**Exercitiul 1**

0,4 nu compileaza cu explicatii

0,1 modificarea

var 1

class C{

int c;

public: C(int p=1){c=p;}

int & get()const{return c;}

};

int f(C op){return op.get();}

int main(){

C o1;

int x=f(o1);

return 0;

}

Explicatii : nu pot intoarce const int & in int &;

Modificare: adaug const la referinta intoarsa

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

var 2

class C{

int c;

public: C(int p=1){c=p;}

const int & get(){return c;}

};

int f(const C\*op){return op->get();}

int main(){

C o1;

int x=f(&o1);

return 0;

}

Explicatii: p protejeaza zona, nu poate apela metoda neconstanta;

Modificare: adaug const la metoda

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

var 3

class C{

int c;

public: C(int p=1){c=p;}

const int & get(){return c;}

};

int f(const C & op){return op.get();}

int main(){

C o1;

int x=f(o1);

return 0;

}

Explicatii: r protejeaza zona, nu poate apela metoda neconstanta;

Modificare: adaug const la metoda

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

var 4

class C{

int c;

public: C(int p=1){c=p;}

const int & get()const{return c++;}

};

int main(){

const C o1;

int x=o1.get();

return 0;

}

Explicatii: metoda constanta nu poate modifica date obiectului implicit;

Modificare: intoarce c

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

var 5

class C{

int c;

public: C(int p=1){c=p;}

const int & get(){return c;}

};

int f(const C op){return op.get();}

int main(){

C o1;

int x=f(o1);

return 0;

}

Explicatii: obiect constant nu poate apela metoda neconstanta ;

Modificare: adauga const la metoda;