

```
// W15_1.cpp : Defines the entry point for the console application.  
//          RTTI: type_id do szablonow klas
```

```
#include "stdafx.h"  
#include <iostream>  
#include <cmath>  
using namespace std;
```

```
template <class T> class B  
{  
    T *dat;  
public:  
    B(size_t dim);  
    ~B() { delete [] dat; }  
    virtual void f() { cout << "Jestem B\n"; }  
};
```

```
template <class T>  
B<T>::B(size_t dim)  
{  
    try  
    {  
        dat = new T [dim];  
    }  
    catch(bad_alloc aa)  
    {  
        cout << "mem_alloc_err\n";  
        system("pause");  
        exit(1);  
    }  
}
```

```
template <class T> class D1 : public B<T>  
{  
public:  
    D1(size_t ii):B(ii) {}  
    void f() { cout << "Jestem D1\n"; }  
};
```

```
void main()  
{  
    B<double> b(5), *pb;
```

```

D1<double> d1(5), *pd1;

pb = &d1;
if(typeid(*pb) == typeid(D1<double>))
    cout << "pb wskazuje na obiekt typu " << typeid(*pb).name() << endl;

if(pd1 = dynamic_cast<D1<double> *> (pb))
{
    cout << "pb wskazuje na obiekt typu class D1<double> \n";
    pd1->f();
}
else
{
    cout << "pb nie wskazuje na obiekt typu class D1<double> \n";
}

B<int> ib(3), *p_ib;
p_ib = dynamic_cast<B<int> *> (&ib);

if(pb = dynamic_cast<B<double> *> (p_ib)) //pb ma typ B<double> *,
                                           //a p_ib - B<int> *
{
    cout << "pb wskazuje na obiekt typu class B<double> \n";
    pb->f();
}
else
{
    //cout << "pb nie jest typem class B<double>, wyrazenie jest wskaznikiem typu B<int>\n";
    cout << "pb nie jest typem class B<double>, wyrazenie jest wskaznikiem typu ";
    cout << typeid(*p_ib).name() << "\n";
}

system("pause");
}

```