

## Program 1 (dst)

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

class Baza {
protected: //1
int a;

public:
Baza(int x=5, int y=2) : a{x*y} //2
    {cout<<"\nBaza konstruktor\n" ; }
~Baza( )
    { cout<<"Baza destruktor\n";}
void wypisz( ) const
    {cout<<"( "<<a<<" )\n"; }
void ustaw_a(int arg) {a+=arg ;}
};

class Pochodna: public Baza {
private: //3
int b;
double c;

public:
Pochodna(int p, int q, double r):Baza{p,p}, b{q}, c{r} //4
    {cout<<"\nPochodna konstruktor\n";}
~Pochodna( ) { cout<<"Pochodna destruktor\n";}
void ustaw_c(double arg) {c=b/arg;}
void wypisz( ) const;
};

void Pochodna::wypisz( ) const {
    Baza::wypisz();
    cout<<" "<<b<<" "<<c<<endl;
}

int main( ) {
    Baza obiekt1{5};
    obiekt1.wypisz( );
    obiekt1.ustaw_a(3);
    obiekt1.wypisz( );
    Pochodna obiekt2{2, 7, 3.4};
    obiekt2.wypisz( );
    obiekt2.ustaw_a(5);
    obiekt2.ustaw_c(10);
    obiekt2.wypisz( );
    return 0;
}
```

- wyr  
- sło  
(m  
wir  
rzu  
dzi  
dy  
- sło  
- dzi  
wir

Zada  
są za  
dost

Baza konstruktor

( 10 )

( 13 )

Baza konstruktor

Pochodna konstruktor

( 4 )

7; 3.4

( 9 )

7; 0.7

Pochodna destruktor

Baza destructor

Baza destructor

Process returned 0 (0x0) execution time : 0.065 s

Press any key to continue.

■

## Program 2 (dst)

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

class Baza
{
protected: //1
int a;
double b;
public:
Baza(int x, double y) : a{x}, b{y}
    {cout<<"\nkonstruktor "<<a<<" "<<b<<"\n" ; }
~Baza( )
    {cout<<"destruktor "<<a<<" "<<b <<endl;}
void wypisz( ) const
    {cout<<endl<<a<<" # "<<b<<endl; }
void zmiany(int arg)
    {a=arg ; b+=arg;}
};

class Pochodna: public Baza //2
{
private:
int c;
double d;

public:
Pochodna(int p, double q):Baza{p,q}, c{p+1}, d{q/10} //3
    {cout<<"\nKONSTRUKTOR "<<c<<" "<<d<<"\n";}
~Pochodna( )
    {cout<<"DESTRUKTOR "<<c<<" "<<d<<endl;}
void zmiany(int arg) { c*=arg; d+= d ;}
void wypisz( ) const;
};

void Pochodna::wypisz( ) const
```

```

{
cout<<endl<< a<<"", "<< b
    <<"", "<< c<<"", "<< d<<endl;
}

```

```

int main(){
    Baza obiekt1{5, 4.9};
    obiekt1.zmiany(3);
    obiekt1.wypisz( );
    Pochodna obiekt2{2, 11.5};
    obiekt2.wypisz( );
    obiekt2.zmiany(-2);
    obiekt2.wypisz( );
    obiekt2.Baza::zmiany(4);
    obiekt2.wypisz( );
    obiekt2.Baza::wypisz( );
    return 0;
}

```

```

konstruktor 5 4.9

```

```

3 # 7.9

```

```

konstruktor 2 11.5

```

```

KONSTRUKTOR 3 1.15

```

```

2, 11.5, 3, 1.15

```

```

2, 11.5, -6, 2.3

```

```

4, 15.5, -6, 2.3

```

```

4 # 15.5

```

```

DESTRUKTOR -6 2.3

```

```

destruktor 4 15.5

```

```

destruktor 3 7.9

```

```

Process returned 0 (0x0)   execution time : 0.066 s
Press any key to continue.

```

### Program 3 (dst)

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

class X
{
protected:
int a;
public:
X(int x) : a{x} { cout<<a<<" konstruktor X\n";}
~X( ) { cout<< a<<" destruktor X\n";}
void metoda(int aa) { a=aa;}
void wypisz( ) const {cout<<"\n( "<< a<<" )\n"; }
};

class Y : public X
{
protected:
int b;
public:
Y(int x,int y):X{x},b{y} { cout<< a<<" "<< b<<" konstruktor Y\n";}
~Y( ) {cout<< a<<" "<< b<<" destruktor Y\n";}
void metoda(int x) { b= b+x;}
void wypisz( ) const {X::wypisz( ); cout<< b<<endl; }
};

