

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x = 61 ) { m_X = new int (x); }
        A ( const A & src ) { m_X = new int ( *src . m_X ); }
        virtual ~A ( void ) { delete m_X; }
        virtual void print ( void ) const { cout << *m_X; }
    private:
        int *m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) : A ( x ) { m_Y = new int (y); }
        B ( const B & src ) : A ( src ) { m_Y = new int (*src.m_Y); }
        virtual ~B ( void ) { delete m_Y; }
        virtual void print ( void ) const { A::print (); cout << *m_Y; }
    private:
        int *m_Y;
};

void foo ( A * val )
{
    val -> print ( );
    delete val;
}

int main ( void )
{
    foo ( new B ( 81, 51 ) );
    return 0;
}
```



Program funguje, ale neuvolní alokovanou paměť



Program nepůjde zkompileovat



Program funguje zcela správně, zobrazí: 8151



Program půjde zkompileovat, ale po spuštění může spadnout/spadne

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x ) { m_X = x; }
        virtual ~A ( void ) { }
        virtual void print ( void ) const { cout << m_X; }
    private:
        int m_X;
};

class B : public A
{

```

```

public:
    B ( int x, int y ) : A ( x ) { m_Y = y; }
    virtual void print ( void ) const { A::print (); cout << m_Y; }
private:
    int m_Y;
};

void foo ( A & val )
{
    val . print ( );
}

int main ( void )
{
    B test ( 39, 44 );


    foo ( test );
    return 0;
}

```

 **Program funguje zcela správně, zobrazí: 3944**

 Program nepůjde zkompileovat

 Program půjde zkompileovat, ale po spuštění může spadnout/spadne

 Program funguje, ale neuvolní alokovanou paměť

Co zobrazí následující kód?

```

class A
{
public:
    A ( int x = 53 ) { m_X = new int (x); }
    A ( const A & src ) { m_X = new int ( *src . m_X ); }
    ~A ( void ) { delete m_X; }
    virtual void print ( void ) const { cout << *m_X; }
private:
    int *m_X;
};

class B : public A
{
public:
    B ( int x, int y ) : A ( x ) { m_Y = new int (y); }
    B ( const B & src ) : A ( src ) { m_Y = new int (*src.m_Y); }
    ~B ( void ) { delete m_Y; }
    virtual void print ( void ) const { A::print (); cout << *m_Y; }
private:
    int *m_Y;
};

void foo ( A * val )
{
    val -> print ( );
    delete val;
}

```

```
int main ( void )
{
    foo ( new B ( 4, 4 ) );
    return 0;
}
```

- ☐ Program nepůjde zkompileovat
 - ☐ Program funguje zcela správně, zobrazí:
 - ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☒ Program funguje, ale neuvolní alokovanou paměť
-

Co zobrazí následující kód?

```
class A
{
public:
    A ( int x ) { m_X = x; }
    virtual ~A ( void ) { }
    friend ostream & operator << ( ostream & os, const A & x )
    { return os << x . m_X; }
private:
    int m_X;
};

class B : public A
{
public:
    B ( int x, int y ) : A ( x ) { m_Y = y; }
    friend ostream & operator << ( ostream & os, const B & x )
    { os << (const A &) x; return os << x . m_Y; }
private:
    int m_Y;
};

void foo ( A * val )
{
    cout << *val;
}

int main ( void )
{
    B test ( 66, 45 );

    foo ( & test );
    return 0;
}
```

- ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☐ Program funguje, ale neuvolní alokovanou paměť
 - ☐ Program nepůjde zkompileovat
 - ☒ Program funguje zcela správně, zobrazí: 66
-

Co zobrazí následující kód?

```
class A
{
    public:
        A ( void ) { cout << *this; }
        virtual ~A ( void ) { }
        virtual void print ( ostream & os ) const { os << "A:"; }
        friend ostream & operator << ( ostream & os, const A & x )
        { x . print ( os ); return os; }
};

class B : public A
{
    public:
        B ( int x ) { cout << *this; m_X = x; }
        virtual void print ( ostream & os ) const { os << "B:" << m_X; }
    private:
        int m_X;
};

int main ( void )
{
    B test ( 36 );
    return 0;
}
```

- ☐ Program půjde zkompilevat, ale po spuštění může spadnout/spadne
 - ☐ Program funguje zcela správně, zobrazí:
 - ☐ Program funguje, ale neuvolní alokovanou paměť
 - ☐ Program nepůjde zkompilevat
 - ☒ Program nespadne, ale výstup není definovaný
-

