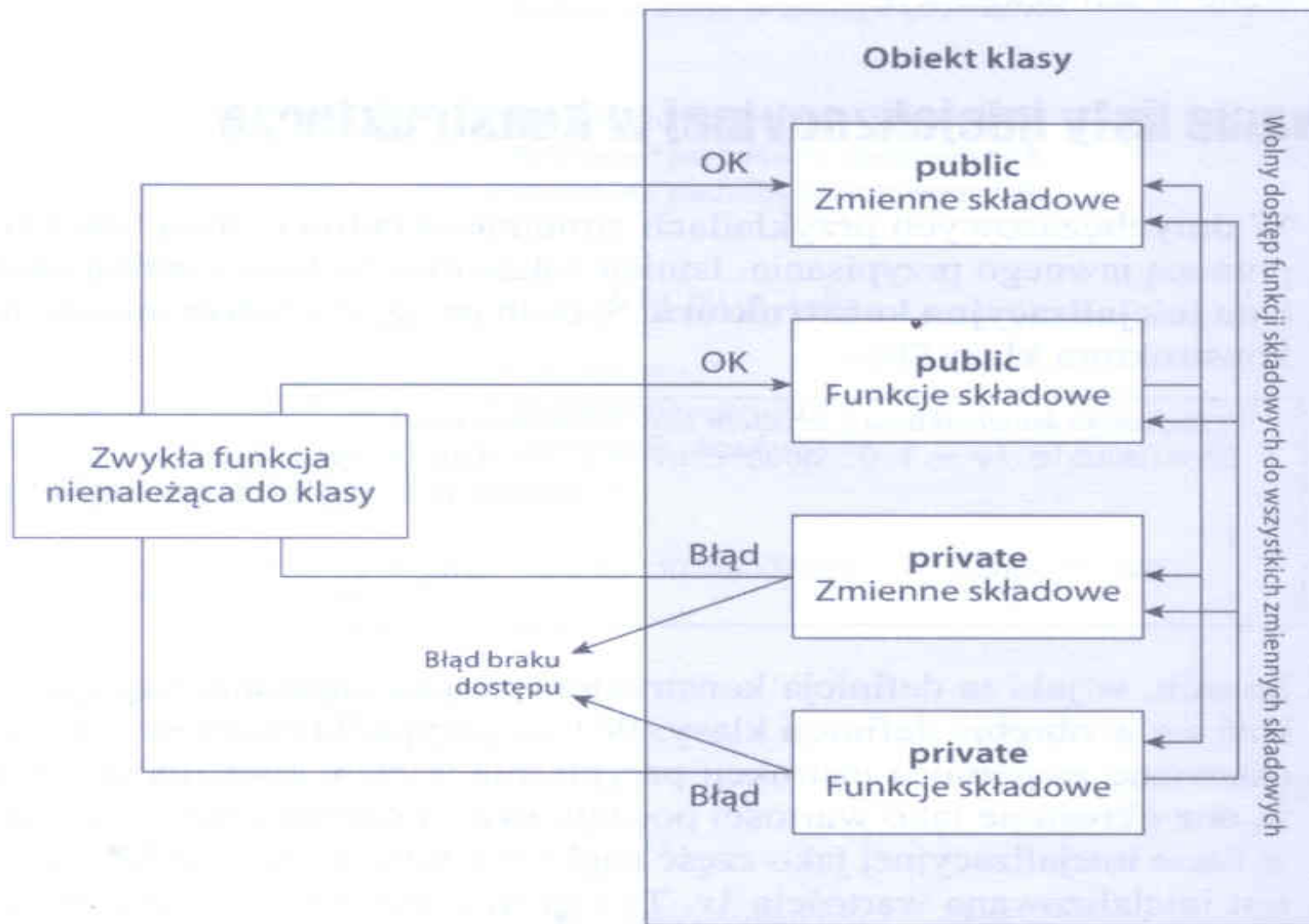
Autor :

Tytul :

Rok : -9.25596e+061

Rozmiar obiektu = 72

Press any key to continue . . .



```
// Klasa umiejscowienie funkcji

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

class Ksiazka
{
    string Autor;
    string Tytul;
    double rok_wydania;

public:
    void Wyszwietl();
    void Wpisz();
};

void Ksiazka::Wyszwietl()
{
    cout<<"\nAutor : \t"<<Autor<<endl;
    cout<<"Tytul : \t"<<Tytul<<endl;
    cout<<"Rok : \t"<<rok_wydania<<endl;
}
```

```

void Ksiazka::Wpisz()
{
    cout<<"Wpisywanie danych :\nAutor : \t";
    cin>>Autor;
    cout<<"Tytul : \t";
    cin>>Tytul;
    cout<<"Rok : \t";
    cin>>rok_wydania;
}

void main()
{
    Ksiazka ksiazka_1;
    cout<<"Rozmiar obiektu = "<<sizeof(ksiazka_1)<<endl;
    cout<<"ksiazka_1\n";
    ksiazka_1.Wpisz();
    ksiazka_1.Wyswietl();
    system ("pause");
    return;
}

```

Rozmiar obiektu = 72

ksiazka_1

Wpisywanie danych :

Autor : Reymont

Tytul : Chłopi

Rok : 1999

Autor : Reymont

Tytul : Chłopi

Rok : 1999

Press any key to continue . . .