**Exercitiul 3**

var 1

class B{

int b;

public: B(int p=1){b=p;}

};

class D: public B{

int \*d;

public: D(int p){d=new int; \*d=p;}

D(const D& s):B(s){d=new int; \*d=\*(s.d);}

~D(){delete d;}

void set(int p){\*d=p;}

};

int main()

{D o1(2),o2(3);

o1=o2;o2.set(4);

return 0;

}

Explicatii : lipseste op = deci se apeleaza operatorul = implicit care copiaza bit cu bit si o1 si o2 vor avea aceasi valoare pt d;

In zona comuna se va pune 4;

Eroare la eliberearea dubla a zonei

var 1

0,1 constructor de initializare baza

0,1 nu e supraincarcat op=/construct copiere

0,1 implicit ->copiere bit cu bit

0,1 zona comuna adresata de acelasi pointer

0,1 eroare la dubla eliberare a zonei comune

------------------------------------------------------

var 2

class B{

int b;

public: B(int p=1){b=p;}

};

class D: public B{

int \*d;

public: D(int p){d=new int; \*d=p;}

D(const D& s):B(s){d=new int; \*d=\*(s.d);}

void set(int p){\*d=p;}

};

int main()

{D o1(2),o2(o1);

o1=o2;o2.set(4);

return 0;

}

Explicatii : lipseste op = deci se apeleaza operatorul = implicit care copiaza bit cu bit si o1 si o2 vor avea aceasi valoare pt d;

In zona comuna se va pune 3; lipseste si destructorul deci eroarea nu va fi semnalata la executie

var 2

0,1 constructor de initializare baza

0,1 nu e supraincarcat op=/construct copiere

0,1 implicit ->copiere bit cu bit

0,1 zona comuna adresata de acelasi pointer

0,1 nu da eroare daca nu se elibereaza zona comuna

-------------------------------------------------------

var 3

class B{

int b;

public: B(int p=1){b=p;}

};

class D: public B{

int \*d;

public: D(int p):B(p){d=new int; \*d=p;}

D(const D& s){d=new int; \*d=\*(s.d);}

D & operator=(const D & s){d=new int; \*d=\*(s.d);return \*this; }

~D(){delete d;}

void set(int p){\*d=p;}

};

int main()

{D o1(2),o2(o1);

o1=o2;

return 0;

}

Explicatii :!!! constructorul de copiere pt o2 -va apela constructorul de initializare din baza cu val implicita p=1 deci o2.b=1;

op= nu copiaza si data b deci o1.b=2;

var 3

0,1 constructor de initializare baza

0,2 constructorul de copiere explicit -nu apeleaza implicit cc de baza (o2.b=1)

0,2 op= implicit -nu apeleza implicit op= din baza (o1.b=2)

-------------------------------------------------------

var 4

class B{

int b;

public: B(int p=1){b=p;}

};

class D: public B {

int \*d;

public: D(int p):B(p){d=new int; \*d=p;}

D & operator=(const D & s){d=new int; \*d=\*(s.d);return \*this; }

~D(){delete d;}

void set(int p){\*d=p;}

};

int main()

{D o1(2),o2(o1);

o2.set(3);

return 0;

}

Explicatii : Constructor de copiere implicit , copiere bit cu bit deci o1 si o2 vor avea aceeasi valoare pt d;

in zona comuna se va pune valoarea 3;

Eroare la eliberarea dubla a zonei

var 4

0,1 constructor de initializare baza

0,1 nu e supraincarcat op=/construct copiere

0,1 implicit ->copiere bit cu bit

0,1 zona comuna adresata de acelasi pointer

0,1 eroare la dubla eliberare a zonei comune

-------------------------------------------------------

var 5

class B{

int b;

public: B(int p=1){b=p;}

};

class D: public B {

int \*d;

public: D(int p):B(p){d=new int; \*d=p;}

D & operator=(const D & s){d=new int; \*d=\*(s.d);return \*this; }

void set(int p){\*d=p;}

};

int main()

{D o1(2),o2(o1);

o2.set(3);

return 0;

}

Explicatii :Constructor de copiere implicit , copiere bit cu bit deci o1 si o2 vor avea aceeasi valoare pt d si b=2;

in zona comuna pt d se va pune valoarea 3; lipseste insa destructorul si eroarea nu va fi detectata la executie

var 5

0,1 constructor de initializare baza

0,1 nu e supraincarcat op=/construct copiere

0,1 implicit ->copiere bit cu bit

0,1 zona comuna adresata de acelasi pointer

0,1 nu da eroare daca nu se elibereaza zona comuna

-------------------------------------------------------

var 6

class B{

int b;

public: B(int p=1){b=p;}

};

class D: public B {

int \*d;

public: D(int p):B(p){d=new int; \*d=p;}

D & operator=(const D & s){d=new int; \*d=\*(s.d);return \*this; }

~D(){delete d;}

void set(int p){\*d=p;}

};

int main()

{D o1(2),o2(o1);

o1=o2;o2.set(3);

return 0;

}

Explicatii : Constructor de copiere implicit , copiere bit cu bit deci o1 si o2 vor avea aceeasi valoare pt d=2 si b=1;

!!! op= aloca zona noua pt o1.d ; doar o2.d va fi 3;

var 6

0,1 constructor de initializare baza

0,1 nu e supraincarcat op=/construct copiere

0,1 implicit ->copiere bit cu bit

0,1 zona comuna adresata de acelasi pointer

0,1 op= definit aloca zona noua -> nu e eroare