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x = 55 ) { m_X = new int (x); }
        A ( const A & src ) { m_X = new int ( *src . m_X ); }
        virtual ~A ( void ) { delete m_X; }
        virtual void print ( void ) const { cout << *m_X; }
    private:
        int *m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) : A ( x ) { m_Y = new int (y); }
        B ( const B & src ) { m_Y = new int (*src.m_Y); }
        virtual ~B ( void ) { delete m_Y; }
        virtual void print ( void ) const { A::print (); cout << *m_Y; }
    private:
        int *m_Y;
}
```

```

};

void foo ( const B & val )
{
    val . print ( );
}

int main ( void )
{
    B test ( 69, 34 );

    foo ( test );
    return 0;
}

```



Program nepůjde zkompilevat



Program funguje, ale neuvolní alokovanou paměť



Program funguje zcela správně, zobrazí: 6934



Program půjde zkompilevat, ale po spuštění může spadnout/spadne

Co zobrazí následující kód?

```

class A
{
public:
    A ( int x ) { m_X = x; }
    void print ( void ) const { cout << m_X; }
private:
    int m_X;
};

class B : public A
{
public:
    B ( int x, int y ) : A ( x ) { m_Y = y; }
    void print ( void ) const { A::print (); cout << m_Y; }
private:
    int m_Y;
};

void foo ( B val )
{
    val . print ( );
}

int main ( void )
{
    B test ( 30, 19 );

    foo ( test );
    return 0;
}

```

- ☒ Program funguje, ale neuvolní alokovanou paměť
 - ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☐ Program nepůjde zkompileovat
 - ☒ Program funguje zcela správně, zobrazí: 3019
-

Co zobrazí následující kód?

```
class A
{
    public:
        A ( void ) { cout <<"A: "; }
        virtual ~A ( void ) { }
        void print ( ostream & os ) const = 0;
        friend ostream & operator << ( ostream & os, const A & x )
        { x . print ( os ); return os; }
};

class B: public A
{
    public:
        B ( int x ) : m_X ( x ) { cout << *this; }
        void print ( ostream & os ) const { os << "B:" << m_X; }
    private:
        int m_X;
};

int main ( void )
{
    B test ( 60 );
    return 0;
}
```

- ☒ Program nepůjde zkompileovat
 - ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☐ Program funguje zcela správně, zobrazí:
 - ☒ Program nespadne, ale výstup není definovaný
 - ☐ Program funguje, ale neuvolní alokovanou paměť
-

```
class A
{
    public:
        A ( int x ) { m_X = x; }
        virtual ~A ( void ) { }
        virtual void print ( ostream & os ) const { os << m_X; }
        friend ostream & operator << ( ostream & os, const A & x )
        { x . print ( os ); return os; }
    private:
        int m_X;
}
```

```

};

class B : public A
{
public:
    B ( int x, int y ) : A ( x ) { m_Y = y; }
    virtual void print ( ostream & os ) const { A::print ( os ); os << m_Y; }

    friend ostream & operator << ( ostream & os, const B & x )
    { os << "B:"; x . print ( os ); return os; }
private:
    int m_Y;
};

void foo ( B * val )
{
    cout << * val;
}

int main ( void )
{
    B test ( 14, 22 );

    foo ( & test );
    return 0;
}

```

- ☒ Program funguje zcela správně, zobrazí: B:1422
 - ☐ Program půjde zkompilevat, ale po spuštění může spadnout/spadne
 - ☐ Program nepůjde zkompilevat
 - ☐ Program funguje, ale neuvolní alokovanou paměť
-

Co zobrazí následující kód?

```

class A
{
public:
    A ( int x ) { m_X = x; }
    virtual ~A ( void ) { }
    virtual void print ( void ) const { cout << m_X; }
private:
    int m_X;
};

class B : A
{
public:
    B ( int x, int y ) : A ( x ) { m_Y = y; }
    virtual void print ( void ) const { A::print (); cout << m_Y; }
private:
    int m_Y;
};

void foo ( A val )

```

```

{
    val . print ( );
}

int main ( void )
{
    B test ( 9, 40 );

    foo ( test );
    return 0;
}

```

- ☒ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
- ☒ **Program nepůjde zkompileovat**
- ☐ Program funguje, ale neuvolní alokovanou paměť
- ☐ Program funguje zcela správně, zobrazí:

Co zobrazí následující kód?