int main()
{
X o1{45};
o1.wypisz();
o1.metoda(-4);
o1.wypisz();
Y o2{1,7};
o2.wypisz();
o2.metoda(3);
o2.wypisz();
X *wsk1=new X{23};
X *wsk2=new Y{2,4};
wsk1->wypisz();
wsk2->wypisz();
wsk1->metoda(9);
wsk2->metoda(12);
wsk1->wypisz();
wsk2->wypisz();
delete wsk1; delete
wsk2; return 0;
}
```

45 konstruktor X

( 45 )

( -4 )

1 konstruktor X

1 7 konstruktor Y

( 1 )

7

( 1 )

10

23 konstruktor X

2 konstruktor X

2 4 konstruktor Y

( 23 )

( 2 )

( 9 )

( 12 )

9 destruktor X

12 destruktor X

1 10 destruktor Y

1 destruktor X

-4 destruktor X

Process returned 0 (0x0) execution

Press any key to continue.

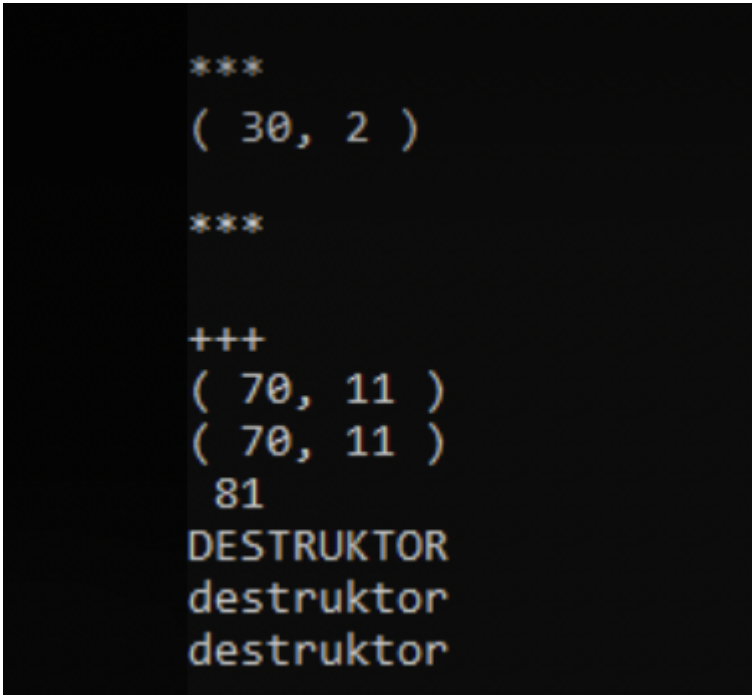
■

**Program 4 (dst)** #include <iostream> using namespace std;

```
class X
{
protected:
int a, b;

public:
X(int x=50, int y=6) : a{x}, b{y} {cout<<"\n***\n" ; }
~X( ) { cout<<"destruktor\n";}
void wypisz( ) const {cout<<"( "<< a<<" , "<< b<<" )\n"; }
};

class Y: public X
{
private:
int c;

public:
Y(int p, int q=11):X{p,q}, c{p+q} {cout<<"\n+++<div data-bbox="122 549 594 858" data-label="Code-Block">

```
***
( 30, 2 )

***

+++
( 70, 11 )
( 70, 11 )
81
DESTRUKTOR
destruktor
destruktor
```


```

\* Czy w linii oznaczonej liczbą 4 można pominąć słowo kluczowe `const`? Odpowiedź uzasadnij.

Czy w linii oznaczonej liczbą 4 można pominąć słowo kluczowe `const`?

- Nie, w linii oznaczonej liczbą 4 nie można pominąć słowa kluczowego `const`. Jest to część deklaracji funkcji w klasie `x`, a później przesłonięta w klasie `y`. Omitowanie `const` w klasie `y` spowoduje błąd kompilacji, ponieważ przesłonięcie funkcji musi mieć taką samą sygnaturę, co obejmuje także specyfikator `const` dla funkcji `wypisz`.

#### Program 6

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

class X
{
protected:
int a;
public:
X(int x=8):a(-x) {cout<<"konstruktor X\n";} // 1
virtual ~X() {cout<<"destruktor X\n";} // 2
virtual void wypisz() const {cout<<a<<"\n";}
};

class Y:public X
{
protected:
int b;
public:
Y(int x, int y):X(x), b(y) {cout<<"konstruktor Y\n";}
~Y() {cout<<"destruktor Y\n";}
void wypisz() const {X:wypisz();cout<<"-> "<<b<<"\n";}
};

class YY: public X {}; //3
```

```
int main()
{
X *a=new X(-23);
X *b=new Y(2, 4);
X *c=new YY;
a->wypisz();
b->wypisz();
c->wypisz();
delete a;
delete b;
delete c;
return 0;
}
```

Pytania:

\* Czy w definicji konstruktora klasy `X` w linii 1 można pominąć wartość domyślną argumentu?

