

```

// W15_2.cpp : Defines the entry point for the console application.
//
//      RTTI - dynamic_cast<> ()
//      Rozpoznanie typu obiektu klasy pochodnej, do ktorego wskazuje wskaznik klasy bazowej
//
//      B      B
//      |      |
//      |      |
//      D1     D2
//
//      B, D1, D2 - klasy polimorficzne
//      tylko klasa D1 zawiera składowa int i, własne to nadaje możliwość wyrażnie pokazać
//      konieczność poprawnego rzutowania typów

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

class B
{
public:
    virtual void f() { cout << "f from B\n";};
};

class D1 : public B
{
public:
    int i;    //!!!
    void f() {cout << "f from D1: i = " << i << "\n";};
};

class D2 : public B
{
public:
    void f() {cout << "f from D2\n";};
};

void main()
{
    int cel, i;
    B b, *pb;
    D1 d1, *pd1;
    D2 d2, *pd2;

```

```

for(i=0; i<=10; i++)
{
    cout << i << endl;

    //ustawiamy w sposob losowy wskaznik typu B
    //do obiektow typow B, D1, D2
    cel = rand()%3;
    if(cel == 0)
        pb = &b;
    else if(cel == 1)
        pb = &d1;
    else
        pb = &d2;

    //dynamic_cast<Typ *> (pb) zakonczy sie powodzeniem wtedy i tylko wtedy,
    //jesli pb wskazuje na obiekt typu Typ - wyznaczamy, do obiektu ktorego typu
    //wskazuje pb
    if(pd1 = dynamic_cast<D1 *> (pb))
    {
        cout << "pb wskazuje na obiekt typu D1\n";
        pd1->i = 1;
        //pd1->f();
        pb->f();
    }
    else if(pd2 = dynamic_cast<D2 *> (pb))
    {
        cout << "pb wskazuje na obiekt typu D2\n";
        //pd2->f();
        pb->f();
    }

    else if(pb = dynamic_cast<B *> (pb))
    {
        cout << "pb wskazuje na obiekt typu B\n";
        pb->f();
    }

    //a to jest to samo, na podstawie typeid
    if(typeid(*pb) == typeid(D1))
    {
        cout << "pb wskazuje na obiekt typu D1\n";
    }
}

```

```

        pd1 = (D1 *)pb;
        pd1->i = 1;
        pb->f();
    }
    else if(typeid(*pb) == typeid(D2))
    {
        cout << "pb wskazuje na obiekt typu D2\n";
        pd2 = (D2 *)pb;
        pd2->f();
    }
    else if(typeid(*pb) == typeid(B))
    {
        cout << "pb wskazuje na obiekt typu B\n";
        pd2 = (D2 *)pb;
        pd2->f();
        //pb->f();
    }
/*
    //a to jest kod niepoprawny - pd1 musi wskazywac tylko na obiekt d1.
    //jesli pb wskazuje na inny typ - blad
    //usun komentarze i sprobuj na wersji debug i release
    pd1 = (D1 *)pb;
    //pb->i =1;    //to jest blad - pb nie widzi skladowej i,
    //zdefiniowanej w klasie pochodnej
    pd1->i = 1;
    //pd1->f();
    pb->f();
*/
    cout << "-----\n\n";
}

system("pause");
}

```