```

class A
{
public:
    A ( int x ) { m_X = x; }
    virtual ~A ( void ) { }
    virtual void print ( ostream & os ) const { os << m_X; }
    friend ostream & operator << ( ostream & os, const A & x )
    { x . print ( os ); return os; }
private:
    int m_X;
};

class B : public A
{
public:
    B ( int x, int y ) : A ( x ) { m_Y = y; }
    virtual void print ( ostream & os ) const { A::print ( os ); os << m_Y; }
    friend ostream & operator << ( ostream & os, const B & x )
    { os << "B:"; x . print ( os ); return os; }
private:
    int m_Y;
};

void foo ( A * val )
{
    cout << *val;
}

int main ( void )
{
    B test ( 41, 86 );

    foo ( &test );
    return 0;
}

```


}



Program funguje, ale neuvolní alokovanou paměť



Program funguje zcela správně, zobrazí: 4186



Program půjde zkompilevat, ale po spuštění může spadnout/spadne



Program nepůjde zkompilevat

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x ) { m_X = x; }
        void print ( void ) const { cout << m_X; }
    private:
        int m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) { m_Y = y; }
        void print ( void ) const { A::print (); cout << m_Y; }
    private:
        int m_Y;
};

void foo ( B val )
{
    val . print ( );
}

int main ( void )
{
    B test ( 59, 76 );

    foo ( test );
    return 0;
}
```



Program půjde zkompilevat, ale po spuštění může spadnout/spadne



Program nepůjde zkompilevat



Program funguje zcela správně, zobrazí:



Program funguje, ale neuvolní alokovanou paměť

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x ) { m_X = x; }
        virtual ~A ( void ) { }
        virtual void print ( void ) const { cout << m_X; }
    private:
        int m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) : A ( x ) { m_Y = y; }
        virtual void print ( void ) const { A::print (); cout << m_Y; }
    private:
        int m_Y;
};

void foo ( A * val )
{
    val -> print ( );
}

int main ( void )
{
    B test ( 92, 90 );

    foo ( & test );
    return 0;
}
```

- ☒ Program funguje, ale neuvolní alokovanou paměť
 - ☒ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☒ Program funguje zcela správně, zobrazí: 9290
 - ☒ Program nepůjde zkompileovat
-

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x ) { m_X = x; }
        virtual ~A ( void ) { }
        friend ostream & operator << ( ostream & os, const A & x )
        { return os << x . m_X; }
    private:
        int m_X;
};

class B : public A
{
    public:
```

```

        B ( int x, int y ) : A ( x ) { m_Y = y; }
    friend ostream & operator << ( ostream & os, const B & x )
    { os << (const A &) x; return os << x . m_Y; }
private:
    int m_Y;
};

void foo ( A & val )
{
    cout << val;
}

int main ( void )
{
    B test ( 35, 57 );

    foo ( test );
    return 0;
}

```

- ☒ Program půjde zkompilevat, ale po spuštění může spadnout/spadne
 - ☐ Program nepůjde zkompilevat
 - ☒ Program funguje zcela správně, zobrazí: 35
 - ☐ Program funguje, ale neuvolní alokovanou paměť
-

Co zobrazí následující kód?

```

class A
{
public:
    A ( int x = 60 ) { m_X = new int (x); }
    A ( const A & src ) { m_X = new int ( *src . m_X ); }
    virtual ~A ( void ) { delete m_X; }
    virtual void print ( void ) const { cout << *m_X; }
private:
    int *m_X;
};

class B : public A
{
public:
    B ( int x, int y ) : A ( x ) { m_Y = new int (y); }
    B ( const B & src ) { m_Y = new int (*src.m_Y); }
    virtual ~B ( void ) { delete m_Y; }
    virtual void print ( void ) const { A::print (); cout << *m_Y; }
private:
    int *m_Y;
};

void foo ( const B val )
{
    val . print ( );
}

int main ( void )
{

```

```
B test ( 44, 94 );

foo ( test );
return 0;
}
```

- ☐ Program funguje, ale neuvolní alokovanou paměť
 - ☒ Program funguje zcela správně, zobrazí: 6094
 - ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☐ Program nepůjde zkompileovat
-