Odpowiedź uzasadnij.

Czy w definicji konstruktora klasy `X` w linii 1 można pominąć wartość domyślną argumentu?

\* Tak, wartość domyślną argumentu w linii 1 można pominąć, ale zastosowanie jej jest dość przydatne.

\* W przypadku, gdy konstruktor jest wywoływany bez podania argumentu, wartość domyślna s zostanie użyta. Dzięki temu konstruktor może być wywołany zarówno bez argumentu, jak i z argumentem.

\* Napisz, jaki będzie wynik pracy powyższego programu, jeśli w linii 2 pominieśmy słowo kluczowe `virtual`.

Napisz, jaki będzie wynik pracy powyższego programu, jeśli w linii 2 pominieśmy słowo kluczowe `virtual`.

\* Jeśli pominieśmy słowo kluczowe `virtual` w linii 2, to destruktor klasy `y` nie będzie wirtualny. W rezultacie, podczas usuwania obiektu typu `x`, wskazującego na obiekt klasy `y`, destruktor klasy `y` nie zostanie wywołany. To może prowadzić do wycieków pamięci i błędów w zarządzaniu zasobami.

\* Uczyń z metody `wypisz` metodę czysto wirtualną oraz wprowadź niezbędne zmiany w programie, tak żeby program skompilować i uruchomić? Napisz, jaki będzie wynik programu po

```
konstruktor X
konstruktor X
konstruktor Y
konstruktor X
23
-2
-> 4
-8
destruktor X
destruktor Y
destruktor X
destruktor X
```



## Program 7

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

class X {
protected:
int a;
public:
X(int x=2):a{x} {cout<<"konstruktor X\n";}
virtual ~X() {cout<<"destruktor X\n";} //0
virtual void wypisz() const {cout<<a<<"\n";} //1
};

class Y:public X {
protected:
int b;
public:
Y(int x, int y):X{x}, b{y} {cout<<"konstruktor Y\n";}
~Y() {cout<<"destruktor Y\n";}
void wypisz() const {cout<<a<<" "<<b<<"\n";}
};

class YY:public X {};

void wywolaj(const X& arg) //2
{arg.wypisz(); }

int main()
{
X *a[3]={new X{23}, new Y{2, 4}, new YY}; // 3
for (int i=0; i<3; ++i) wywolaj(*a[i]);
for (int i=0; i<3; ++i) delete a[i];
return 0;
}
```

ch05\_01\_01\_testing\_conceptive

konstruktor X

konstruktor X

konstruktor Y

konstruktor X

23

2 4

2

destruktor X

destruktor Y

destruktor X

destruktor X

## Program 8

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

class X
{
protected:
int a;
public:
X(int x=2):a{x} {cout<<"#";}
virtual ~X() {cout<<"destruktor X\n";}
virtual void wypisz() const {cout<<a<<"\n";} //1
};

class Y: public X //2
{
protected:
int b;
public:
Y(int x, int y):X{x}, b{y} {cout<<"$\n";}
~Y() {wypisz();cout<<"destruktor Y\n";}
void wypisz() const {cout<<a<<" "<<b<<"\n";}
virtual void zmiany(int x, int y) {a+=x; b=y;} //3
};

class YY: public Y
{
int c;
public:
YY(int x):Y{x,-x}, c{2*x} {cout<<"%\n";}
void zmiany(int x, int y) {a=y; b=x; c=x*y;}
void wypisz() const {cout<<a<<" "<<b<<" "<<c<<"\n";}
};

void wywolaj(X& arg) //4
{
cout<<"\n***\n";
arg.wypisz();
}
```

```

if (Y *wsk=dynamic_cast<Y*>(&arg))
{
    wsk->zmiany(3,2);
    wsk->wypisz();
}

```

```

int main()
{
    X *a[3]={new X{23}, new Y{2, 7}, new YY{6}};
    for (int i=0; i<3; ++i) wywołaj(*a[i]);
    for (int i=0; i<3; ++i) delete a[i];
    return 0;
}

```

```

##$
#$
%

```

```

***

```

```

23

```

```

***

```

```

2 7

```

```

5 2

```

```

***

```

```

6 -6 12

```

```

2 3 6

```

```

destruktor X

```

```

5 2

```

```

destruktor Y

```

```

destruktor X

```

```

2 3

```

```

destruktor Y

```

```

destruktor X

```

```

Process returned 0 (0x0)    execution time : 0.064 s

```

```

Press any key to continue.

```

```

_

```