Co zobrazí následující kód?

```
class A
{
public:
    A ( int x = 50 ) { m_X = x; }
    void print ( void ) const { cout << m_X; }
private:
    int m_X;
};

class B : A
{
public:
    B ( int x, int y ) { m_Y = y; }
    void print ( void ) const { A::print (); cout << m_Y; }
private:
    int m_Y;
};

void foo ( B & val )
{
    val . print ( );
}

int main ( void )
{
    B test ( 44, 94 );

    foo ( test );
    return 0;
}
```

- ☒ Program funguje zcela správně, zobrazí: 5094
 - ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☐ Program nepůjde zkompileovat
 - ☐ Program funguje, ale neuvolní alokovanou paměť
-

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x = 54 ) { m_X = new int (x); }
        A ( const A & src ) { m_X = new int ( *src . m_X ); }
        virtual ~A ( void ) { delete m_X; }
        virtual void print ( void ) const { cout << *m_X; }
    private:
        int *m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) : A ( x ) { m_Y = new int (y); }
        B ( const B & src ) : A ( src ) { m_Y = new int (*src.m_Y); }
        ~B ( void ) { delete m_Y; }
        virtual void print ( void ) const { A::print (); cout << *m_Y; }
    private:
        int *m_Y;
};

void foo ( A * val )
{
    val -> print ( );
    delete val;
}

int main ( void )
{
    foo ( new B ( 52, 84 ) );
    return 0;
}
```

- ☒ Program funguje zcela správně, zobrazí: 5284
 - ☐ Program půjde zkompilevat, ale po spuštění může spadnout/spadne
 - ☐ Program funguje, ale neuvolní alokovanou paměť
 - ☐ Program nepůjde zkompilevat
-

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x = 10 ) { m_X = x; }
        void print ( void ) const { cout << m_X; }
    private:
        int m_X;
};

class B : public A
{

```

```

    public:
        B ( int x, int y ) { m_Y = y; }
        void print ( void ) const { A::print (); cout << m_Y; }
    private:
        int m_Y;
};

void foo ( A val )
{
    val . print ( );
}

int main ( void )
{
    B test ( 11, 19 );

    foo ( test );
    return 0;
}

```

- ☐ Program funguje, ale neuvolní alokovanou paměť
 - ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☒ Program funguje zcela správně, zobrazí: 10
 - ☐ Program nepůjde zkompileovat
-

Co zobrazí následující kód?

```

class A
{
    public:
        A ( void ) { cout << *this; }
        virtual ~A ( void ) { }
        virtual void print ( ostream & os ) const { os << "A:"; }
        friend ostream & operator << ( ostream & os, const A & x )
        { x . print ( os ); return os; }
};

class B : public A
{
    public:
        B ( int x ) : m_X ( x ) { cout << *this; }
        virtual void print ( ostream & os ) const { os << "B:" << m_X; }
    private:
        int m_X;
};

int main ( void )
{
    B test ( 57 );
    return 0;
}

```

- ☒ Program nepůjde zkompileovat
- ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne

- ☐ Program funguje zcela správně, zobrazí: A:B:57
 - ☐ Program nespadne, ale výstup není definovaný
 - ☐ Program funguje, ale neuvolní alokovanou paměť
-

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x ) { m_X = x; }
        virtual ~A ( void ) { }
        virtual void print ( ostream & os ) const { os << m_X; }
        friend ostream & operator << ( ostream & os, const A & x )
        { x . print ( os ); return os; }
    private:
        int m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) : A ( x ) { m_Y = y; }
        virtual void print ( ostream & os ) const { A::print ( os ); os << m_Y; }
    friend ostream & operator << ( ostream & os, const B & x )
    { os << "B:"; x . print ( os ); return os; }
    private:
        int m_Y;
};

void foo ( A val )
{
    cout << val;
}

int main ( void )
{
    B test ( 35, 78 );

    foo ( test );
    return 0;
}
```

- ☐ Program funguje, ale neuvolní alokovanou paměť
 - ☐ Program funguje zcela správně, zobrazí: 35
 - ☐ Program nepůjde zkompileovat
 - ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
-

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x ) { m_X = x; }
        virtual ~A ( void ) { }
        virtual void print ( void ) const { cout << m_X; }
    private:
        int m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) : A ( x ) { m_Y = y; }
        virtual void print ( void ) const { A::print (); cout << m_Y; }
    private:
        int m_Y;
};

void foo ( A val )
{
    val . print ( );
}

int main ( void )
{
    B test ( 42, 41 );

    foo ( test );
    return 0;
}
```



Program nepůjde zkompileovat



Program funguje zcela správně, zobrazí: 42



Program funguje, ale neuvolní alokovanou paměť



Program půjde zkompileovat, ale po spuštění může spadnout/spadne

Co zobrazí následující kód?

```
class A
{
    public:
        A ( void ) { cout << *this; }
        virtual ~A ( void ) { }
        virtual void print ( ostream & os ) const { os << "A: "; }
        friend ostream & operator << ( ostream & os, const A & x )
        { x . print ( os ); return os; }
};

class B : public A
{
    public:
        B ( int x ) { m_X = x; }
```



```

        virtual void print ( ostream & os ) const { os << "B:" << m_X; }
    private:
        int m_X;
};

int main ( void )
{
    B test ( 53 );
    return 0;
}

```

- ☐ Program nespadne, ale výstup není definovaný
 - ☐ Program nepůjde zkompileovat
 - ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☒ Program funguje zcela správně, zobrazí: **A:**
 - ☐ Program funguje, ale neuvolní alokovanou paměť
-

Co zobrazí následující kód?

```

class A
{
    public:
        A ( int x ) { m_X = x; }
        void print ( void ) const { cout << m_X; }
    private:
        int m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) : A ( x ) { m_Y = y; }
        void print ( void ) const { A::print (); cout << m_Y; }
    private:
        int m_Y;
};

void foo ( A & val )
{
    val . print ( );
}

int main ( void )
{
    B test ( 59, 58 );

    foo ( test );
    return 0;
}

```

- ☐ Program nepůjde zkompileovat
- ☐ Program funguje, ale neuvolní alokovanou paměť

☒ Program funguje zcela správně, zobrazí: 59

☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne

Co zobrazí následující kód?

```
class A
{
    public:
        A ( void ) { cout << *this; }
        virtual ~A ( void ) { }
        virtual void print ( ostream & os ) const = 0;
        friend ostream & operator << ( ostream & os, const A & x )
        { x . print ( os ); return os; }
};

class B : public A
{
    public:
        B ( int x ) { m_X = x; }
        virtual void print ( ostream & os ) const { os << "B:" << m_X; }
    private:
        int m_X;
};

int main ( void )
{
    B test ( 95 );
    return 0;
}
```

☐ Program nespadne, ale výstup není definovaný

☐ Program funguje zcela správně, zobrazí:

☐ Program nepůjde zkompileovat

☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne

☐ Program funguje, ale neuvolní alokovanou paměť

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x ) { m_X = x; }
        virtual ~A ( void ) { }
        virtual void print ( ostream & os ) const { os << m_X; }
        friend ostream & operator << ( ostream & os, const A & x )
        { x . print ( os ); return os; }
    private:
        int m_X;
};
```

```

class B : public A
{
    public:
        B ( int x, int y ) : A ( x ) { m_Y = y; }
        virtual void print ( ostream & os ) const { A::print ( os ); os << m_Y; }
}

friend ostream & operator << ( ostream & os, const B & x )
{ os << "B:"; x . print ( os ); return os; }

private:
    int m_Y;
};

void foo ( B & val )
{
    cout << val;
}

int main ( void )
{
    B test ( 66, 23 );

    foo ( test );
    return 0;
}

```



Program funguje, ale neuvolní alokovanou paměť



Program půjde zkompileovat, ale po spuštění může spadnout/spadne



Program nepůjde zkompileovat



Program funguje zcela správně, zobrazí: B:6623

Co zobrazí následující kód?

```

class A
{
    public:
        A ( void ) { cout <<"A:"; }
        virtual ~A ( void ) { }
        virtual void print ( ostream & os ) const = 0;
        friend ostream & operator << ( ostream & os, const A & x )
        { x . print ( os ); return os; }
};

class B : public A
{
    public:
        B ( int x ) : m_X ( x ) { cout << *this; }
        virtual void print ( ostream & os ) const { os << "B:" << m_X; }
    private:
        int m_X;
};

int main ( void )
{
    B test ( 98 );
    return 0;
}

```

- ☐ Program nepůjde zkompileovat
 - ☒ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☐ Program nespadne, ale výstup není definovaný
 - ☐ Program funguje, ale neuvolní alokovanou paměť
 - ☒ Program funguje zcela správně, zobrazí: A:B:98
-

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x ) { m_X = x; }
        virtual ~A ( void ) { }
        friend ostream & operator << ( ostream & os, const A & x )
        { return os << x . m_X; }
    private:
        int m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) : A ( x ) { m_Y = y; }
        friend ostream & operator << ( ostream & os, const B & x )
        { os << (const A &) x; return os << x . m_Y; }
    private:
        int m_Y;
};

void foo ( A val )
{
    cout << val;
}

int main ( void )
{
    B test ( 65, 81 );

    foo ( test );
    return 0;
}
```

- ☒ Program funguje zcela správně, zobrazí: 65
 - ☐ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☐ Program funguje, ale neuvolní alokovanou paměť
 - ☐ Program nepůjde zkompileovat
-

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x ) { m_X = x; }
        virtual ~A ( void ) { }
        virtual void print ( ostream & os ) const { os << m_X; }
        friend ostream & operator << ( ostream & os, const A & x )
        { x . print ( os ); return os; }
    private:
        int m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) : A ( x ) { m_Y = y; }
        virtual void print ( ostream & os ) const { A::print ( os ); os << m_Y; }
}

friend ostream & operator << ( ostream & os, const B & x )
{ os << "B:"; x . print ( os ); return os; }
private:
    int m_Y;
};

void foo ( A & val )
{
    cout << val;
}

int main ( void )
{
    B test ( 78, 42 );

    foo ( test );
    return 0;
}
```



Program nepůjde zkompileovat



Program funguje zcela správně, zobrazí: 7842



Program funguje, ale neuvolní alokovanou paměť



Program půjde zkompileovat, ale po spuštění může spadnout/spadne

Co zobrazí následující kód?

```
class A
{
    public:
        A ( int x = 34 ) { m_X = new int (x); }
        A ( const A & src ) { m_X = new int ( *src . m_X ); }
        ~A ( void ) { delete m_X; }
        virtual void print ( void ) const { cout << *m_X; }
    private:
        int *m_X;
}
```

```

};

class B : public A
{
public:
    B ( int x, int y ) : A ( x ) { m_Y = new int (y); }
    B ( const B & src ) : A ( src ) { m_Y = new int (*src.m_Y); }
    virtual ~B ( void ) { delete m_Y; }
    virtual void print ( void ) const { A::print (); cout << *m_Y; }
private:
    int *m_Y;
};

void foo ( A * val )
{
    val -> print ( );
    delete val;
}

int main ( void )
{
    foo ( new B ( 74, 47 ) );
    return 0;
}

```



Program nepůjde zkompilevat



Program funguje, ale neuvolní alokovanou paměť



Program půjde zkompilevat, ale po spuštění může spadnout/spadne



Program funguje zcela správně, zobrazí:

Co zobrazí následující kód?

```

class A
{
public:
    A ( int x = 60 ) { m_X = x; }
    void print ( void ) const { cout << m_X; }
private:
    int m_X;
};

class B : public A
{
public:
    B ( int x, int y ) { m_Y = y; }
    void print ( void ) const { A::print (); cout << m_Y; }
private:
    int m_Y;
};

void foo ( A & val )
{
    val . print ( );
}

int main ( void )

```

```

{
    B test ( 84, 42 );

    foo ( test );
    return 0;
}

```

- ☒ Program půjde zkompileovat, ale po spuštění může spadnout/spadne
 - ☒ Program funguje zcela správně, zobrazí: 60
 - ☒ Program funguje, ale neuvolní alokovanou paměť
 - ☒ Program nepůjde zkompileovat
-

```

class A
{
    public:
        A ( int x = 86 ) { m_X = x; }
        void print ( void ) const { cout << m_X; }
    private:
        int m_X;
};

class B : public A
{
    public:
        B ( int x, int y ) { m_Y = y; }
        void print ( void ) const { A::print (); cout << m_Y; }
    private:
        int m_Y;
};

void foo ( B val )
{
    val . print ( );
}

int main ( void )
{
    B test ( 43, 30 );

    foo ( test );
    return 0;
}

```

- ☒ Program nepůjde zkompileovat
- ☒ Program funguje zcela správně, zobrazí: 8630
- ☒ Program funguje, ale neuvolní alokovanou paměť
- ☒ Program půjde zkompileovat, ale po spuštění může spadnout/spadne